



HAL
open science

Document Understanding with Deep Learning Techniques

Kim-Anh Laura Nguyen

► **To cite this version:**

Kim-Anh Laura Nguyen. Document Understanding with Deep Learning Techniques. Document and Text Processing. Sorbonne Université, 2024. English. NNT : 2024SORUS077 . tel-04626992

HAL Id: tel-04626992

<https://theses.hal.science/tel-04626992>

Submitted on 27 Jun 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT DE SORBONNE UNIVERSITÉ
Spécialité **Informatique**
École Doctorale Informatique, Télécommunications et Électronique (Paris)

**Document Understanding
with Deep Learning Techniques**
**Analyse Intelligente de Documents avec des Approches
Neuronales**

Présentée par
Kim-Anh Laura Nguyen

Dirigée par
Benjamin Piwowarski

Pour obtenir le grade de
DOCTEUR de SORBONNE UNIVERSITÉ

Présentée et soutenue publiquement le

Devant le jury composé de :

| | |
|---|--------------------------------|
| Nathalie AUSSENAC-GILLES <i>Directrice de recherche, CNRS, IRIT Toulouse</i> | Rapportrice |
| Julien VELCIN <i>Professeur des universités, Université Lyon 2</i> | Rapporteur |
| Pascale SÉBILLOT <i>Professeure des universités, INSA Rennes</i> | Examinatrice |
| François YVON <i>Directeur de recherche, Sorbonne Université Paris</i> | Examineur et Président du jury |
| Antoine GOURRU <i>Maître de conférences, Télécom Saint-Étienne</i> | Examineur |
| JULIO LABORDE <i>Ingénieur en Intelligence Artificielle, reciTAL</i> | Encadrant de thèse |
| BENJAMIN PIWOWARSKI <i>Chargé de recherche, CNRS, Sorbonne Université Paris</i> | Directeur de thèse |

ABSTRACT

The field of *Document Understanding*, which addresses the problem of solving an array of Natural Language Processing (NLP) tasks for visually-rich documents, faces challenges due to the complex structures and diverse formats of documents. Real-world documents rarely follow a strictly sequential structure. The visual presentation of a document, especially its layout, conveys rich semantic information, highlighting the crucial need for document understanding systems to include multimodal information. Despite notable advancements attributed to the emergence of Deep Learning, the field still grapples with various challenges in real-world applications. This thesis addresses two key challenges: 1) developing *efficient* and *effective* methods to encode the *multimodal* nature of documents, and 2) formulating strategies for *efficient* and *effective* processing of *long and complex* documents, considering their *visual appearance*.

Our strategy to address the first research question involves designing approaches that rely *only* on layout to build meaningful representations. Multimodal pre-trained models for Document Understanding often neglect efficiency and fail to fully capitalize on the strong correlation between text and layout. We address these issues by introducing an attention mechanism based exclusively on layout information, enabling performance improvement and attention sparsification.

Furthermore, we introduce a strategy based solely on layout to address reading order issues. While layout inherently captures the correct reading order of documents, existing pre-training methods for Document Understanding rely solely on Optical Character Recognition (OCR) or PDF parsing to establish the reading order of documents, potentially introducing inaccuracies that can impact the entire text processing pipeline. Therefore, we discard sequential position information and propose a model that strategically leverages layout information as an alternative means to determine the reading order of documents.

In addressing the second research axis, we explore the potential of leveraging layout to enhance the performance of models for tasks related to long and complex documents. The importance of document structure in information processing, particularly in the context of long documents, underscores the need for efficient modeling of layout information. To fill a notable void in resources and approaches for multimodal long document modeling, we introduce a dataset collection for summarization of long documents with consideration for their visual appearance, and present novel baselines that can handle long documents with awareness of their layout.

RÉSUMÉ

Le domaine de l'Analyse Intelligente de Documents (*Document Understanding*), dédié au traitement automatique des documents, fait face à des défis liés à leurs structures complexes et formats variés. Les documents possèdent rarement une structure strictement séquentielle. Leur présentation visuelle, notamment leur mise en page, contient une information sémantique riche, soulignant la nécessité d'inclure des informations multimodales dans les systèmes d'analyse intelligente de documents. Malgré des progrès notables découlant de l'avènement de l'apprentissage profond, le domaine doit relever des défis importants. Cette thèse traite deux défis clés : 1) développer des méthodes *performantes* et *efficaces* pour encoder la nature *multimodale* des documents, et 2) formuler des stratégies pour le traitement *performant* et *efficace* de documents *longs*, en tenant compte de leur *apparence visuelle*.

Pour répondre à la première question de recherche, nous développons des approches basées *uniquement* sur les informations de mise en page afin de construire des représentations pertinentes pour les tâches subséquentes. Les modèles pré-entraînés multimodaux existants étant développés sans considération d'efficacité et n'exploitant pas pleinement la forte corrélation entre le texte et la mise en page, nous présentons un mécanisme d'attention exclusivement basé sur la mise en page, permettant d'améliorer les performances et de rendre l'attention plus parcimonieuse.

De plus, nous proposons une stratégie basée exclusivement sur la mise en page pour résoudre les problèmes d'ordre de lecture. Bien que la mise en page capture l'ordre de lecture des documents, les méthodes de pré-entraînement existantes dédiées à l'analyse intelligente de documents s'appuient uniquement sur la Reconnaissance Optique de Caractères (OCR) ou l'analyse de PDF pour établir l'ordre de lecture des documents, introduisant potentiellement des erreurs qui peuvent impacter l'ensemble du processus de traitement du texte. Par conséquent, nous proposons un modèle qui exploite uniquement les informations de mise en page pour déterminer l'ordre de lecture des documents.

Dans le cadre du deuxième axe de recherche, nous explorons le potentiel de la mise en page pour améliorer les performances des modèles pour les tâches liées aux documents longs et complexes. Pour pallier le manque de ressources et de méthodes pour la modélisation multimodale de documents longs, nous construisons une collection de jeux de données pour le résumé de documents longs avec

prise en compte de leur apparence visuelle, et introduisons de nouveaux modèles pouvant traiter des documents longs en tenant compte de leur mise en page.

REMERCIEMENTS

Il y a plus de trois ans, en plein confinement pendant la pandémie de COVID-19, j'ai commencé ma thèse. Maintenant que cette belle aventure touche à sa fin, je souhaite exprimer ma gratitude à toutes celles et ceux qui ont contribué à sa réalisation et m'ont soutenue tout au long de ce parcours.

Je tiens tout d'abord à remercier mon directeur de thèse, Benjamin Piwowarski. Je te suis reconnaissante de m'avoir acceptée en thèse et de m'avoir fait confiance tout au long de ces années. Merci pour ta gentillesse et pour avoir toujours cru en mon potentiel. Merci d'avoir partagé ton expertise pour me former en tant que chercheuse, ainsi que de m'avoir guidée en me suggérant de nombreuses pistes de recherche.

Je souhaite également remercier mes encadrants de thèse chez reciTAL, Jacopo Staiano et Julio Laborde. Merci, Jacopo, pour m'avoir accueillie et formée dans le monde académique et professionnel ; j'ai appris beaucoup de choses grâce à toi. Merci, Julio, pour tes conseils, tes propositions et ton soutien. Merci également pour ta gentillesse et pour m'avoir fait me sentir rapidement à l'aise et intégrée dans l'équipe.

Merci à Nathalie Aussenac-Gilles, Julien Velcin, Pascale Sébillot, François Yvon et Antoine Gourru d'avoir accepté de faire partie de mon jury de thèse et d'avoir pris le temps d'évaluer mon travail.

Un grand merci à mes collègues chez reciTAL, dont la bonne humeur a été d'une grande aide. Merci également à Gilles Moyse pour sa confiance et pour m'avoir acceptée en tant que doctorante au sein de l'entreprise. Merci, Gaud, pour ta bienveillance, ton attention, et pour avoir si bien pris en charge mon dossier CIFRE tout au long de ma thèse.

À mes amis – en particulier Léopoldine et Rupsi, pour leur soutien inconditionnel. Merci pour votre écoute, pour avoir été à mes côtés, pour avoir toujours cru en moi et pour m'avoir donné de la force tout au long de ces années.

À Paul, qui a été un véritable pilier toutes ces années. Merci de m'avoir accompagnée à chacune des étapes de cette thèse, dans les moments de joie comme dans les moments difficiles. Merci de toujours trouver le moyen de me faire rire et de me remonter le moral. Merci d'avoir patiemment écouté mes innombrables répétitions de présentation, au point de commencer à les connaître par cœur toi aussi. Merci d'avoir toujours été fier de moi.

À toute ma famille, à qui je dois tant. Merci pour votre amour inconditionnel. À mes parents, merci pour votre soutien inébranlable. Merci d'avoir toujours été si fiers de moi et de m'avoir constamment encouragée à me surpasser. À mon frère et à ma sœur, merci pour tous vos encouragements et votre enthousiasme vis-à-vis de mon travail. À ma grand-mère, Bà Ngoai, merci pour ton humeur rayonnante qui me donne toujours le sourire.

À mon grand-père, Ông Ngoai, à qui je dédie cette thèse. Merci d'avoir toujours cru en moi et de m'avoir encouragée à poursuivre de longues études. Merci de m'avoir sans cesse inspirée avec ta détermination et ta joie de vivre. Merci pour les précieuses leçons de vie que tu m'as transmises et pour tout ce que tu as fait pour nous. J'aurais tellement aimé partager ces moments avec toi, je sais que tu aurais été immensément fier.

CONTENTS

| | |
|---|------|
| ABSTRACT | i |
| RÉSUMÉ | iii |
| REMERCIEMENTS | v |
| CONTENTS | vii |
| LIST OF FIGURES | ix |
| LIST OF TABLES | xiii |
| ACRONYMS | xvii |
| 1 INTRODUCTION | 1 |
| 2 LANGUAGE MODELING | 7 |
| 2.1 Language Modeling | 7 |
| 2.2 Neural Language Models | 20 |
| 3 TRANSFORMER ARCHITECTURE & PRE-TRAINED LANGUAGE MODELS | 27 |
| 3.1 Transformers | 29 |
| 3.2 Transformer-based Pre-trained Language Models | 39 |
| 3.3 Long-range Modeling | 50 |
| 3.4 Conclusion | 58 |
| 4 DOCUMENT UNDERSTANDING | 59 |
| 4.1 Document Understanding Tasks and Datasets | 61 |
| 4.2 Task-specific Deep Learning Models For Document Understanding | 65 |
| 4.3 Deep Fusion of Modalities via General-purpose Multimodal Pre-training | 68 |
| 4.4 Conclusion | 82 |
| 5 LEVERAGING LAYOUT FOR SPARSE ATTENTION | 85 |
| 5.1 Preliminary Experiments: Human Evaluation | 87 |
| 5.2 Skim-Attention: A Novel Layout-Aware Attention Mechanism . . . | 88 |
| 5.3 Experiments | 94 |
| 5.4 Results | 97 |
| 5.5 Conclusion | 102 |
| 6 LEVERAGING LAYOUT TO AVOID READING ORDER ISSUES | 103 |
| 6.1 Reconstructing Positional Information from 2D Positions | 107 |
| 6.2 Experiments | 115 |
| 6.3 Results and Discussion | 121 |
| 6.4 Conclusion | 124 |
| 7 LEVERAGING LAYOUT TO DEAL WITH LONG AND LAYOUT-RICH DOCUMENTS | 125 |
| 7.1 Datasets Construction | 129 |

| | |
|--|-----|
| 7.2 Experiments | 136 |
| 7.3 Results and Discussion | 138 |
| 7.4 Conclusion | 145 |
| 8 CONCLUSION | 147 |
| 8.1 Summary of Contributions | 147 |
| 8.2 Perspectives | 150 |
| BIBLIOGRAPHY | 153 |

LIST OF FIGURES

| | |
|---|--|
| CHAPTER 1: INTRODUCTION | 1 |
| CHAPTER 2: LANGUAGE MODELING | 7 |
| Figure 2.1 | Architecture of Bengio et al. (2000)'s Neural Language Model. 20 |
| Figure 2.2 | Compressed (left) and unfolded (right) views of a Recurrent Neural Network. 22 |
| CHAPTER 3: TRANSFORMER ARCHITECTURE & PRE-TRAINED LANGUAGE MODELS | 27 |
| Figure 3.1 | Multi-head attention. 33 |
| Figure 3.2 | Transformer encoder. 34 |
| Figure 3.3 | Transformer decoder. 35 |
| Figure 3.4 | Masked Language Modeling illustrated with the sentence "how are you doing today". Each token is mapped to a distribution over the vocabulary V . Depicted for the masked token "you". For the sake of clarity, segment embeddings and positional encodings are not shown. 40 |
| Figure 3.5 | Illustration of beam search, where $K = 2$. From Von Platen (2020). 44 |
| Figure 3.6 | Attention mechanisms used in BigBird. Illustration from Zaheer et al. (2020). 51 |
| CHAPTER 4: DOCUMENT UNDERSTANDING | 59 |
| Figure 4.1 | Document Understanding tasks and examples. 61 |
| Figure 4.2 | Layout encoding process in LayoutLM (Yiheng Xu et al. 2020). 71 |
| Figure 4.3 | Comparisons of LayoutLMv3 (Y. Huang et al. 2022), DocFormer (Appalaraju et al. 2021) and SelfDoc (P. Li et al. 2021) on image embedding construction. Adapted from Y. Huang et al. (2022). 76 |
| CHAPTER 5: LEVERAGING LAYOUT FOR SPARSE ATTENTION | 86 |
| Figure 5.1 | Documents selected for our preliminary cognitive experiment. 88 |

| | | |
|---|---|-----|
| Figure 5.2 | Skimformer model architecture. The input consists of two components: a sequence of tokens (right-hand side) and a sequence of token bounding box coordinates (left-hand side). Only the layout embeddings (left) are used to compute Skim-Attention. L denotes the number of Transformer encoder layers. Q and K are the queries and keys obtained by projecting the layout embeddings. V represents the values produced by projecting the encoder layers' textual inputs. The attention is solely based on token spatial positions and computed only once. The attention scores are then distributed to each layer of a Transformer encoder. | 91 |
| Figure 5.3 | Skimming Mask model architecture. The layout embeddings, Key and Query projections are initialized from an already pre-trained Skimformer model. By filtering the k most attended tokens for each token, the Skim-Attention scores are then converted to an attention mask and given as input to a text-based Transformer model. | 93 |
| Figure 5.4 | Model perplexity on the MIX validation set with respect to the number of optimization steps. All models are trained from scratch. | 98 |
| Figure 5.5 | Comparison of time and memory usage for LayoutLM (green), Skimformer with layout contextualizer (orange) and without (blue). Results are plotted against sequence length. | 99 |
| Figure 5.6 | Skim-Attention maps obtained on three sample documents. We consider the skim-attention matrix averaged over all the attention heads. Given a semantic unit, we plot the average attention score for each token. | 101 |
| CHAPTER 6: LEVERAGING LAYOUT TO AVOID READING ORDER ISSUES | | 104 |
| Figure 6.1 | Examples of documents for each layout category, arranged from the simplest to the most complex. | 107 |
| Figure 6.2 | Ground-truth reading order (a) compared to the reading order generated by Tesseract (b) for a sample table. Arrows emphasize the differences in reading order in the first row. Non-highlighted text indicates that it does not appear in the serialized sequence. | 109 |

| | | |
|---|--|------------|
| Figure 6.3 | Ground-truth reading order (a) compared to the reading order generated by Tesseract (b) for a document with a two-column layout. The document was cropped for better visibility. Arrows emphasize the differences in reading order. Non-highlighted text indicates that it does not appear in the serialized sequence. | 110 |
| Figure 6.4 | Distances between: the left edge of each box ($x'_0 - x_0$, in green), the right edge of the second box and the left edge of the first ($x'_1 - x_0$, in orange), and the right edge of each box ($x'_1 - x_1$, in blue). | 111 |
| Figure 6.5 | Layout2Pos Architecture. | 113 |
| Figure 6.6 | Architecture of Layout2Pos integrated into a BART model, <i>i.e.</i> , BART+Layout2Pos. The input consists of two components: a sequence of tokens (middle) and a sequence of token bounding box coordinates (left). | 114 |
| CHAPTER 7: LEVERAGING LAYOUT TO DEAL WITH LONG AND LAYOUT-RICH DOCUMENTS | | 127 |
| Figure 7.1 | Dataset Construction Process. | 130 |
| Figure 7.2 | Distribution of failure types in arXiv-Lay (top) and PubMed-Lay (bottom). | 131 |
| Figure 7.3 | Distribution of topics covered by all publishers (red) vs distribution of topics covered by publishers whose name contains the word <i>Korean</i> (blue). | 133 |
| Figure 7.4 | Distribution of research areas in arXiv-Lay (a) and HAL (b). | 135 |
| Figure 7.5 | LoRaLay evaluation interface. | 142 |
| Figure 7.6 | Benefit of using layout on arXiv-Lay (blue) and PubMed-Lay (red), defined as the difference in ROUGE-L scores between BigBird-Pegasus+Layout and BigBird-Pegasus. For each dataset, quartiles are calculated from the distributions of article lengths (a), summary lengths (b) and variance in the height of the bounding boxes (c). ROUGE-L scores are then computed per quartile range, and averaged over each range. | 144 |

LIST OF TABLES

| | |
|---|---|
| CHAPTER 1: INTRODUCTION | 1 |
| CHAPTER 2: LANGUAGE MODELING | 7 |
| CHAPTER 3: TRANSFORMER ARCHITECTURE & PRE-TRAINED LANGUAGE MODELS | 27 |
| CHAPTER 4: DOCUMENT UNDERSTANDING | 59 |
| Table 4.1 | Summary of general-purpose, multimodal pre-training document understanding models. 70 |
| CHAPTER 5: LEVERAGING LAYOUT FOR SPARSE ATTENTION | 86 |
| Table 5.1 | Average (std) time (in seconds) required to answer questions from documents, depending on whether layout is provided. 88 |
| Table 5.2 | Test perplexity on the MIX dataset after 10k optimization steps. Each model was trained from scratch. Bold denotes the best score. 97 |
| Table 5.3 | Ablation study on the MIX dataset, where perplexity on the test set is reported. All models were trained from scratch. Bold denotes the best score. 98 |
| Table 5.4 | Model performance (in %) on the DocBank-LA dataset. <i>Seq. Len</i> indicates the number of tokens attended with either standard attention (for Skimformer, BERT-based and LayoutLM-based models), or Longformer attention (for Longformer and LongSkimformer). <i>Nb Attention</i> represents the number of times attention (original and Skim-Attention) is computed and stored. <i>Total Compute</i> specifies the ratio of the final computational cost (# operations needed to compute attention) w.r.t. BERT/LayoutLM or Longformer. Each model was pre-trained from scratch on the MIX dataset, then fine-tuned on DocBank-LA. 100 |
| CHAPTER 6: LEVERAGING LAYOUT TO AVOID READING ORDER ISSUES | 104 |
| Table 6.1 | Accuracy (in %) obtained by each OCR engine, for each document layout type. 108 |

| | | |
|---|---|------------|
| Table 6.2 | Example document from FUNSD, SROIE, and CORD, accompanied by their corresponding target sequences that include the entities to be extracted paired with their corresponding keys. Best viewed in color. | 117 |
| Table 6.3 | Accuracy in predicting the next token for pairs sourced from ReadingBank, which were not used for pre-training. Selected pairs are considered "difficult", meaning that the tokens are positioned on different lines. | 121 |
| Table 6.4 | Model performance (in %) on FUNSD, SROIE, and CORD, reported for 1) the original reading order and 2) three shuffled orders (averaged). Best F1 scores for each dataset/reading order are reported in bold. | 123 |
| CHAPTER 7: LEVERAGING LAYOUT TO DEAL WITH LONG AND LAYOUT-RICH DOCUMENTS | | 127 |
| Table 7.1 | Datasets statistics. Article and summary lengths are computed in words. For KoreaScience, words are obtained via white-space tokenization. Difference between arXiv and arXiv-Lay is due to the fact that we retain the whole document, while Cohan et al. (2018) truncate it after the conclusion. | 134 |
| Table 7.2 | Datasets splits and statistics. Input and output lengths are computed in tokens, obtained using Pegasus and mBART-50's tokenizers for the English and non-English datasets, respectively. | 135 |
| Table 7.3 | ROUGE scores on arXiv-Lay and PubMed-Lay. Reported results obtained by Pegasus and BigBird-Pegasus on the original arXiv and PubMed are highlighted with a gray background. The best results obtained on arXiv-Lay and PubMed-Lay are denoted in bold. | 138 |
| Table 7.4 | Absolute ROUGE-L score differences between each pair of models, on arXiv-Lay/PubMed-Lay (column – row). | 139 |
| Table 7.5 | Quartiles calculated from the distributions of article lengths, summary lengths, and variation in the height of bounding boxes, for arXiv-Lay and PubMed-Lay. | 139 |
| Table 7.6 | ROUGE scores on the non-English datasets. The best results for each dataset are reported in bold. | 140 |
| Table 7.7 | Percent confidence obtained for the main language, for each dataset split. | 141 |

| | | |
|-----------|---|-----|
| Table 7.8 | Average human judgement scores obtained by comparing gold-truth abstracts and summaries generated by BigBird and BigBird+Layout from 50 documents sampled from arXiv-Lay and HAL. Inter-rater agreement is computed using Krippendorff's alpha coefficient, and enclosed between parentheses. Best scores are reported in bold. \uparrow means that higher is considered "better", whereas \downarrow signifies the opposite. | 143 |
| Table 7.9 | Quartiles calculated from the distributions of article lengths, summary lengths, and variation in the height of bounding boxes, for arXiv-Lay and PubMed-Lay. | 144 |

ACRONYMS

| | |
|---------------------|---|
| AI | Artificial Intelligence |
| CNN | Convolutional Neural Network |
| GCN | Graph Convolutional Network |
| ReLU | Rectified Linear Unit |
| RNN | Recurrent Neural Network |
| NLP | Natural Language Processing |
| NLI | Natural Language Inference |
| MLM | Masked Language Modeling |
| MVLM | Masked Visual-Language Modeling |
| NSP | Next Sentence Prediction |
| LSTM | Long Short Term Memory |
| NER | Named Entity Recognition |
| POS | Part-of-Speech |
| ELMo | Embeddings from Language Models |
| BERT | Bidirectional Encoder Representations from Transformers |
| RoBERTa | A Robustly Optimized BERT Pretraining Approach |
| ALBERT | A Lite BERT |
| BART | Bidirectional and Auto-Regressive Transformers |
| GPT | Generative Pre-trained Transformer |
| T ₅ | Text-to-Text Transfer Transformer |
| ETC | Extended Transformer Construction |
| LAMP _{reT} | Layout-Aware Multimodal PreTraining |
| LSPE | Learnable Sinusoidal Positional Encoding |
| LiLT | Language-independent Layout Transformer |
| BPE | Byte Pair Encoding |
| BPC | Bits-per-character |
| ROUGE | Recall-Oriented Understudy for Gisting Evaluation |
| BLEU | Bilingual Evaluation Understudy |
| SQuAD | Stanford Question Answering Dataset |

| | |
|---------|--|
| GLUE | General Language Understanding Evaluation |
| LRA | Long-Range Arena |
| SCROLLS | Standardized Comparison Over Long Language Sequences |
| FUNSD | Form Understanding in Noisy Scanned Documents |
| CORD | Consolidated Receipt Dataset |
| SROIE | Scanned Receipts OCR And Key Information Extraction |
| LSH | Locality Sensitive Hashing |
| OCR | Optical Character Recognition |
| OOV | Out Of Vocabulary |
| PLSA | Probabilistic Latent Semantic Analysis |
| LDA | Latent Dirichlet Allocation |
| LCS | Longest Common Subsequence |

INTRODUCTION

Driven by factors such as increased digitization, the adoption of electronic communication, and the expansion of online platforms, the past two decades have seen an unprecedented and ever-increasing trend in data production (Hilbert and López 2011; Clissa 2022). Notably, it is estimated that the stock of textual data is currently expanding at a rate of 7% per year (Villalobos et al. 2022). In a world saturated with information, where the volume of electronic documents keeps growing, the ability of machines to *read, understand and interpret documents*¹ with human-like proficiency becomes increasingly crucial. This question forms the core focus of our investigation into improving the automated process of reading, interpreting, and extracting meaningful information from documents, *i.e., Document Understanding*.

The foundation of digital transformation lies in automated information processing, with escalating demands for increased processing power, speed, and accuracy across multiple domains and industries such as law, business, and health-care. In the business field, electronic documents play a central role. Ranging from purchase receipts and industry reports to sales contracts and financial statements, business documents encapsulate a plethora of complex information. In this context, hyperautomation has emerged as a highly-demanded² approach to automate and optimize business document processing by introducing Artificial Intelligence (AI) technologies. Examples of successful products that have empowered a range of industries with hyperautomation technologies include Microsoft Azure AI Document Intelligence,³ Amazon Textract,⁴ and Google Document AI,⁵ among others. At the core of hyperautomation lies Document Understanding, also known as *Document Intelligence*. Encompassing the techniques used to automatically analyze and understand documents, Document Understanding is challenging due to the complex structures and varied formats of documents, the quality of scans and OCR systems used, and the diversity of knowledge domains.

1. <https://document-intelligence.github.io/DI-2022/>

2. <https://www.gartner.com/en/documents/4019586>

3. <https://azure.microsoft.com/en-us/products/ai-services/ai-document-intelligence/>

4. <https://aws.amazon.com/fr/textract/>

5. <https://cloud.google.com/document-ai?hl=en>

In contrast to mainstream NLP, which typically deals with plain text documents, the content encountered in real-world documents—such as scientific articles, news reports, or emails—is rarely strictly sequential. When creating documents, writers rely on the assumption that structured and visual formats, such as tables, graphs, or infographics, are more understandable to readers than sequential text. This preference is grounded in human visual perception and our ability to comprehend spatial context within a text. In other words, the appearance of a document, and especially its layout—defined as the spatial arrangement and organization of both graphical and textual elements—guides our reading process and conveys rich semantic information. Conventional NLP approaches do not consider this information, which may hinder their effectiveness in handling real-world documents. This underscores the critical need for document understanding systems to incorporate multimodal information, extending beyond mere textual content to encompass document layout and visual appearance.

Addressing the challenges posed by Document Understanding requires a *multi-disciplinary* approach that extends beyond NLP, encompassing fields such as Computer Vision to handle visual elements and Information Retrieval for document search and clustering. Over the past thirty years, document understanding systems have incorporated a fundamental aspect of *multimodality*, evolving from rule-based heuristics and Machine Learning approaches to methods based on *Deep Learning*. Notably, a significant breakthrough in the field has been observed with the widespread adoption of large-scale multimodal pre-training for general document-level understanding (Yiheng Xu et al. 2020). This versatile framework produces powerful *language models* with general knowledge, making them easily adaptable for various applications. Large-scale pre-training has become a cornerstone for document understanding systems, leading to a remarkable leap in performance across various tasks in the field. Yet, despite its importance for digitization, the academic literature on resources and methodologies for addressing Document Understanding remains relatively scarce. Furthermore, the field faces critical limiting factors for achieving satisfying results in practical applications.

A key challenge lies in the constraint on the input length of current large-scale pre-trained language models, restricted to a maximum of 1,024 tokens. This limitation hinders effective multi-page and cross-page understanding of long and complex documents. Furthermore, the discrepancy between high-quality annotated training data and real-world documents, commonly obtained from scanning equipment, and exhibiting lower quality, can result in sub-optimal performance. Besides, document understanding systems face challenges in practical applications due to insufficient computing resources and labeled training samples. In addition, existing document understanding tasks are often treated independently, lacking effective leveraging of correlations between tasks. Finally, accuracy in text recognition and word ordering (*i.e.*, serialization) from OCR and PDF-parsing en-

gines plays a pivotal role in downstream tasks, sometimes even more than the choice of model architecture (Borchmann et al. 2021).

Aligning with the current trend in the field, this thesis explores the use of Deep Learning methods, with a particular emphasis on general-purpose pre-training techniques, to advance automatic document understanding. Within the scope of this thesis, we cover two key challenges in the field: 1) Developing *efficient* and *effective* methods to encode the *multimodal* nature of documents, and 2) Formulating strategies to efficiently and effectively process *long and complex* documents, with consideration for their *visual appearance*.

The organization of this PhD thesis is as follows: Chapters 2 to 4 review existing literature and works relevant to our research, while Chapters 5 to 7 present our contributions to the field.

Literature review Chapter 2 explores the extensive literature that forms the foundation of Language Models, tracing their evolution from the early era of Statistical Language Models to the emergence of Neural Language Models. Chapter 3 centers on the Transformer architecture (Vaswani et al. 2017) and its application in creating powerful Pre-trained Language Models, unraveling how these foundational models have reshaped the landscape of NLP. However, the applicability of these models to long documents is hindered by the quadratic complexity of the Transformer. This complexity primarily stems from its core component, the self-attention mechanism, which necessitates each element in the sequence to acquire information from every other element in the sequence. Therefore, Chapter 3 also delves into modeling advances and architectural innovations that tackle the complexity issue of self-attention. Closer to the applicative field of this PhD thesis, Chapter 4 offers an overview of the field of Document Understanding, exploring recent advancements driven by Transformer-based pre-training techniques for deep fusion of modalities.

Leveraging layout-only information We address the research question of developing efficient and effective methods to encode the multimodal nature of documents. Our strategy involves designing approaches that rely only on layout information to build meaningful representations, aiming to enhance both the effectiveness and efficiency of document understanding processes. As existing multimodal pre-trained models for Document Understanding are developed without sufficient attention to efficiency considerations, they still grapple with the quadratic complexity of self-attention. Incorporating layout information, while beneficial, increases resource demands compared to models that exclusively handle text. Chapter 5 (*Leveraging layout for sparse attention*) focuses on *exploiting layout*

in a computationally efficient manner by investigating 1) whether attention can be determined solely from layout information, and 2) whether layout can contribute to mitigating the complexity of self-attention. Drawing inspiration from insights in cognitive science, we address these questions by introducing *Skim-Attention*, a novel attention mechanism exclusively based on document layout. This approach not only offers a novel means to integrate layout information for enhanced performance but also enables the sparsification of attention.

Furthermore, we introduce a strategy based solely on layout to address and avoid reader order issues. While layout inherently encapsulates the correct reading order of documents through visually organized content, existing pre-training methods for Document Understanding often overlook this aspect, relying solely on OCR or PDF parsing to determine the reading order of documents. Yet, accurately retrieving the reading order of visually-rich documents proves challenging, potentially compromising performance in downstream tasks. In Chapter 6 (*Leveraging layout to avoid reading order issues*), we investigate whether discarding reading order information obtained via OCR and, instead, *strategically depending on layout information to establish an alternative for the provided reading order of documents* can sustain performance in visual information extraction tasks. Based on the intuition that layout contains the information to supplement reading order, we present *Layout2Pos*—a shallow Transformer designed to learn position embeddings from layout exclusively. By avoiding reading order issues linked to the use of sequential position information, this design choice enhances the robustness of models to changes in reading order.

Processing long documents with awareness of their layout We explore the potential of leveraging layout to improve the performance of models for tasks associated with long and complex documents. The significance of document structure for information processing, as demonstrated by prior works in cognitive sciences and particularly crucial in long documents, emphasizes the need for efficient modeling of layout information. However, methods for efficient long document understanding remain under-explored. In Chapter 7 (*Leveraging layout to deal with long and layout-rich documents*), we investigate the significance of *integrating layout information into long-range modeling* for Long Text Summarization. This task, well-suited to benefit from a global context, relies on document structures to guide summary generation. To spur further research on how to incorporate multimodal information to better capture long-range dependencies, we build *LoRaLay*, a multilingual corpus of datasets designed for long document summarization with consideration for their visual appearance. These datasets, enriched with layout/visual information, along with novel baselines combining layout-aware and long-range models, provide valuable resources for exploring the use of multimodal information in long document modeling.

By developing strategies relying exclusively on layout information, we contribute to the efficient and effective encoding of the multimodal nature of documents, offering practical advancements in Document Understanding. Additionally, our investigations into leveraging layout for processing long and complex documents provide valuable resources and insights to efficiently and effectively address the challenges associated with these documents. These contributions offer novel solutions to challenges in Document Understanding, augmenting the practical utility of document understanding models.

LANGUAGE MODELING

Contents

| | | |
|-------|---|----|
| 2.1 | Language Modeling | 7 |
| 2.1.1 | Tasks | 8 |
| 2.1.2 | Language Modeling | 12 |
| 2.1.3 | Evaluation of Language Models | 16 |
| 2.2 | Neural Language Models | 20 |
| 2.2.1 | Distributed Representations of Text | 20 |
| 2.2.2 | Word Embedding Models | 24 |

Language, a complex and intricate system guided, stands out as a distinctive human ability that develops in early childhood and undergoes continuous evolution throughout a lifetime (Hauser et al. 2002). Comprehending and communicating in human language poses a significant challenge for Artificial Intelligence (AI) algorithms. Achieving this goal has been a longstanding research challenge, originating from the proposal of the Turing test in the 1950s (Turing 1950). The aim is to empower machines with the capability to read, write, and communicate in a manner akin to humans. In particular, *language modeling* has emerged as a major approach, extensively studied for language understanding and generation over the past two decades. Since their emergence, language models have consistently pushed the boundaries of state-of-the-art performance across various Natural Language Processing (NLP) benchmarks. In this chapter, we delve into the extensive literature that underpins the evolution of language models, tracing their trajectory from the early era of statistical language models to the advent of neural language models.

2.1 Language Modeling

Language modeling stands out as the major approach to advancing language understanding and generation. A language model is a probabilistic model designed to capture the probability distribution of words within a given language, thereby constructing effective representations of text. Originally conceived for

text generation, language models have recently emerged as a powerful means to establish parametric models that can be fine-tuned on a wide range of tasks. In this section, we explore the diverse tasks in NLP, discuss the building blocks and historical approaches for Language Modeling, and describe how language models are evaluated.

2.1.1 Tasks

The primary goal of language models is to understand and generate human-like text. Playing a pivotal role in numerous NLP tasks encompassing both text understanding and generation, language models are crucial in advancing our understanding of language. We explore several NLP tasks that have received extensive attention due to their practical significance and their role in advancing the understanding on language. Addressing these challenges requires models that can comprehend semantics, context, and syntactic structures in text, making them central to the development of sophisticated NLP systems.

2.1.1.1 Natural Language Understanding

Comprising a broad array of tasks, *Natural Language Understanding* focuses on the ability of machines to process written language.

Text Classification involves categorizing text into one or more pre-defined classes or categories. This task finds applications in various scenarios, including sentiment analysis, spam detection, and content moderation. Automating these processes through language models can streamline data management, decrease manual workload, and enhance the accuracy and efficiency of analysis.

Information Extraction consists in automatically extracting structured information from unstructured and/or semi-structured documents, primarily texts. The goal of information extraction is to convert large volumes of textual data into a more organized and usable format, enabling machines to understand the content. Information extraction involves identifying specific pieces of information, such as entities (*e.g.*, person names, organizations, and quantities), and relationships between entities within a given text. Information extraction consists in extracting semantic *entities* (*entity recognition*) and their relationships (*relation extraction*). Entity recognition is often framed as a Named Entity Recognition (NER) task. NER is a sequence-labeling task which consists in identifying and classifying named entities using a tagging scheme (*e.g.*, BIO format), which provides a structured way to label tokens based on their role in an entity.

Natural Language Inference (NLI) is the task of determining the relationship between two given texts such as *entailment* (the target text implies the source), *contradiction* (the source is false), and *neutral* (there is no relation between the source and the target texts).

Part-of-Speech (POS) tagging is the process of determining the grammatical category, such as noun, verb, adjective, *etc.*, of a given word or text segment, relying on its use and context. This information helps in identifying the syntactic roles of words and understanding the grammatical relationships between them. POS tagging stand as a component of traditional NLP systems, serving as a subtask for practical applications.

Coreference Resolution is the task of finding all linguistic expressions (or *mentions*) that refer to the same entity in a text. It constitutes a crucial stage for many higher level NLP tasks that require a deep understanding of natural language, *e.g.*, information extraction. Similar to POS tagging, coreference resolution constitutes a component of traditional NLP systems.

2.1.1.2 Natural Language Generation

Natural Language Generation focuses on the automatic generation of human-like language. The primary goal of natural language generation is to enable machines to produce coherent and contextually appropriate text.

Text Generation refers to the process of automatically creating human-like text for diverse purposes, such as articles, blogs, research papers, social media posts, source codes, and more.

Text Summarization is a generation task that aims to generate concise and coherent summaries from lengthy texts. Summarization can be categorized into two categories: *extractive* summarization and *abstractive summarization*. Extractive summarization consists in selecting and combining existing sentences from the text to create the summary. On the other hand, abstractive summarization goes beyond verbatim copying and may generate new phrases and sentences that are not present in the source text. Abstractive summarization, with its ability to generate more concise and coherent summaries, has the potential to capture the overall meaning of a text. Overall, text summarization is a crucial component in the development of applications that require efficient information processing, allowing users to access relevant information more quickly and effectively. It plays

a significant role in reducing information overload and improving the accessibility of large volumes of text.

Machine Translation is the automated process of translating text from one language to another. The aim of machine translation is to produce translations that are linguistically accurate and convey the intended meaning of the source text in the target language. Machine translation finds application in a range of domains and industries, including language service providers, global businesses, content localization and information access.

Question Answering involves providing accurate and relevant answers to questions posed in natural language. It encompasses various types of questions, and the answers can be generated from a variety of sources, including knowledge bases, databases, documents, or a combination of sources. Question answering tasks can be open-domain or closed-domain, fact-based or reasoning-based. The emphasis is on formulating an appropriate response to a question. *Extractive* and *abstractive* question answering are two different approaches to formulating answers for questions. Extractive question answering involves selecting and extracting a span of text from a document as the answer to a question. While maintaining factual accuracy, this approach is limited to information explicitly present in the document. To produce concise answers and allow for potential novel insights, abstractive question answering consists in generating a response that may not be explicitly stated in the document. As such, it potentially involves rephrasing or synthesizing information to provide a concise and coherent response. This approach is more challenging as it requires ensuring the generated answers are accurate and contextually appropriate. It has found wide application in scenarios such as search engines and customer support.

Machine Reading Comprehension is a specific type of question answering task that focuses specifically on questions related to a given passage of text, and requires comprehending and extracting information from that passage. The questions are typically formulated based on the content of the provided text, and the goal is to understand the text and extract relevant information to answer the questions. Usually, the passage is provided and the goal is to extract the answer directly from it.

Dialog Systems are designed to engage in natural language conversations with users. They play a crucial role in human-machine interaction, facilitating effective communication between humans and machines. Dialog systems are required to comprehend and interpret user input, keep track of the conversation context, create responses that are appropriate and linguistically coherent, and maintain an

understanding of the state of the conversation and user preferences throughout the interaction. Their applications span various domains such as customer service, education, and entertainment.

2.1.1.3 Towards Evaluating Universal Models

Properly evaluating models designed to handle various [NLP](#) tasks poses a significant difficulty (Jones 2005). As the demand for models capable of addressing diverse linguistic challenges continues to grow, it becomes necessary to establish robust evaluation methodologies. We explore the evaluation process of these universal models, exploring [NLP](#) benchmarks and their role in providing comprehensive insights into model performance across a spectrum of tasks.

Benchmarking emerged as a prominent methodology in the 1980-1990s to address the challenge of conducting proper evaluation. The Stanford Question Answering Dataset ([SQuAD](#)) (Rajpurkar et al. 2016) dataset, a collection of question-answer pairs derived from Wikipedia articles, serves as a widely used evaluation benchmark for question answering models. With the rise of more general-purpose methods in [NLP](#), often replacing task-specific methods, the emergence of new and exhaustive benchmarks followed suit. The General Language Understanding Evaluation ([GLUE](#)) benchmark (A. Wang et al. 2018) was developed to train and assess the performance of natural language understanding models across a diverse set of language tasks. [GLUE](#) covers nine sentence/sentence-pair language understanding tasks (*e.g.*, grammaticality judgments, sentence similarity, [NLI](#)) selected to cover a broad array of dataset sizes, text genres, degrees of difficulty, and various linguistic aspects. The goal of [GLUE](#) is to encourage the development of models that can generalize well, exhibit a broad understanding of natural language, and demonstrate robust performance across different tasks. To advance research aimed at developing models with task-agnostic knowledge representations, the [KILT](#) benchmark (Petroni et al. 2021) was introduced. Comprising eleven datasets across five different tasks—including fact checking, open-domain question answering, slot filling, entity linking, and dialogue—the [KILT](#) benchmark is designed to evaluate models that must condition on specific information within large textual resources. These benchmarks offer both a training set and an evaluation set for each task, enabling researchers to train models on one subset of the data and evaluate their performance on another, ensuring fair assessments of generalization. Additionally, unlike earlier benchmarks, they assign each model a vector of scores to gauge accuracy across a range of scenarios.

2.1.2 Language Modeling

Language Models have incited substantial interest across both academic and industrial domains, owing to their unprecedented performance in various tasks and domains, including medical language processing (Thirunavukarasu et al. 2023), scientific research (H. Wang et al. 2023), and code generation (F. F. Xu et al. 2022).

Language modeling aims to predict the next element in a given sequence of tokens. In the early days of NLP, researchers developed rule-based systems to process language (Manning and Schütze 1999). These systems relied on hand-crafted linguistic rules to analyze and generate text. While these approaches were valuable for specific tasks, they lacked the ability to capture the richness and variability of natural language. This limitation paved the way for the emergence of *Statistical Language Models*.

We begin by discussing text representation units and the methods employed to obtain them. We then explain how probabilities over text sequences are calculated, before delving into the early iterations of language models, *i.e.*, Statistical Language Models.

2.1.2.1 Text Representation Units

Natural language inputs can be commonly modeled as sequences of tokens, with various levels of granularities—characters, words, sentences, and more. To process these sequences, text is usually tokenized. Tokenization is a crucial pre-processing step that consists in splitting the input text into smaller units, *i.e.*, tokens. Tokens serve as the fundamental components of language modeling, and all models operate on raw text at the token level. These tokens are used to build the vocabulary, which represents a set of unique tokens within a corpus. A token can be a character, a word, or a subword. Various algorithms adopt distinct processes to perform tokenization.

Word-based Tokenization divides a text into words using a delimiter, with space and punctuations being the most commonly employed in English. Rules are added into the tokenization process to deal with special cases such as negative forms (for instance, space and punctuation-based tokenization generates three tokens for the word "don't": "don", "'", and "t", whereas a more effective tokenization using specific rules would break it into "do", and "n't").

In English, words like "helps", "helped", and "helping" are derived forms of the base word "help". Similarly, the relationship between "dog" and "dogs" is analogous to that between "cat" and "cats", and "boy" and "boyfriend" show the

same relationship as "girl" and "girlfriend". In some other languages like French and Spanish, verbs can have more than 40 inflected forms. However, word-based tokenization does not cater for the internal structure of words, as morphological information, i.e., word formation and relationships, are not taken into account by the tokenization process. Instead, different inflected forms of the same word (e.g., "cat" and "cats") are tokenized into two distinct tokens. Consequently, models would have more difficulty recognizing the similarity between those words. In addition, word-based tokenization leads to a very large vocabulary. Furthermore, at inference, words not included in the vocabulary must be handled. This typically involves the use of an Out Of Vocabulary (OOV) token, a practice that often contributes to suboptimal results.

Character-based Tokenization (Wehrmann et al. 2017) can be used to alleviate the vocabulary problem. This tokenization process splits the raw text into individual characters, resulting in a very small vocabulary with little to no OOV words.

However, few languages convey a significant amount of information within each character. Therefore, character-based tokenization suffers from a weak correlation between characters and semantic/syntactic aspects of the language. Furthermore, working at the character level results in much longer sequences, which are more challenging to deal with.

Subword-based Tokenization Modern NLP models address both word and character-based tokenization issues by tokenizing a text into *subword* units, a solution between word and character-based tokenization. Subword-based tokenization algorithms aim to break down texts in a way that tackles both the limitations of word-based tokenization—by maintaining a consistent vocabulary size—and the drawbacks of character-based tokenization—by minimizing the number of tokens required to represent a given set of texts. In practice, they use the following principles: 1) frequently used words should not be split into smaller subwords, and 2) rare words should be split into smaller, meaningful words.

Gage (1994) proposed the Byte Pair Encoding (BPE) method, a compression algorithm that breaks down words into subwords to form a compact, fixed-size vocabulary with subwords of varying lengths. The BPE algorithm performs a statistical analysis of the training dataset to identify common symbols within words, e.g., consecutive characters of arbitrary lengths. It starts with an initial vocabulary consisting of symbols of length 1 (characters), and iteratively merges the most frequent pairs of adjacent symbols to produce new, longer symbols. The process stops until a specified number of iterations or a predefined vocabulary size is reached. The resulting symbols can be used as subwords to segment words.

BPE is widely used for input representations in NLP models, and has contributed significantly to improving their performance by enhancing their ability to handle morphologically-rich languages and OOV words.

WordPiece (Y. Wu et al. 2016) is another subword segmentation algorithm. Similar to BPE, WordPiece initializes the vocabulary to include every character found in the training data and iteratively learns a given number of merge rules. Using this vocabulary, a language model is built on the training data. In contrast to BPE, WordPiece does not choose the most frequent symbol pair, but the one that yields the highest increase in the likelihood of the training data once added to the vocabulary. From the updated vocabulary, a new language model is built and the process is repeated until a predefined limit of word units is reached or the likelihood increase falls below a certain threshold.

Subword-based tokenization often maintains linguistic meaning, such as morphemes. For example, BPE decomposes the word "understandable" into meaningful subword units such as "under", "stand", and "able". Consequently, even though a word may be unknown to the model, individual subword tokens may retain enough information for the model to deduce its meaning to a certain degree. Additionally, using subword units helps keeping the vocabulary at a reasonable size.

2.1.2.2 Language Model Definition

A language model is a probabilistic model of a natural language that predicts probability distributions over sequences of tokens. Given a sequence of tokens w_1, w_2, \dots, w_n , a language model aims to calculate the joint probability $P(w_1, w_2, \dots, w_n)$ of the whole sequence. Using the chain rule, the probability of the sequence can be decomposed into a product of conditional distribution on tokens. Most commonly, the probability P of a sequence of words can be obtained from the probability of each word given the preceding ones:

$$P(w_1, \dots, w_n) = \prod_{t=1}^n P(w_t | w_1, \dots, w_{t-1}). \quad (2.1)$$

Many tasks, if not all, can be formulated as a sequence-to-sequence task, wherein w_1, \dots, w_k represents the input (e.g., a text to summarize). The probability of the output w_{k+1}, \dots, w_n (e.g., the corresponding summary) conditioned on w_1, \dots, w_k can then be calculated as follows:

$$P(w_{k+1}, \dots, w_n | w_1, \dots, w_k) = \prod_{t=k+1}^n P(w_t | w_1, \dots, w_{t-1}). \quad (2.2)$$

In other words, the probability of a sequence is estimated as a product of each token's probability given its preceding tokens. *Causal*¹, or *autoregressive* language models use this decomposition.

A successful language model estimates the distribution across text sequences, encoding not only the grammatical structure, but also the potential knowledge embedded in the training corpora (Jozefowicz et al. 2016).

2.1.2.3 Statistical Language Models

The history of language models can be traced back to the 1990s, a period that marked the emergence of Statistical Language Models. Such language models are rooted in probabilistic approaches to predict word sequences. The underlying idea is to simplify the word prediction model using the Markov assumption, *e.g.*, approximating the probability of the next word using the most recent context. Prominent examples include models based on n -grams (P. F. Brown et al. 1992; Omar and Al-Tashi 2018). It is worth noting that these models employ word-based tokenization algorithms, which could have influenced their performance.

N -gram Models simplify the calculation of the joint probability by operating on the assumption that the likelihood of the next token in a sequence solely depends on a fixed-size window spanning the $n - 1$ previous tokens (*n-gram*). If only one prior token is considered, it is termed a bigram model; with two tokens, a trigram model; and with $n - 1$ words, an n -gram model. Given a window size n , the calculation of the joint probability is simplified as follows:

$$P(w_1, \dots, w_T) \approx \prod_{t=1}^T P(w_t | w_{t-n}, \dots, w_{n-1}). \quad (2.3)$$

N -grams models often approximate $P(w_1, \dots, w_T)$ using frequency counts based on n -grams.

On the Curse of Dimensionality Statistical Language Models represent tokens through one-hot encoding, where each token is represented as a sparse binary vector, with a dimension for each unique token in the vocabulary. In this encoding, all dimensions are zero except for the one corresponding to the token, which is set to one. Hence, one-hot encoding leads to very high-dimensional and sparse representations. This often hinders the accurate estimation of language models, as one-hot encoding requires estimating an exponential number of transition proba-

1. This name is common in the literature but is misleading as it has little connection to the proper study of causality.

bilities. Furthermore, one-hot encoding introduces greater difficulty in capturing semantic relationships between tokens (each individual token is treated independently of the others) and handling OOV tokens efficiently. This phenomenon is referred to as the *curse of dimensionality*. To tackle this issue, specific smoothing strategies, including backoff estimation (Katz 1987) and Good-Turing estimation (Gale and Sampson 1995), have been introduced to alleviate the problem of data sparsity.

Statistical Language Models have found extensive application in boosting performance across NLP tasks such as speech recognition (Bahl et al. 1989) and POS tagging (Theide and Harper 1999). Although capable of basic text generation and word prediction, their limitations become apparent when attempting to capture complex contextual relationships (Rosenfeld 2000; Arisoy et al. 2012).

2.1.3 Evaluation of Language Models

As language models play an increasingly critical role in both research and daily applications, the importance of their evaluation grows significantly. The evaluation of language models stands as a crucial phase in assessing their efficacy and performance, bridging the gap between theoretical advancements and practical utility. We explore *automatic evaluation*, where several key metrics can be employed to provide valuable insights into the capacities and limitations of a language model. Language models can be evaluated using *intrinsic* or *extrinsic* evaluation.

2.1.3.1 Intrinsic Evaluation

An intrinsic evaluation metric measures the quality of the language model independently of any application, and can be used to quickly assess potential improvements in the model. There are three commonly employed metrics associated with the likelihood of the data given a model.

Perplexity is a widely used intrinsic metric that measures how uncertain a language model is about the predictions it makes. Given an input sequence $\mathbf{w} = (w_1, \dots, w_n)$, and $P(w_1, \dots, w_n)$ the probability assigned to \mathbf{w} by a model P , the perplexity of P on \mathbf{w} can be defined as follows:

$$\text{PPL}(\mathbf{w}) = P(w_1, \dots, w_n)^{-\frac{1}{n}} \quad (2.4)$$

The lower the perplexity of a language model, the more confident (but not necessarily accurate) it is. Perplexity often correlates well with the model’s performance on the target tasks, and it can be easily computed from the probability distribution learned during training. Hence, perplexity is a reliable metric to filter out models that are unlikely to perform well in real-world scenarios. However, comparing perplexity across different datasets, context lengths, vocabulary sizes, and tokenization procedures is challenging.

Cross-entropy (also referred to as *negative log-likelihood*) serves as an intrinsic metric for measuring the ability of a model to compactly represent a sequence of tokens. The cross-entropy of P on a sequence of tokens \mathbf{w} is defined as:

$$\text{CE}(\mathbf{w}) = -\frac{1}{n} \log P(w_1, \dots, w_n) = \log(\text{PPL}(\mathbf{w})). \quad (2.5)$$

The lower the cross-entropy value, the more efficient the model is at encoding the sequence.

Bits-per-character (BPC) calculates the average number of bits needed to represent each character in a text using the model’s encoding scheme. It is equivalent to character-level cross-entropy, computed with a binary logarithm. Given an input sequence of characters $\mathbf{w} = (w_1, \dots, w_n)$, **BPC** is defined as:

$$\text{BPC}(\mathbf{w}) = -\frac{1}{n} \sum_{i=1}^n \log_2 P(w_i). \quad (2.6)$$

Both cross-entropy and **BPC** are often used to assess the compression capabilities of language models. Notably, **BPC** serves as a metric for evaluating models in the Hutter Prize contest and its associated enwiki8 benchmark on data compression.²

While no single metric may capture all dimensions of model behavior, combining them offers a more nuanced understanding of a model’s strengths and weaknesses, enabling robust evaluation and facilitating comparison across different aspects of performance.

2.1.3.2 Extrinsic Evaluation

Good intrinsic evaluation scores do not always translate into better performance for downstream tasks. Therefore, extrinsic evaluation, also called task-based eval-

2. <http://prize.hutter1.net/>

uation, is used to gauge the practical utility of a language model for specific applications. In the following, we focus on NLP tasks.

Sequence Labeling Precision, recall, and F1 score are essential metrics for sequence labeling tasks, such as text classification and information extraction. Precision quantifies the accuracy of positive predictions by calculating the ratio of true positive predictions to the total number of positive predictions. Recall measures the model’s ability to capture all relevant instances of a positive class by computing the ratio of true positive predictions to the total number of actual positive instances. A high precision indicates that the model’s positive predictions are mostly correct, minimizing false positives, while a high recall indicates that the model is effective at identifying most of the actual positive instances, minimizing false negatives. F1 score provides a balanced view by considering both precision and recall. F1 score is the harmonic mean of precision and recall, with higher values indicating a better balance between precision and recall. It is particularly valuable when both false positives and false negatives need to be minimized. However, it is not suitable for highly imbalanced data.

Text Generation Furthermore, when dealing with text generation and summarization tasks, metrics such as Recall-Oriented Understudy for Gisting Evaluation (ROUGE) (C.-Y. Lin 2004) and Bilingual Evaluation Understudy (BLEU) (Papineni et al. 2002) are used to measure the similarity between the generated text and one or more reference texts. Both metrics range between 0 and 1, with 1 indicating perfect overlap with the reference.

ROUGE is a family of metrics commonly used for summarization tasks. It evaluates the similarity between the text generated by the model and the human-produced reference summary by calculating precision, recall, and F1 score based on the overlap in n -grams. Formally:

$$\begin{aligned} \text{ROUGE-}n_{\text{precision}} &= \frac{|\text{predicted } n\text{-grams} \cap \text{reference } n\text{-grams}|}{|\text{predicted } n\text{-grams}|} \\ \text{ROUGE-}n_{\text{recall}} &= \frac{|\text{predicted } n\text{-grams} \cap \text{reference } n\text{-grams}|}{|\text{reference } n\text{-grams}|} \\ \text{ROUGE-}n_{\text{F1}} &= 2 * \frac{\text{recall} * \text{precision}}{\text{recall} + \text{precision}}. \end{aligned} \quad (2.7)$$

ROUGE features multiple variants, each corresponding to specific n -gram overlaps — ROUGE-1 for unigrams, ROUGE-2 for bigrams, and so forth. Additionally, the widely-used ROUGE-L variant takes into account sentence-level structure similarity by employing the Longest Common Subsequence (LCS), which represents the

longest sequence of common, not necessarily consecutive, ordered words between two sequences. **ROUGE-L** is defined as follows:

$$\begin{aligned} \text{ROUGE-L}_{\text{precision}} &= \frac{\text{LCS}(\text{prediction}, \text{reference})}{\# \text{ words in prediction}} \\ \text{ROUGE-L}_{\text{recall}} &= \frac{\text{LCS}(\text{prediction}, \text{reference})}{\# \text{ words in reference}} \\ \text{ROUGE-L}_{\text{F1}} &= 2 * \frac{\text{recall} * \text{precision}}{\text{recall} + \text{precision}}. \end{aligned} \quad (2.8)$$

ROUGE usually denotes the F1 score from the n -gram precision and recall. Higher **ROUGE** scores indicate better content overlap.

BLEU is another widely used metric for evaluating the quality of machine-generated text, particularly in translation tasks. Similarly to **ROUGE**, **BLEU** evaluates the precision of n -grams in the generated translation by comparing them to the reference translations. **BLEU** considers precision across different n -gram levels (unigrams, bigrams, trigrams, *etc.*). The precision is then adjusted with a brevity penalty to account for translations that are shorter than the reference translations, ensuring the generation of sequences of appropriate length. Using the geometric mean, the cumulative **BLEU** score combines precision scores for all specified n -gram levels. **BLEU** can handle multiple reference texts, offering a more robust evaluation that accounts for variations in human-produced references.

Both **ROUGE** and **BLEU** primarily focus on n -gram overlap and may not fully capture the fluency, coherence, or semantic quality of generated text. As a result, human evaluation remains crucial for a comprehensive evaluation of text generation quality.

Human Evaluation consists in having human annotators evaluate the quality of generated text on specific tasks. Annotators can rate the generated text based on its fluency, coherence, and relevance to the given output. Human evaluation considers factors that might be difficult to quantify, *e.g.*, the overall quality of the generated text, creativity, or the ability to handle ambiguous or nuanced language. While it can be time-consuming and subjective, human evaluation offers valuable insights into how language models perform in real-world scenarios. Integrating human judgment helps uncovering potential limitations, biases, or domains where models might struggle.

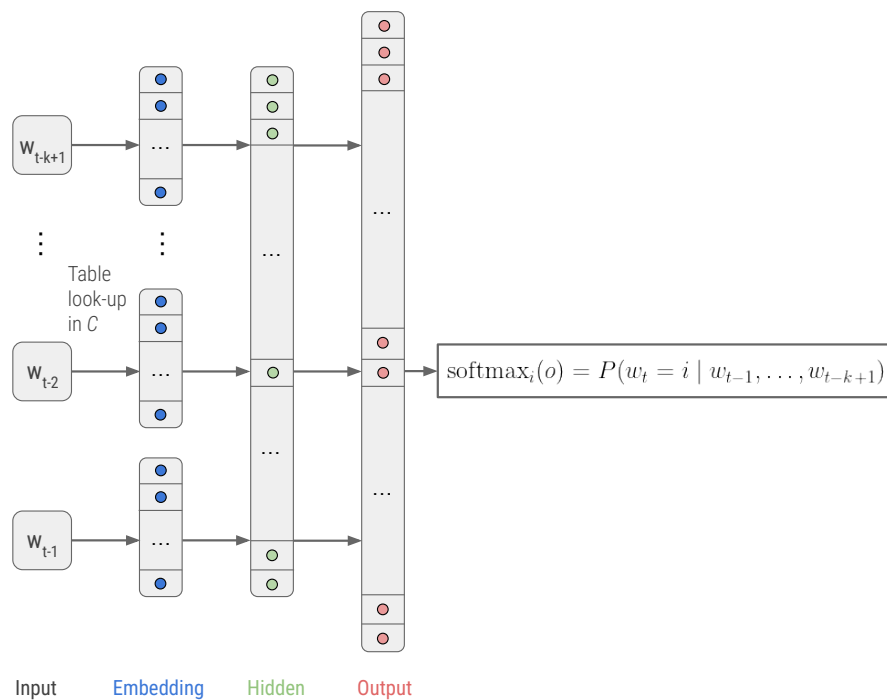


Figure 2.1 – Architecture of Bengio et al. (2000)'s Neural Language Model.

2.2 Neural Language Models

Starting in the 2000s, neural networks began to be used for language modeling (Bengio et al. 2000), and representation of text shifted from being non-continuous to being continuous (*distributed*). The mid-2010s marked a significant milestone in language modeling with the emergence of Deep Learning, laying foundation for the development of *Neural Language Models*. A Neural Language Model is a language model that exploits the ability of neural networks to learn distributed representations of text. Neural Language Models delve into vast amounts of data to learn the intricate patterns and structures of language, allowing them to significantly improve their ability to understand context. In this section, we first describe how distributed representations of textual data can be obtained from neural architectures, before exploring notable word embedding models.

2.2.1 Distributed Representations of Text

A fundamental challenge that renders language modeling challenging is the curse of dimensionality, primarily stemming from the sparse and high-dimensional nature of discrete tokens within a large vocabulary. Text representation has there-

fore evolved from a non-continuous to a continuous (*i.e.*, *distributed*) form, serving as the building blocks for contemporary NLP. Distributed representations are dense, low-dimensional, continuous-valued representations that rely on the distributional hypothesis, which posits that tokens with similar contexts have similar (or related) meaning (Mikolov et al. 2013). Using distributed representations requires a much smaller number of features than the size of the vocabulary. Additionally, the geometry of the space in which the vocabulary is embedded induces the similarity structure between words, *i.e.*, words with related meanings appear in the same region in the embedding space (Shazeer et al. 2016).

Initial attempts to estimate distributed word representations, as seen in methods like Probabilistic Latent Semantic Analysis (PLSA) (Hofmann 2001) and Latent Dirichlet Allocation (LDA) (Blei et al. 2003), centered around extracting token representations from co-occurrence matrices to represent words and documents in a latent topical space. Such continuous representations, while effective for their intended purposes such as document clustering and term analysis, had a limited success when applied to a broader range of NLP tasks. These limitations arise from a couple of key factors. Firstly, these approaches overlook the context and relationships between words, essential for understanding natural language in diverse applications. Second, these methods were designed for discovering latent topics within a corpus without incorporating explicit task-specific supervision. However, no powerful supervised models leveraging such a representation were created at this time. The prevailing approach involved using a classifier atop a continuous representation, which was generated without supervision, typically through a language modeling task. Consequently, the representation lacked the requisite task-specificity to be successfully applied across various NLP tasks.

Due to these limitations, researchers and practitioners have shifted towards using representations generated by *Neural Language Models*, as pioneered by Bengio et al. (2000). Using neural networks, Neural Language Models learn the probability distribution of a word sequence given the previous context, while embedding the vocabulary in a continuous space to obtain distributed representations. Given a context of size k , and a training sequence of words $(w_1, \dots, w_n) \in V^n$, where V is the vocabulary, the objective is to estimate $P(w_t \mid w_1, \dots, w_{t-1})$ through a neural network $F(w_t, \dots, w_{t-k+1})$. Therefore, generalization can be obtained more easily: a sequence of words that has never been encountered before is assigned a high probability if it consists of words with representations similar to those in a sentence used during training. The distributed word vectors are first built using a matrix $C \in \mathbb{R}^{|V| \times h}$ whose parameters are the representations of all the words of the vocabulary: the i -th row in C is the distributed representation $C_i \in \mathbb{R}^h$ for word i . To obtain the next word w_t , the probability function over words is expressed as a function G that maps the input sequence of feature vectors in the context, $(C_{w_{t-k+1}}, \dots, C_{w_{t-1}})$, to a conditional probability distribution over words

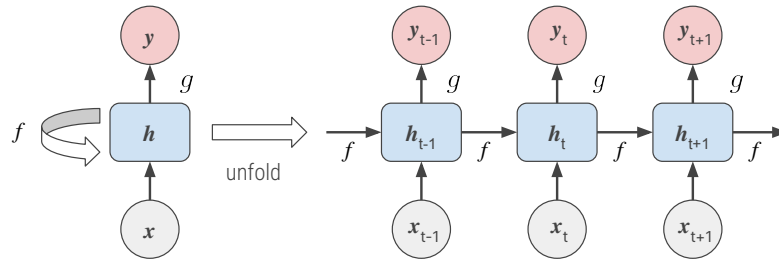


Figure 2.2 – Compressed (left) and unfolded (right) views of a Recurrent Neural Network.

in V . The output of G is a vector whose i -th element provides an estimate of the probability $P(w_t = i \mid w_1, \dots, w_{t-1})$. Depicted in Figure 2.1, the function F is a composition of C and G , expressed as follows:

$$F_\omega(w_{t-1}, \dots, w_{t-k+1}) = G_\omega(C(w_{t-1}), \dots, C(w_{t-k+1})). \quad (2.9)$$

The function G can be implemented as a feed-forward network—the simplest type of neural network where information is only processed in one direction—or any another parameterized function, and is trained by minimizing cross-entropy. More specifically, Neural Language Models leverage the capabilities of the prevailing architectures of the 2010s, Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs).

2.2.1.1 Recurrent Neural Networks

RNNs are neural networks designed for sequential data, renowned for their capacity to maintain "memory" across time steps. This enables them to process each token in a sequence with information from prior tokens, allowing them to capture sequential dependencies and achieve superior performance compared to feed-forward neural networks.

A RNN is formally expressed as a function parameterized by a set of parameters θ shared across all time steps. At each time step t , the model takes input x_t and a fixed-size hidden state vector \mathbf{h}_{t-1} from the previous time step. The hidden state at time t , \mathbf{h}_t , acts as a "memory", summarizing information from previous words (w_1, \dots, w_{t-1}). Using two activations functions f_θ and g_θ (usually sigmoid, hyperbolic tangent, and Rectified Linear Unit (ReLU) functions), the hidden state \mathbf{h}_t and the distribution of probabilities over the output $\hat{\mathbf{y}}_t$ at time t are computed as follows:

$$\mathbf{h}_t = f_\theta(\mathbf{x}_t, \mathbf{h}_{t-1}) \quad (2.10)$$

$$\hat{\mathbf{y}}_t = g_{\theta}(\mathbf{h}_t) \quad (2.11)$$

Figure 2.2 illustrates the architecture of an RNN.

RNNs offer theoretically unlimited context, with the hidden state ideally encapsulating information about all preceding tokens in the sequence. Outperforming n -gram models, RNN-based language models generate more natural-sounding text across various scenarios (Mikolov et al. 2010; Kovačević and Kečo 2022). The final state \mathbf{h}_n post-training represents the entire text (w_1, \dots, w_n) and proves valuable in diverse tasks like speech recognition (Schwenk 2007) and morpho-syntactical labeling (Collobert 2011).

In practice, handling long sequences introduces complications, where the repetition of recurrent connections leads to the vanishing and exploding gradient problems (Hochreiter et al. 2001). The vanishing gradient problem results in slow learning or information loss over time, while the exploding gradient problem causes unstable training. To overcome these challenges, the Long Short Term Memory (LSTM) model (Hochreiter and Schmidhuber 1997) introduces gated structures, allowing selective retention of information over time. With more complex cells featuring gates driven by sigmoid activation functions, LSTMs alleviate the vanishing gradient problem, enhancing the model’s ability to manage long sequences.

2.2.1.2 Convolutional Neural Networks

CNNs constitute a family of neural network models characterized by a specific layer known as the convolutional layer. In this layer, features are extracted by convolving a learnable filter (or kernel) across various positions of a vectorial input. CNNs, initially designed to deal with images (LeCun et al. 1989), are built upon two fundamental concepts: (1) the processing of an image region should not depend on its specific location (two-dimensional equivariance of data), (2) patterns should be captured at various levels of abstraction, progressing from regions composed of basic shapes to larger ones representing real-world objects. These concepts can also be applied to texts, where the translation equivariance is unidimensional rather than bi-dimensional.

In NLP, CNNs are commonly used for static classification tasks, including sentiment analysis (Kalchbrenner et al. 2014), relation extraction (T. H. Nguyen and Grishman 2015), and entity recognition (Adel et al. 2016). CNNs can integrate information from large context windows, learn high-level abstraction patterns, and exhibit greater stability compared to RNNs in handling the vanishing gradient problem. However, CNNs face more limitations when applied to text in contrast to images. For tasks like language modeling, deeper models tend to harm per-

formance (N.-Q. Pham et al. 2016). This is attributed to the distinct nature of visual and linguistic data, with convolution in language processing potentially distorting the input to the point where it is no longer recognizable as text.

2.2.2 Word Embedding Models

Word embeddings have been shown to significantly improve and simplify many NLP applications (Collobert et al. 2011). However, the approach introduced by Bengio et al. (2000) requires calculating a probability distribution for all words in the vocabulary (Equation 2.9). As a result, their embeddings are slow to compute and cannot be effectively learned from large datasets.

The work of Bengio et al. (2000) laid the foundation for the development of more computationally efficient methods. Subsequent research in NLP primarily focused on unsupervised learning of token representations from large corpora, with the intent of leveraging them across diverse NLP tasks.

2.2.2.1 Word2Vec

Mikolov et al. (2013) introduced Word2Vec, a shallow neural network designed to efficiently learn continuous word embeddings by grouping semantically similar words in the same region of the vector space. While the model may not represent data as precisely as a neural network with limited data, its efficiency improves significantly when trained on larger datasets, enabling more accurate data representation.

Skip-gram is the simplest and most widely used model proposed by Mikolov et al. (2013). The idea behind Skip-gram is to learn word representations in a manner that allows the context to be inferred from these representations. Therefore, words that co-occur in similar contexts have similar representations. Given a word, the Skip-gram model tries to predict the words that are likely to appear around it. The training objective of the model consists in maximizing the following log-probability:

$$\sum_{(t,c)} \log P(t \text{ appears in the context } c) = \sum_{(t,c)} \log P(t | c), \quad (2.12)$$

where (t, c) corresponds to the set of terms t associated with the context c . The context is defined by a fixed-sized window centered on t , such that any word in the window but t are part of the context. Given \underline{c} the embedding for the context c and \underline{t} the embedding for the target word t , the probability of t to appear in c is expressed as follows:

$$P(t | c) = \frac{\exp(\underline{t} \cdot \underline{c})}{\sum_{t' \in V} \exp(\underline{t}' \cdot \underline{c})} \quad (2.13)$$

To maximize the similarity of words appearing in the same context and minimizing it when they occur in different contexts, it is necessary to sample pairs of words that should not appear in the same context, *i.e.*, negative samples.

2.2.2.2 FastText

One major limitation of Word2Vec lies in its inability to effectively handle OOV words. Additionally, Word2Vec lacks shared representations at the subword level, which can become a challenge when dealing with morphologically rich languages such as Arabic or German. FastText (Bojanowski et al. 2017) mitigates these issues by introducing subword information into the model, allowing for better handling of morphologically rich languages and enhanced understanding of subword structures. While Word2Vec treats each word as an atomic unit, fastText represents words as bags of character n -grams. Rather than learning word embeddings, the model generates subword representations, and words are represented by the sum of their subword vectors. Formally, given \mathcal{G}_t the set of all subwords of the word t , the target word embedding \underline{t} in the Skip-gram model can be defined as:

$$\underline{t} = \sum_{g \in \mathcal{G}_t} z_g, \quad (2.14)$$

where z_g is the vector of subword g in the dictionary. The rest of the process is identical to the Skip-gram model.

The significant advantage of fastText over Word2Vec lies in its ability to capture the structure of words, including uncommon or unseen ones during training, by sharing parameters among words with similar structures. This capability is particularly valuable for languages with complex word forms.

2.2.2.3 Conclusion

The success of the Word2Vec algorithm (Mikolov et al. 2013) has sparked immense interest in word embeddings within NLP researchers and practitioners, leading to the development of a myriad of alternative models (Pennington et al. 2014; Shazeer et al. 2016; Bojanowski et al. 2017). However, a significant drawback of such word embedding algorithms is that they produce static, context-independent embeddings. In other words, a word embedding remains the same

across linguistic context — for instance, "jaguar" would have the same embedding whether referring to an animal or a car brand. Moreover, these approaches overlook multi-word expressions, grammatical nuances, and word-sense information, which can be crucial for handling polysemy and homonymy.

To overcome these limitations, *contextual* word embedding models have been introduced, initiating the use of language models for representation learning beyond mere word sequence modeling. These methods use language modeling to generate contextual word embeddings that adapt to the word's usage by considering its complete context. These context-aware representations, learned by pre-training the language model on large unlabeled corpora, serve as highly effective general-purpose semantic features, significantly raising the performance bar of NLP tasks. These studies have inspired numerous follow-up work, establishing the "*pre-training then fine-tuning*" paradigm as the prevailing learning approach. The next chapter will delve into an in-depth exploration of this paradigm.

TRANSFORMER ARCHITECTURE & PRE-TRAINED LANGUAGE MODELS

Contents

| | | |
|-------|---|----|
| 3.1 | Transformers | 29 |
| 3.1.1 | Attention Mechanism | 29 |
| 3.1.2 | Attention Layers in Transformers | 33 |
| 3.1.3 | Defining a distribution over the tokens | 36 |
| 3.1.4 | Sequential Information in Transformers | 37 |
| 3.1.5 | Segment Information in Transformers | 39 |
| 3.2 | Transformer-based Pre-trained Language Models | 39 |
| 3.2.1 | Bidirectional Models | 39 |
| 3.2.2 | Generative Models | 43 |
| 3.2.3 | Conclusion | 49 |
| 3.3 | Long-range Modeling | 50 |
| 3.3.1 | Long-range Models | 51 |
| 3.3.2 | Benchmarking Long-range Models | 55 |
| 3.3.3 | Conclusion | 57 |
| 3.4 | Conclusion | 58 |

Natural Language Processing (NLP) models were historically trained *from scratch* in a supervised manner to perform specific tasks, based on (limited) training data. As a result, training deep neural networks on such small datasets led to overfitting, making the models sensitive to even slight shifts in the data distribution.

While word embeddings such as Word2Vec’s (Mikolov et al. 2013) are learned from large corpora, their application in task-specific neural models is restricted to the input layer. Task-specific neural models must be built nearly from scratch, given that the majority of model parameters handling token interactions need to be optimized for the task at hand. This optimization process requires substantial amounts of data to attain a high-performance model.

The seminal works of Peters et al. (2018), Devlin et al. (2018), and Radford et al. (2018) marked a paradigm shift by generalizing the use of large unsupervised training datasets in NLP, initiating a new era of *Pre-trained Language Models*, also

referred to as *Foundation Models*. This shift is driven by the idea that modeling language using large corpora of text is sufficient for training powerful language models capable of performing well across various NLP tasks. Rather than building task-specific models from scratch using scarce labeled data, Pre-trained Language Models leverage extensive unsupervised (or *self-supervised*) pre-training to learn general language representations, and apply the knowledge acquired to achieve broad task applicability in NLP. The *pre-training* phase allows the model to capture context and a general understanding of syntax and semantics. After pre-training, the model can be *fine-tuned* on specific downstream tasks (e.g., text classification, Named Entity Recognition (NER), machine translation, and more). Fine-tuning consists in further training the pre-trained model on a smaller, labeled dataset specific to the downstream task. The knowledge acquired during pre-training is leveraged and tailored to the new task, resulting in improved performance.

As one of the early endeavors to adopt the "pre-training then fine-tuning" paradigm in NLP, Embeddings from Language Models (ELMo) generates contextual representations that can be fine-tuned by adding task-specific layers. However, ELMo's effectiveness is somewhat constrained compared to models like Bidirectional Encoder Representations from Transformers (BERT), in part due to its reliance on Recurrent Neural Networks (RNNs). The breakthrough in Pre-trained Language Models came with the introduction of the Transformer architecture in the work of Vaswani et al. (2017). Transformer-based Pre-trained Language Models (Devlin et al. 2018; Radford et al. 2018) have demonstrated that fine-tuning improves the state of the art across a wide range of language tasks. This suggests that task-specific architectures are no longer a necessity. Furthermore, Transformer-based Pre-trained Language Models perform better with an increased volume of data, model size, and training compute, demonstrating superior scaling behavior (J. Kaplan et al. 2020). Hence, the Transformer has become the go-to component in the modern NLP stack, largely replacing other architectures such as RNNs.

In this chapter, we first describe the Transformer architecture, with a focus on its core component—the attention mechanism. Subsequently, we discuss how Transformers can be leveraged to build effective language models, ranging from bidirectional models capable of producing robust, general-purpose word representations to generative models able to create coherent and contextual relevant text, thereby laying the groundwork for *Large Language Models*. Finally, we focus on efficient self-attention model variants designed to efficiently process long texts.

3.1 Transformers

Transformer (Vaswani et al. 2017) tackles two primary limitations of RNNs. RNNs suffer from the vanishing/exploding gradient problem, which hinders their ability to capture long-range dependencies. In addition, the sequential processing of input in RNNs hampers efficient parallelization (Vaswani et al. 2017). To overcome these limitations, the Transformer removes recurrence altogether by leveraging the *self-attention* mechanism to capture global dependencies.

Text is tokenized into subwords using a subword-based tokenization algorithm (Gage 1994; Y. Wu et al. 2016). To obtain text representations, the Transformer uses an embedding table, which contains vectors for each token in the vocabulary. Token embeddings are derived by associating each token in the sequence with their respective vector in this table.

We elaborate on the concept of *attention* and explore a few attention mechanisms, including *self-attention*. We then study the application of self-attention within the Transformer architecture and discuss the formulation of a conditional distribution over the vocabulary. Finally, we provide details on the processing of sequential and segment information in Transformers.

3.1.1 Attention Mechanism

The concept of attention can be best explained through an analogy with human biological systems. In various problems involving language, speech, or vision, specific parts of the input are more important than others. For instance, in tasks like machine translation and summarization, only certain words in the input sequence may hold relevance for predicting the next word. An attention mechanism integrates this idea of relevance by allowing the model to dynamically *pay attention* to specific portions of the input that contribute to effectively performing the task at hand. In the following, we formalize the concept of "paying attention", specifically in the context of Transformers.

3.1.1.1 Bahdanau's Attention Mechanism

The earliest use of attention was proposed by Bahdanau et al. (2014) for a *sequence-to-sequence* modeling task. A sequence-to-sequence task involves mapping a sequence of n input vectors to a sequence of m target vectors, where m is unknown apriori. A sequence-to-sequence model (Sutskever et al. 2014) consists of an *encoder-decoder* architecture, where the encoder encodes an input sequence

$(\mathbf{x}_1, \dots, \mathbf{x}_n)$ into a sequence of fixed-size state vectors $(\mathbf{h}_1, \dots, \mathbf{h}_n)$. The decoder is then fed the fixed-size vector \mathbf{h}_n and generates an output sequence $(\mathbf{y}_1, \dots, \mathbf{y}_m)$.

In a traditional encoder-decoder architecture (usually based on RNNs), the encoder must compress all the necessary input information for the task at hand into a single fixed-size vector \mathbf{h}_n that is fed to the decoder f . Given a position i , let \mathbf{y}_{i-1} be the previous output token and \mathbf{s}_{i-1} the previous decoder hidden state. The probability of \mathbf{y}_i , conditioned on $\mathbf{y}_1, \dots, \mathbf{y}_{i-1}$ and $\mathbf{x}_1, \dots, \mathbf{x}_n$, is expressed as follows:

$$P(\mathbf{y}_i \mid \mathbf{y}_1, \dots, \mathbf{y}_{i-1}; \mathbf{x}_1, \dots, \mathbf{x}_n) = f(\mathbf{s}_{i-1}, \mathbf{y}_{i-1}, \mathbf{h}_n). \quad (3.1)$$

However, encoding a variable-length input into a fixed-size vector \mathbf{h}_n squashes the information of the input sequence, irrespective of its length, causing the performance to deteriorate rapidly as the input sequence length increases (K. Cho et al. 2014). In addition, in sequence-to-sequence tasks, each output token is expected to be more influenced by specific parts of the input sequence. However, the decoder lacks any mechanism to selectively focus on relevant input tokens.

To alleviate these challenges, Bahdanau et al. (2014) introduce the attention mechanism, a principle that allows the decoder to access the entire encoded input sequence $(\mathbf{h}_1, \dots, \mathbf{h}_n)$ and dynamically *attend to* information deemed relevant to generate the next output token. Formally:

$$P(\mathbf{y}_i \mid \mathbf{y}_1, \dots, \mathbf{y}_{i-1}; \mathbf{x}_1, \dots, \mathbf{x}_n) = f(\mathbf{s}_{i-1}, \mathbf{y}_{i-1}, \mathbf{c}_i), \quad (3.2)$$

where \mathbf{c}_i is the context vector associated with decoding position i , acting as a summary of the encoding states. It is defined as the weighted sum of all encoder hidden states $\{\mathbf{h}_j\}_{j=1, \dots, n}$ and their corresponding attention weights $\{\alpha_{ij}\}_{j=1, \dots, n}$. The context vector \mathbf{c}_i is expressed as follows:

$$\mathbf{c}_i = A(\mathbf{s}_{i-1}, \mathbf{h}) = \sum_{j=1}^n \alpha_{ij} \mathbf{h}_j. \quad (3.3)$$

The attention weights $\{\alpha_{ij}\}_{i=1, \dots, n}$ determine the relevance between each encoder hidden state \mathbf{h}_j and each decoder hidden state \mathbf{s}_{i-1} :

$$\alpha_{ij} = \text{softmax}_j((a(\mathbf{s}_{i-1}, \mathbf{h}_1), \dots, a(\mathbf{s}_{i-1}, \mathbf{h}_n))) = \frac{\exp(a(\mathbf{s}_{i-1}, \mathbf{h}_j))}{\sum_k \exp(a(\mathbf{s}_{i-1}, \mathbf{h}_k))}, \quad (3.4)$$

where a is an alignment function implemented as a feed-forward network. The alignment score $a(s_{i-1}, h_j)$ indicates how well the element h_j of the input sequence aligns with the current output s_{i-1} .

3.1.1.2 Generalized Attention

The attention mechanism can be reformulated into a general form where the alignment does not inherently rely on the hidden RNN states. The *generalized attention* model (Chaudhari et al. 2021) extends the attention mechanism of Bahdanau et al. (2014) by allowing for more flexibility and adaptability in capturing dependencies between different parts of the input and output sequences. While the original attention mechanism focuses on aligning parts of the input sequence with the current position in the output sequence, the generalized attention model introduces parameters and mechanisms to customize and control the attention process. In the generalized attention model, attention weights are not solely determined by the relevance between the current decoder hidden state and the encoder hidden states. Instead, the model introduces learnable parameters and scoring functions that can be adjusted to capture different types of relationships. This allows the attention mechanism to consider various aspects, such as semantic similarity, positional information, or other task-specific factors.

Let $\mathbf{x} = (x_1, \dots, x_n)$ be a source sequence and $\mathbf{y} = (y_1, \dots, y_n)$, a target sequence. A generalized attention model is characterized by three components: the queries \mathbf{Q} , the keys \mathbf{K} , and the values \mathbf{V} , all derived from either \mathbf{x} or \mathbf{y} using learnable weight matrices $\mathbf{W}^{(q)}$, $\mathbf{W}^{(k)}$, and $\mathbf{W}^{(v)}$. Queries \mathbf{Q} indicate which information is requested from the input sequence, keys \mathbf{K} are vectors associated with each element in the input sequence and determine which elements are relevant for the queries, while values \mathbf{V} contain the information to be propagated. Given a set of key-value pairs (\mathbf{K}, \mathbf{V}) and a query $q_i \in \mathbf{Q}$, generalized attention is defined as follows:

$$A(q_i, \mathbf{K}, \mathbf{V}) = \sum_j \alpha_{ij} v_j, \quad (3.5)$$

where:

$$\alpha_{i1}, \dots, \alpha_{in} = p((a(q_i, k_1), \dots, a(q_i, k_n))). \quad (3.6)$$

The alignment function, a , dictates the combination of keys and queries, such as through dot product or cosine similarity. The query vector is compared with each key to compute alignment (or similarity) scores. The distribution function p takes these scores as input and ensures that the attention weights lie between

0 and 1 and are normalized to sum to 1, achieved through mechanisms like logistic sigmoid or softmax function. The final vector $A(\mathbf{q}_i, \mathbf{K}, \mathbf{V})$ is formulated as a weighted sum of all value vectors $\{\mathbf{v}_j\}$ and their corresponding attention weights $\{\alpha_{ij}\}$.

The attention mechanism of Bahdanau et al. (2014) can be seen as a special case of generalized attention where the queries and values are analogous to the encoded outputs, *i.e.*, $\mathbf{K} = \mathbf{V} = \{\mathbf{h}_j\}_{j=1,\dots,n}$, and the query is analogous to the previous decoder output, *i.e.*, $\mathbf{q} = \mathbf{s}_{i-1}$.

3.1.1.3 Attention in Transformers

One common form of the generalized attention model is *self-attention*, or scaled dot-product attention, introduced by Vaswani et al. (2017) in the Transformer architecture. The alignment function a is defined by a scaled dot product, *i.e.*:

$$a(\mathbf{q}_i, \mathbf{k}_j) = \frac{\mathbf{q}_i^\top \mathbf{k}_j}{\sqrt{d_k}}, \quad (3.7)$$

while the distribution function p , used to obtain the final attention weights, corresponds to the softmax:

$$\begin{aligned} \alpha_{ij} &= \text{softmax}_j \left(\frac{\mathbf{q}_i^\top \mathbf{K}}{\sqrt{d_k}} \right) \\ &= \frac{\exp(\mathbf{q}_i^\top \mathbf{k}_j / \sqrt{d_k})}{\sum_{j'} \exp(\mathbf{q}_i^\top \mathbf{k}_{j'} / \sqrt{d_k})} \end{aligned} \quad (3.8)$$

Self-attention is therefore defined as follows:

$$A(\mathbf{q}_i, \mathbf{K}, \mathbf{V}) = \sum_j \text{softmax}_j \left(\frac{\mathbf{q}_i^\top \mathbf{K}}{\sqrt{d_k}} \right) \cdot \mathbf{v}_j, \quad (3.9)$$

where d_k is the dimensionality of the key vectors. The alignment score $\frac{\mathbf{q}_i^\top \mathbf{k}_j}{\sqrt{d_k}}$ indicate how to weigh the value \mathbf{v}_j based on the query vector \mathbf{q}_i . The more similar a key vector \mathbf{k}_j is to \mathbf{q}_i , the more important is the corresponding value vector \mathbf{v}_j for the output vector.

Rather than computing attention in a single step, Vaswani et al. (2017) propose to decompose the self-attention operation in multiple heads to capture different aspects of the relationships between tokens. The dimensionality d is split into h fixed-size segments, with one segment per attention head. Each of the h heads uses three weight matrices (for queries, keys, and values) to project the segment

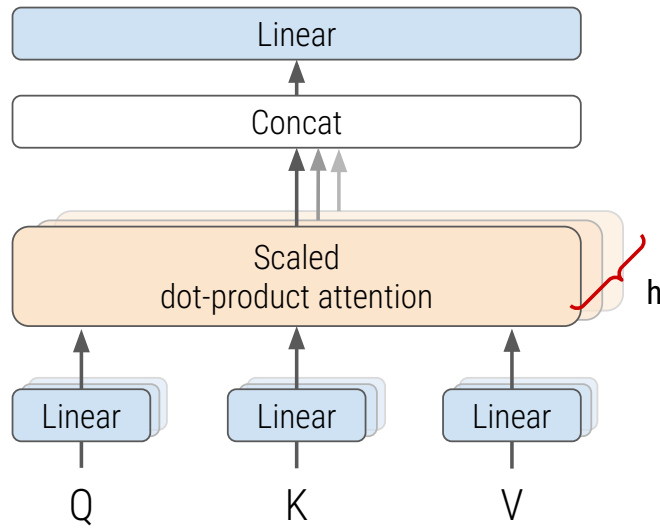


Figure 3.1 – Multi-head attention.

into different subspaces. Self-attention is then computed across each of the h segments in parallel, following Equation 3.9. The outputs of each head are then concatenated to form the complete attention output, and projected back into the original d -dimensional representation space. In other words, a distinct attention mechanism is employed for each segment of the input vector.

Given a projection matrix $\mathbf{W}_o \in \mathbb{R}^{d_v \times d}$, multi-head attention, illustrated in Figure 3.1, is defined as follows:

$$A^{(\text{MHA})}(\mathbf{q}_i, \mathbf{K}, \mathbf{V}) = \begin{pmatrix} A(\mathbf{q}_i \mathbf{W}_1^{(q)}, \mathbf{K} \mathbf{W}_1^{(k)}, \mathbf{V} \mathbf{W}_1^{(v)}) \\ A(\mathbf{q}_i \mathbf{W}_1^{(q)}, \mathbf{K} \mathbf{W}_2^{(k)}, \mathbf{V} \mathbf{W}_2^{(v)}) \\ \dots \\ A(\mathbf{q}_i \mathbf{W}_h^{(q)}, \mathbf{K} \mathbf{W}_h^{(k)}, \mathbf{V} \mathbf{W}_h^{(v)}) \end{pmatrix} \mathbf{W}_o \quad (3.10)$$

In addition, multi-head attention enables parallelized computation of attention across different representation subspaces.

3.1.2 Attention Layers in Transformers

In a Transformer, both encoder and decoder are composed by stacking a series of Transformer layers on top of each other. Each Transformer layer is characterized by a *multi-head attention* module and two position-wise feed-forward networks. To help the model train faster and more accurately, a residual connection (K. He et al. 2016) is added to all sublayers, followed by layer normalization.

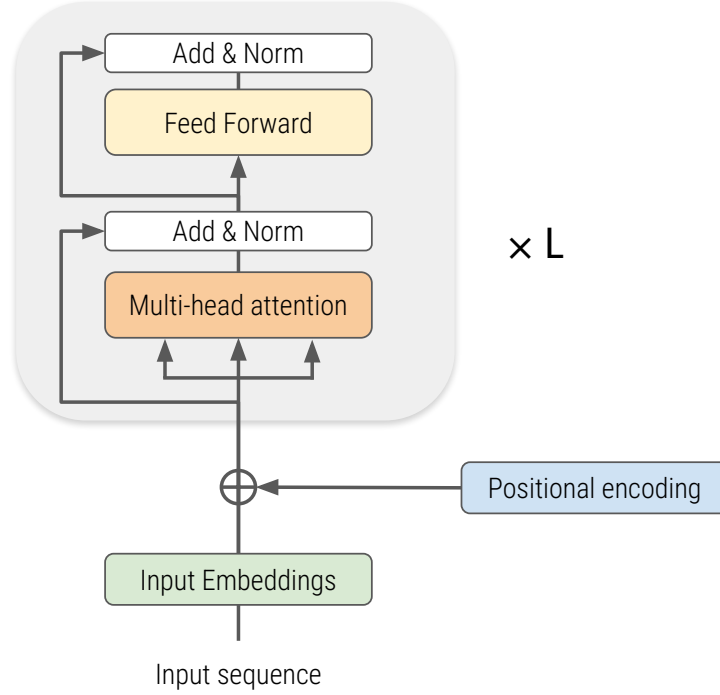


Figure 3.2 – Transformer encoder.

Given two sequences $\mathbf{Z} = (z_1, \dots, z_n)$ and $\mathbf{Z}' = (z'_1, \dots, z'_m)$, we use the following notation for multi-head attention between \mathbf{Z} and \mathbf{Z}' :

$$\text{MHA}(\mathbf{Z}, \mathbf{Z}'; \theta) = \begin{pmatrix} A^{(\text{MHA})}(z_1 \mathbf{W}^{(q)}, \mathbf{Z}' \mathbf{W}^{(k)}, \mathbf{Z}' \mathbf{W}^{(v)}) \\ A^{(\text{MHA})}(z_2 \mathbf{W}^{(q)}, \mathbf{Z}' \mathbf{W}^{(k)}, \mathbf{Z}' \mathbf{W}^{(v)}) \\ \dots \\ A^{(\text{MHA})}(z_n \mathbf{W}^{(q)}, \mathbf{Z}' \mathbf{W}^{(k)}, \mathbf{Z}' \mathbf{W}^{(v)}) \end{pmatrix}^\top \quad (3.11)$$

where $\mathbf{W}^{(q)} \in \mathbb{R}^{d \times d_q}$, $\mathbf{W}^{(k)} \in \mathbb{R}^{d \times d_k}$, and $\mathbf{W}^{(v)} \in \mathbb{R}^{d \times d_v}$ are layer-specific weight matrices for queries, keys, and values, respectively, and $\theta = \{\mathbf{W}^{(q)}, \mathbf{W}^{(k)}, \mathbf{W}^{(v)}\}$. In general, the dimensions d_q, d_k, d_v are set $\frac{d}{h}$.

Self-attention is a special case of attention where $\mathbf{Z} = \mathbf{Z}'$, *i.e.*, the contextualized and attended-to sequences are identical. We denote the self-attention operation as $\text{SA}(\mathbf{Z}; \theta) = \text{MHA}(\mathbf{Z}, \mathbf{Z}; \theta)$.

3.1.2.1 Self-Attention in the Encoder

In the encoder, illustrated in Figure 3.2, self-attention is used to map the input sequence (x_1, \dots, x_n) to a sequence of context-dependent vectors $(\bar{x}_1, \dots, \bar{x}_n)$.

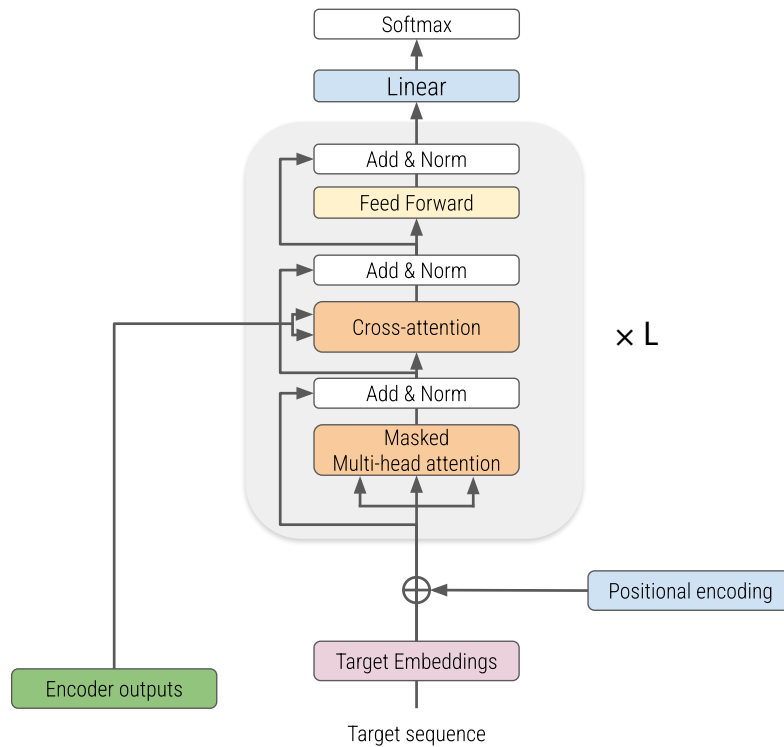


Figure 3.3 – Transformer decoder.

Each attention layer builds the queries, keys and values from the outputs of the previous encoder layer, and uses *bidirectional* self-attention to put each input token in relation with all input tokens in the sequence. Given $(\mathbf{x}_1^{(l)}, \dots, \mathbf{x}_n^{(l)})$ the input sequence to the l -th encoder layer, the outputs $(\mathbf{x}_1^{(l+1)}, \dots, \mathbf{x}_n^{(l+1)})$ constructed using bidirectional self-attention can be expressed as:

$$\mathbf{x}_1^{(l+1)}, \dots, \mathbf{x}_n^{(l+1)} = (\mathbf{x}_1^{(l)}, \dots, \mathbf{x}_n^{(l)}) + \text{SA}(\mathbf{x}_1^{(l)}, \dots, \mathbf{x}_n^{(l)}; \theta_{\text{enc}}^{(l+1)}). \quad (3.12)$$

Each encoder layer builds a contextualized representation of its input sequence, and the following layer further refines this context-dependent representation. Compared to RNNs, bidirectional self-attention reduces the amount of computation steps that information needs to flow from one element of the sequence to another. Therefore, information loss is reduced, making long-range dependencies more easily learnable.

3.1.2.2 Self-Attention in the Decoder

The decoder, depicted in Figure 3.3, models the distribution of a target sequence $(\mathbf{y}_1, \dots, \mathbf{y}_m)$ conditioned on the input sequence $(\mathbf{x}_1, \dots, \mathbf{x}_n)$. Each decoder layer contains three sublayers: *decoder self-attention*, *cross-attention*, and a module made

of two position-wise feed-forward networks. The final decoder layer is followed by a feed-forward network which produces a probability distribution over the whole vocabulary.

The decoder self-attention layer conditions each decoder output vector on all previous decoder input vectors. As opposed to the encoder, self-attention in the decoder is masked (*unidirectional* or *causal*) to ensure that each vector attends only to the previous positions. The output vectors $\mathbf{y}'_1, \dots, \mathbf{y}'_i$ generated by unidirectional self-attention are defined as follows:

$$\mathbf{y}'_1, \dots, \mathbf{y}'_i = (\mathbf{y}_1^{(l-1)}, \dots, \mathbf{y}_i^{(l-1)}) + \text{SA}(\mathbf{y}_1^{(l-1)}, \dots, \mathbf{y}_i^{(l-1)}; \theta_{\text{dec}}^{(l)}). \quad (3.13)$$

To condition the probability distribution of the next target vector on the encoder's input, *cross-attention* is applied to put $(\mathbf{y}'_1, \dots, \mathbf{y}'_i)$ into relation with all contextualized input vectors $(\bar{\mathbf{x}}_1, \dots, \bar{\mathbf{x}}_n)$. The output vectors $(\mathbf{y}_1^{(l)}, \dots, \mathbf{y}_i^{(l)})$ built using cross-attention are expressed as:

$$\mathbf{y}_1^{(l)}, \dots, \mathbf{y}_i^{(l)} = (\mathbf{y}'_1, \dots, \mathbf{y}'_i) + \text{MHA}(\mathbf{y}'_1, \dots, \mathbf{y}'_i; \bar{\mathbf{x}}_1, \dots, \bar{\mathbf{x}}_n; \theta_{\text{enc} \rightarrow \text{dec}}^{(l)}). \quad (3.14)$$

Cross-attention ensures that, the more similar a decoder query is to an encoder key, the more does the corresponding encoder value influence the decoder output representation.

3.1.3 Defining a distribution over the tokens

A Transformer establishes a conditional distribution of target vectors $(\mathbf{y}_1, \dots, \mathbf{y}_m)$ given a source sequence $(\mathbf{x}_1, \dots, \mathbf{x}_n)$. The probability distribution of the target vector sequence can be factorized to a product of conditional probability distribution of the target vector \mathbf{y}_i given the encoded hidden states $(\bar{\mathbf{x}}_1, \dots, \bar{\mathbf{x}}_n)$ and all previous target vectors $(\mathbf{y}_0, \dots, \mathbf{y}_{i-1})$. Formally:

$$p_{\theta}(\mathbf{y}_1, \dots, \mathbf{y}_m \mid \bar{\mathbf{x}}_1, \dots, \bar{\mathbf{x}}_n) = \prod_{i=1}^m p_{\theta}(\mathbf{y}_i \mid \mathbf{y}_0, \dots, \mathbf{y}_{i-1}; \bar{\mathbf{x}}_1, \dots, \bar{\mathbf{x}}_n). \quad (3.15)$$

Let $\bar{\mathbf{y}}_i$, where $i \in \{1, \dots, m\}$, be the representation of the target vector \mathbf{y}_i obtained by the last decoder layer. A probability distribution over the whole vocabulary can be obtained by mapping the representation $\bar{\mathbf{y}}_i$ to a logit, on which a softmax operation is then applied:

$$p_{\theta}(\mathbf{y}_i \mid \mathbf{y}_1, \dots, \mathbf{y}_{i-1}; \bar{\mathbf{x}}_1, \dots, \bar{\mathbf{x}}_n) = \text{Softmax}(\mathbf{W}_{\text{dec}}^{\top} \cdot \bar{\mathbf{y}}_i). \quad (3.16)$$

Likewise, a probability distribution over the vocabulary can be obtained from any contextualized representation $\bar{\mathbf{x}}_i$, where $i \in \{1, \dots, n\}$ and the token at position i is masked using a special token (additional details are provided in Section 3.2.1.1):

$$p_{\theta}(\mathbf{x}_i \mid \mathbf{x}_1, \dots, \mathbf{x}_n) = \text{Softmax}(\mathbf{W}_{enc}^{\top} \cdot \bar{\mathbf{x}}_i). \quad (3.17)$$

3.1.4 Sequential Information in Transformers

The position and order of words deeply impact the semantics of a sentence. By processing sequences token by token in a sequential manner, RNNs inherently integrate the order of the sequence. Unlike RNNs, Transformers simultaneously process each token in the sequence, making them entirely invariant to sequence ordering. Consequently, there is a need to explicitly incorporate information about the order of tokens into the Transformer.

A satisfactory positional encoding method must be deterministic, produce a unique encoding at each time step, generalize to longer sequences, and ensure that distance between any two elements are consistent across sequences with different lengths.

3.1.4.1 Absolute Position Encodings

Instead of integrating this encoding into the model itself, the dominant approach for preserving information about the sequence order is to modify the initial token representation with information about the token’s position in the sequence. These inputs $\mathbf{p}_i \in \mathbb{R}^d$ are called absolute position encodings (or embeddings) and can either be learned or fixed a priori. Absolute position encodings encode the absolute position of a token within a sequence, meaning that each token is assigned a fixed vector based on its position in the sequence. The commonly sought property for absolute position encodings is that $\mathbf{p}_i \cdot \mathbf{p}_j \geq \mathbf{p}_i \cdot \mathbf{p}_{j'}$ if $|i - j| \leq |i - j'|$. For a token t_i at position i with token representation \mathbf{x}_t , its initial input representation is denoted as $\mathbf{x}_i = \mathbf{x}_t + \mathbf{p}_i$.

Vaswani et al. (2017) propose a simple scheme for fixed absolute positional encodings, where each position is mapped to a vector using sinusoidal functions. These position encodings are not learned during training.

An alternative form of absolute positional encoding involves learning position embeddings \mathbf{p}_i jointly with the model during training (Devlin et al. 2018). Instead of using fixed functions, learned positional encodings introduce trainable parameters. Each position in the sequence is associated with a unique vector, and these vectors are treated as parameters that the model can learn during training.

Learned positional encodings offer flexibility as the model can adapt to various tasks and datasets. The embeddings are not predetermined by fixed functions, enabling the model to learn patterns related to position.

3.1.4.2 Relative Position Encodings

Although absolute positional encodings show satisfactory performance, they still face limitations. First, absolute positional encodings do not generalize well to sequences of lengths not seen at training time (Z. Dai et al. 2019). Another limitation of absolute positional encoding lies in its lack of translation invariance. Shifting the same sequence alters the absolute positions, leading to different attention patterns. In the case of learned positional encodings, there is a restriction on the number of tokens a model can handle. For instance, if a language model can only encode up to 1,024 positions, any sequence longer than 1,024 tokens cannot be processed by the model. Relative positional encoding address these issues by using a different vector for each pair of tokens, based on their relative distance (Shaw et al. 2018; C.-Z. A. Huang et al. 2018; Ke et al. 2020). Shaw et al. (2018) are the first to leverage pairwise distances to create positional encodings. To account for pairwise distances when computing alignment scores, the relative position embedding $\mathbf{b}_{ij}^K \in \mathbb{R}^{d_k}$ representing the distance between tokens t_i and t_j is added to the keys as follows:

$$a(\mathbf{q}_i, \mathbf{k}_j) = \frac{\mathbf{q}_i^\top (\mathbf{k}_j + \mathbf{b}_{ij}^K)}{\sqrt{d_k}}. \quad (3.18)$$

Pairwise distances are clipped beyond a maximum distance to allow for generalization to sequence lengths not seen during training. The relative position embedding \mathbf{b}_{ij}^K is expressed as:

$$\begin{aligned} \mathbf{b}_{ij}^K &= \mathbf{w}_{\text{clip}(j-i, k)}^K \\ \text{clip}(x, k) &= \max(-k, \min(k, x)), \end{aligned} \quad (3.19)$$

where $\mathbf{w}^K = (\mathbf{w}_{-k}^K, \dots, \mathbf{w}_k^K) \in \mathbb{R}^{2k \times d_a}$ is a learned vector.

Raffel et al. (2020) introduce a simplified form of relative position embeddings where the distance between every pair of tokens (t_i, t_j) is mapped to a scalar $\mathbf{b}_{ij} \in \mathbb{R}$, which is then added to the corresponding alignment score:

$$a(\mathbf{q}_i, \mathbf{k}_j) = \frac{(\mathbf{q}_i^\top \mathbf{k}_j) + \mathbf{b}_{ij}}{\sqrt{d_k}}. \quad (3.20)$$

3.1.5 Segment Information in Transformers

Some tasks require different inputs, and a commonly employed strategy to maintain the same architecture involves the use of segment embeddings (Devlin et al. 2018), which inform the model about sentence delineations within a sequence. These embeddings, not part of the original Transformer architecture, encode to which of two sentences (*segments*) a word belongs, aiding the model in distinguishing between two segments within the same input sequence. Unlike word embeddings and position encodings, segment embeddings remain identical across all tokens within a segment, and vary only between segments. These embeddings are then added to the input representations.

3.2 Transformer-based Pre-trained Language Models

In this section, we explore how Transformers can be employed to construct effective language models. In contrast to earlier methods that employ convolution and recurrent modules for feature extraction, Transformer-based Pre-trained Language Models learn text representations from Transformers. Transformer-based Pre-trained Language Models are typically categorized into three main types: *bidirectional* models based on the encoder only, *encoder-decoder* models leveraging the entire Transformer architecture, and *generative* models relying on the decoder alone.

3.2.1 Bidirectional Models

Bidirectional Pre-trained Language Models refer to models that employ the Transformer encoder. These models are pre-trained on large corpora and learn deep contextualized representations of words and phrases by jointly conditioning on left and right context in all self-attention layers. Bidirectional Transformer-based models are effective for capturing dependencies and contextual information in both directions, making them suitable for various natural language understanding tasks including sentiment analysis, *NER*, text classification, and more. The exploration of bidirectional Transformer-based Pre-trained Language Models began with *BERT*, introduced by Devlin et al. (2018), and has led to the development of a myriad of variants.

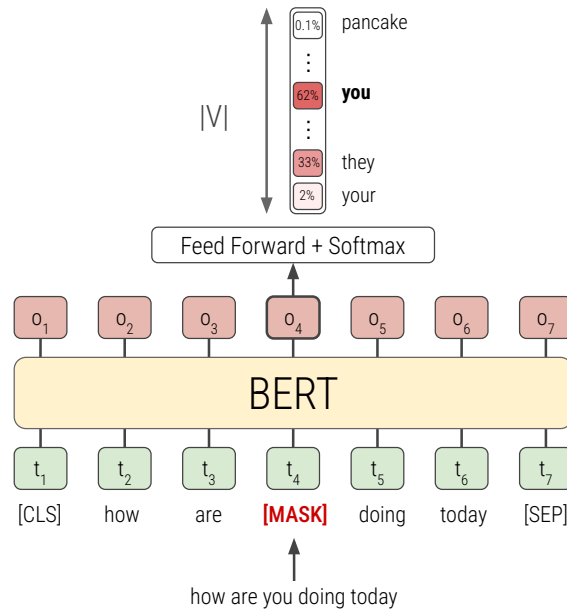


Figure 3.4 – Masked Language Modeling illustrated with the sentence "how are you doing today". Each token is mapped to a distribution over the vocabulary V . Depicted for the masked token "you". For the sake of clarity, segment embeddings and positional encodings are not shown.

3.2.1.1 BERT

BERT employs the encoder architecture of the Transformer while incorporating learned position embeddings. The model marked a paradigm shift in the construction of word representations. In contrast to prior models, which typically processed text in a unidirectional manner, BERT achieves bidirectionality by using self-attention.

Special tokens are inserted into the tokenized text to cater for different **NLP** tasks. A classification token [CLS] is prepended to the input sequence and is used to represent the whole sequence. In addition, a special separation token [SEP], indicating the end of the sequence, is added to the end of the sequence. In the case of text pairs, an extra [SEP] token is added between the pair to separate the texts.

Pre-training BERT **BERT** is trained on a large-scale dataset (Zhu et al. 2015), as a language model that operates at both the word-level and the sentence-level. The training involves two unsupervised tasks: Masked Language Modeling (**MLM**) and Next Sentence Prediction (**NSP**).

Illustrated in Figure 3.4, the **MLM** task consists in randomly masking out tokens and using all remaining tokens to recover the masked-out tokens in a self-

supervised fashion. Instead of following the same probability distribution as standard language models that process text in a unidirectional manner, **BERT** uses the following approximation:

$$P(\mathbf{w}) \propto \prod_{i \in C} P_{\theta}(w_i | \tilde{\mathbf{w}}) = \prod_{i \in C} P_{\theta}(w_i | \bar{\mathbf{x}}), \quad (3.21)$$

where C is a random set of tokens, with 15% of tokens selected to be in C , $\bar{\mathbf{x}}$ is the sequence of contextualized token representations obtained from $\tilde{\mathbf{w}}$, and $\tilde{\mathbf{w}}$ is the input sequence \mathbf{w} corrupted as follows:

$$\tilde{w}_t = \begin{cases} w_t, & \text{if } w_t \notin C \\ \text{mask token} & \text{if } w_t \in C, \text{ with probability } 80\% \\ \text{random token} & \text{if } w_t \in C, \text{ with probability } 10\% \\ w_t & \text{if } w_t \in C, \text{ with probability } 10\%. \end{cases} \quad (3.22)$$

Because the mask token is never used during fine-tuning, a discrepancy between pre-training and fine-tuning can occur. The process involves masking a selected token with the mask token 80% of the time, substituting it with a random token 10% of the time, and leaving it unchanged 10% of the time. The cross-entropy loss between the masked tokens and their predictions is minimized during pre-training. The primary benefit of the **MLM** task, in contrast to a causal language model, is that token representations are influenced by the whole sequence.

While **MLM** effectively captures bidirectional context to represent words, it does not explicitly capture the logical correlation between pairs of texts. To address this, the **NSP** task is introduced. This task involves determining whether two sentences follow each other and helps in modeling the relationship between texts. The training dataset is constructed such that half of the pairs are made of consecutive sequences, while for the other half the second sequence is randomly sampled from the corpus. Given a pair of sequences (s_1, s_2) , a binary single-layer feed-forward network classifier is trained to determine whether s_2 follows s_1 in the corpus. The classifier is fed with the **BERT** representation of the [CLS] token, which encodes both sequences, and outputs the probability that the sequences are successive sentences.

Fine-tuning BERT The knowledge gained during pre-training can then transferred to various downstream tasks through fine-tuning. The contextualized token representations obtained by the pre-trained **BERT** are fed to a feed-forward network built over the last encoder layer. While the parameters of the pre-trained encoder are reused and fine-tuned for the task, the additional layer is initialized

randomly and trained from scratch. However, it contains significantly fewer parameters. This layer can output predictions for individual tokens or the entire sequence, rendering the model suitable for tasks involving token classification and sequence classification, respectively.

The bidirectional capability of [BERT](#) addressed a crucial limitation in previous models, especially for tasks requiring a deep understanding of context and relationships between words. [BERT](#) demonstrated remarkable performance across various natural language understanding benchmarks (*e.g.*, sentiment analysis, question answering, text classification), showcasing the potential of bidirectional Transformer-based Pre-trained Language Models.

3.2.1.2 BERT Variants

The success of [BERT](#) has expanded exploration of bidirectional Transformer-based Pre-trained Language Models, with researchers and practitioners building upon the foundation laid by the model.

Some extensions of [BERT](#) have introduced different optimization objectives to further improve the quality of the contextual representations. To enhance the robustness and generalization capability of the model, A Robustly Optimized BERT Pretraining Approach ([RoBERTa](#)) ([Yinhan Liu et al. 2019](#)) removes the [NSP](#) objective of [BERT](#), and employs a bigger dataset. Additionally, it uses dynamic masking, wherein the selection and masking of tokens in a sentence vary across epochs. A Lite BERT ([ALBERT](#)) ([Lan et al. 2019](#)) introduces a variant of [NSP](#) where negative examples correspond to two consecutive segments with their order reversed. This modification of the [NSP](#) objective, emphasizing coherence, makes the task more challenging, thereby improving the robustness and generalization of the model. [SpanBERT](#) ([Joshi et al. 2020](#)) modify the masking strategy in [MLM](#) by masking contiguous random spans, rather than random tokens. Pre-trained using three type of language modeling tasks (unidirectional, bidirectional, and sequence-to-sequence prediction), [UniLM](#) ([L. Dong et al. 2019](#)) can be fine-tuned on both natural language understanding and generation tasks.

Furthermore, several adaptations of [BERT](#), pre-trained on specific datasets tailored to particular languages or domains, have been proposed. [SciBERT](#) ([Beltagy et al. 2019](#)) has been pre-trained on scientific texts and is specialized for tasks in the scientific research domain. Similarly, [BioBERT](#) ([J. Lee et al. 2020](#)) has been designed for biomedical texts. [CamemBERT](#) ([Martin et al. 2019](#)) and [FlauBERT](#) ([H. Le et al. 2019](#)), on the other hand, are French versions of [BERT](#), pre-trained on French texts, and adapted for French language understanding tasks. For usage across different languages, multilingual pre-trained models such as [mBERT](#) (De-

vlin et al. 2018), XLM (Lample and Conneau 2019), and XLM-RoBERTa (Conneau et al. 2019) have been proposed.

To enable the use of models in resource-constrained scenarios, and given the quadratic complexity of BERT with respect to the sequence length in both memory and time, researchers have explored approaches to reduce the number of parameters. ALBERT achieves efficiency by sharing parameters across layers and reducing the rank of the embedding matrix. DistilBERT (Sanh et al. 2019) simplifies BERT by using parameter reduction techniques such as factorized embedding parameterization and cross-layer parameter sharing. TinyBERT (Jiao et al. 2019) further simplifies the model for increased efficiency. This research is part of a broader effort to make Transformers more computationally efficient, which will be further discussed in Section 3.3.

3.2.2 Generative Models

In contrast to bidirectional Pre-trained Language Models that focus on predicting labels for given inputs, the goal of *generative* Pre-trained Language Models is to produce new text of arbitrary length that resembles human language. These models employ the Transformer decoder to generate content in an autoregressive fashion. Given their ability to capture contextual information and generate diverse and contextually fitting text, generative Pre-trained Language Models play a pivotal role in improving the quality of text generation for natural language generation tasks (e.g., machine translation, summarization, text completion). Generative Transformer-based Pre-trained Language Models can either use the entire Transformer architecture (*encoder-decoder* models) or solely the decoder component (*decoder-only* models).

3.2.2.1 Training and Inference Framework

Training and inference for generative models differ from those of bidirectional models in several aspects.

Training While bidirectional models are typically trained for specific pre-defined tasks, generative language models are trained to predict the next token in a sequence, given the preceding ones (*autoregressive language modeling*). However, compounding errors might occur from incorrect predictions, leading the decoder to potentially drift too far away from the target. To mitigate this issue and guide the training process, the *teacher forcing* (i.e., maximum likelihood) strategy is used: to predict the next token, the decoder is fed with the ground-truth target tokens, rather than its own predictions from the previous step. Furthermore, using

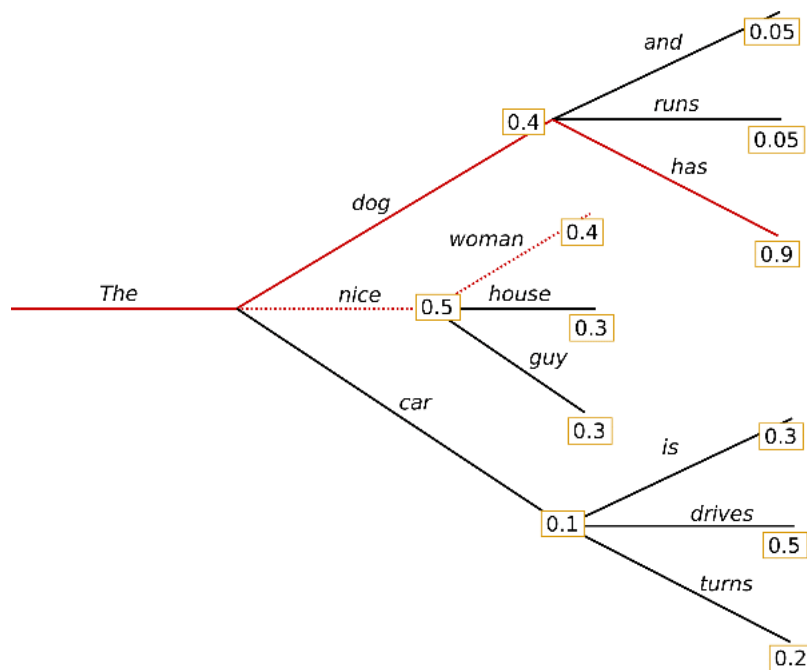


Figure 3.5 – Illustration of beam search, where $K = 2$. From Von Platen (2020).

ground-truth target tokens as inputs accelerates convergence, as stronger gradient signals are provided during backpropagation. More sophisticated learning strategies have emerged, including the use of reinforcement-learning models to optimize sequence-level metrics such as ROUGE (Paulus et al. 2017), BLEU (Ranzato et al. 2015), or metrics based on question answering (Scialom et al. 2019).

Inference Teacher forcing is obviously not applicable at inference time. Instead, the sequential generation process requires the model to be fed its own predictions from the previous step as inputs to generate the next token. Different strategies can be used to determine how the model predicts the next element of the sequence.

Greedy search consists in selecting the token with the highest probability at each step. While simple and computationally efficient, it misses high probabilities that can be found in posterior tokens.

To reduce this risk, *beam search* extends greedy search by maintaining a fixed number K of sequences with the highest probabilities. At each step, it picks the K best sequences so far based on their joint probabilities. Finally, the sequence with the highest probability is selected as the output sequence. Figure 3.5 demonstrates an instance of beam search with two beams.

To ensure that the less probable tokens should not have any chance of being selected, *top-k sampling* (Fan et al. 2018) filters the k most likely next tokens and

redistributes the probability mass among those k tokens only. Alternatively, *nucleus sampling* chooses from the smallest possible set of tokens whose cumulative probability exceeds a certain threshold p . The probability mass is then redistributed among this set of tokens. Nucleus sampling balances randomness and predictability better than traditional sampling.

3.2.2.2 Encoder-decoder Models

Encoder-decoder Transformer-based Pre-trained Language Models use the original Transformer architecture, consisting of both an encoder and a decoder. The encoder processes an input sequence, producing contextualized representations of each token. The decoder attends to these representations and generates the output tokens one at a time, considering the context provided by the encoder. This architecture is particularly suited for sequence-to-sequence tasks, where the goal is to transform a source sequence into a corresponding target sequence. The encoder-decoder setup is commonly used in sequence-to-sequence tasks like machine translation and text summarization.

BART Bidirectional and Auto-Regressive Transformers (BART) (M. Lewis et al. 2019) is pre-trained with a denoising objective, where the model is trained to reconstruct the original sequence from a corrupted version. BART extends the MLM approach by adding more perturbations: replacing text spans with a single mask token (*text infilling*), permuting sentences, deleting or replacing tokens, and rotating documents. The corrupted sequence is encoded using the bidirectional encoder, and the decoder is trained to reconstruct the original sequence. When fine-tuned, BART shows remarkable results for natural language generation tasks such as text summarization, machine translation, question answering. Additionally, the model also works well for natural language understanding tasks, *e.g.*, NER, Natural Language Inference (NLI), and coreference resolution. Inspired by the success of BART, Yinhan Liu et al. (2020) introduce mBART, a multilingual version of BART pre-trained on large-scale monolingual corpora in many languages. mBART can be fine-tuned for any of the language pairs, whether in supervised or unsupervised settings, without necessitating task-specific or language-specific adjustments or initialization methods.

Pegasus (Jingqing Zhang et al. 2020) is specifically tailored for abstractive text summarization. It is trained using the MLM strategy coupled with the *Gap-Sentences Generation* task, a novel pre-training approach intentionally similar to summarization. For each document, a subset comprising 15% of sentences is selected and masked from the original documents. These masked sentence (*gap*

sentences) are combined to form a pseudo-summary, which serves as the training label during pre-training. To refine the pseudo-summary and approach summary-like quality, the top- m sentences are selected based on their importance, determined by the Recall-Oriented Understudy for Gisting Evaluation (ROUGE)-1 score between each gap sentence and the rest of the document. The model is then tasked to generate the gap sentences using the remaining sentences.

T5 Text-to-Text Transfer Transformer (T₅) (Raffel et al. 2020) converts all NLP tasks into a sequence-to-sequence problem: for any task, the input of the encoder is a task-specific prefix (e.g., “Summarize:”) followed by the task’s input (e.g., a sequence of tokens from an article), and the decoder predicts the task’s output (e.g., a sequence of tokens summarizing the input article). The pre-training includes a mixture of both supervised and unsupervised tasks. Supervised pre-training is conducted on downstream tasks (translation, question answering, and more). Unsupervised pre-training uses corrupted tokens, by randomly removing 15% of the tokens and replacing them with individual (sentinel) tokens. Given the corrupted sequence encoded by the encoder and the original sequence fed to the decoder, T₅ has to reconstruct the dropped out tokens. Casting all NLP tasks into the same sequence-to-sequence problem allows for the use of the same model, loss function, and hyperparameters across a diverse set of tasks.

Limitations of encoder-decoder models Tasks such as machine translation and summarization commonly favor encoder-decoder models, as a holistic understanding of the input sequence is crucial for generating accurate and coherent outputs. However, encoder-decoder models face several challenges compared to decoder-only models. The introduction of an encoder adds more parameters to the model, which can be a limitation in scenarios where model size is a critical consideration. Furthermore, the incorporation of bidirectional self-attention and cross-attention introduces additional computational overhead. The interaction between the encoder and decoder can render training more complex, potentially making the model more susceptible to overfitting. Lastly, the encoder may encode redundant or irrelevant information from the input sequence, and the decoder has to learn to filter and use this information effectively.

3.2.2.3 Decoder-only Models

The use of decoder-only Transformer architectures in Pre-trained Language Models has seen a recent surge, with several groundbreaking models (Radford et al. 2018; T. Brown et al. 2020; Ouyang et al. 2022; Touvron et al. 2023) emerging. Decoder-only Pre-trained Language Models, also referred to as *autoregressive*

or *causal* language models, remove the Transformer encoder and cross-attention layers. Each source sequence is concatenated with the corresponding target sequence to form a single input sequence, which is then used to train a language model. This design significantly simplifies the architecture and provides potential advantages in computational efficiency and ease of training.

Among the most performant generative models of the decade is the series of Generative Pre-trained Transformer (GPT) models introduced by (Radford et al. 2018). GPT is the first autoregressive language model that uses a Transformer decoder as its backbone. It learns to predict the next word in a sequence using autoregressive language modeling. GPT surpassed state-of-the-art NLPs models that were trained in a supervised fashion with task-specific architectures. In addition, it improved zero-shot performance in various NLP tasks such as question answering, schema resolution, and sentiment analysis. GPT established the core architecture for the GPT-series models and laid down the fundamental principle to model natural language text, *i.e.*, predicting the next word.

To learn an even stronger language model, Radford et al. (2019) propose GPT-2, a much larger version of GPT that increases the number of parameters tenfold, from 100 million to 1.5 billion. Similar to T5 (Raffel et al. 2020), GPT-2 seeks to perform tasks via self-supervised language modeling, without explicit fine-tuning with labeled data. To achieve this, Radford et al. (2019) introduce *task conditioning*, a probabilistic form for multi-task learning, which consists in predicting the output based on the input and task information, *i.e.*, $P(\text{output} \mid \text{input}, \text{task})$. Task conditioning is performed by providing examples of natural language instructions to perform a task, *e.g.*, for English to French translation, the model is given an English sentence followed by “French: ”. Therefore, input to GPT-2 is given in a format which expects the model to understand the nature of the task. Trained on a sufficiently extensive and large dataset (Radford et al. 2019), GPT-2 achieved state-of-the-art performance on language modeling benchmarks (M. Marcus et al. 1993; Chelba et al. 2013; Merity et al. 2016) in zero-shot scenarios. On downstream tasks such as question answering, summarization, and translation, GPT-2 demonstrates the ability to learn these tasks directly from raw text, without relying on task-specific training data. The model’s versatility in handling various tasks in a zero-shot setting suggests that high-capacity models, trained to optimize the likelihood of diverse text corpora, inherently learn how to perform a remarkable range of tasks without the need for explicit supervision.

3.2.2.4 Large Language Models

These decoder-only Pre-trained Language Models have showcased their ability to generate coherent and contextually relevant text, establishing them as versatile

tools for various NLP tasks. Researchers have observed that scaling Pre-trained Language Models, whether by increasing model size or training data, frequently results in enhanced model capacity for a variety of downstream tasks. This phenomenon aligns with the scaling law, as suggested by J. Kaplan et al. (2020). The success of decoder-only Pre-trained Language Models has spurred further development of larger and more sophisticated decoder-only Pre-trained Language Models, now referred to as Large Language Models.

In particular, the next iteration in the GPT series, GPT-3 (T. Brown et al. 2020), represents a significant milestone in the progression from Pre-trained Language Models to Large Language Models. GPT-3 is a slightly modified version of GPT-2 that demonstrates a significant capacity leap by scaling to a staggering size of 175 billion of parameters. GPT-3 introduced the concept of *in-context* learning, which allows the model to perform specific tasks by conditioning its responses on context provided in the prompt. In-context learning, also referred to as *prompting*, encompasses zero-shot, one-shot, and few-shot learning. GPT-3 uses the same architecture as GPT-2 with the exception that attention patterns are sparse at alternating layers (see Section 3.3). Pre-trained on an even larger dataset, GPT-3 has empirically shown that scaling Pre-trained Language Models to a significant size and formulating text to guide models to perform specific tasks (in-context learning) can lead to a huge increase in model capacity, especially in few and zero-shot learning scenarios.

The series of GPT models has allowed significant progress in the field of NLP by demonstrating the power of Large Language Models. Building on the success of the GPT series, a myriad of Large Language Models have been released (Scao et al. 2022; Chowdhery et al. 2022; Touvron et al. 2023). The progress in Large Language Models has expanded into more specific domains, with models tailored for specialized tasks such as medical language processing (Thirunavukarasu et al. 2023), scientific research (H. Wang et al. 2023), website development (Junjie Wang et al. 2023), and code generation (F. F. Xu et al. 2022). A groundbreaking application of Large Language Models is ChatGPT,¹ an adaptation of the GPT-series designed for dialogue. ChatGPT exhibits exceptional conversational abilities with humans and has sparked a wealth of reviews and discussions on the advancements of Large Language Models (Zhao et al. 2023; Mohamadi et al. 2023; Hadi et al. 2023).

In recent years, the size of Pre-trained Language Models has been scaled from a few million parameters (BERT, 110M) to hundreds of billions of parameters (PaLM, 540B). This scaling has boosted capabilities, enabling models to generate even more coherent and natural-sounding text and further pushing the boundaries of NLP. Despite substantial progress, there are still limitations associated

1. <https://openai.com/blog/chatgpt>

with Large Language Models. First, pre-training effective Large Language Models is challenging due to the very high computational needs and the sensitivity to data quality and training settings. In addition, these have a tendency to generate inaccurate information, either conflicting with existing sources or unverifiable by available sources (Bang et al. 2023). Even robust Large Language Models such as ChatGPT encounter significant difficulties in managing hallucinations within generated texts. Another significant challenge for Large Language Models is their limitation in solving tasks that require the latest knowledge beyond what was available in their training data. Fine-tuning such large models with new data is a costly process and can potentially lead to catastrophic forgetting when incrementally training these models. Keeping up with and incorporating information from real-time or rapidly evolving context remains an open research problem (Yao et al. 2023). Besides, Large Language Models may struggle with tasks that demand generating structured data (J. Jiang et al. 2023) or require domain-specific knowledge (Junjie Ye et al. 2023). Incorporating this knowledge into these models while preserving their original capabilities is non-trivial. Furthermore, Large Language models have been shown to internalize, spread, and potentially amplify harmful information present in the training data they are exposed to. This often includes toxic language such as offensiveness, hate speech, and insults (Gehman et al. 2020), as well as social biases such as stereotypes directed towards individuals with specific demographic identities (Sheng et al. 2021). Addressing all of these limitations remains an ongoing area of research to enhance the robustness and ethical deployment of Large Language Models in various applications.

3.2.3 Conclusion

Transformer-based Pre-trained Language models have marked a significant milestone in Artificial Intelligence (AI). With sophisticated pre-training strategies and a vast number of model parameters, these models excel in absorbing knowledge from large amounts of data through unsupervised pre-training. The knowledge acquired can be leveraged for a wide range of tasks in NLP. Consequently, Pre-trained Language Models are widely embraced as the foundational framework for downstream tasks, replacing the traditional approach of training models from scratch.

The contributions presented in this PhD thesis leverage the capabilities of Transformer-based Pre-trained Language Models for applications in Document Understanding, focusing specifically on the earlier iteration (*e.g.*, BERT and BART) rather than the Large Language Models that emerged toward the end of our research timeline.

3.3 Long-range Modeling

In real-world scenarios, long texts are a major information medium documenting human activities, *e.g.*, academic articles, official reports, and meeting transcripts. Reading, analyzing, and extracting important information from large volumes of long texts poses a challenge for humans. Therefore, a compelling need arises for NLP systems to model long texts and extract information of human interest. Broadly, the objective of *long-range modeling* is to capture salient semantics from long texts through informative representations, which hold utility for diverse downstream applications such as long document summarization (Cohan et al. 2018; Sharma et al. 2019) and long question answering (Dasigi et al. 2021).

Handling long texts poses a challenge in NLP due to the *length limitation* imposed by Transformers on input sequences. Tokens exceeding the predefined maximum context length, typically shorter than the length of short texts, are discarded. A straightforward method involves truncating the input text to the pre-defined maximum length (M. Lewis et al. 2019). Yet, with increasing length, salient information in a long document tends to be evenly distributed. Therefore, truncating the text may lead to substantial loss of information (Koh et al. 2022). Another approach consists in splitting the text into chunks, each fitting within the maximum length of the model. Each chunk is processed independently, and the resulting outputs are then aggregated. However, the model’s receptive field is confined to a chunk, causing the disruption of long-range dependencies that extend across multiple chunks (Ding et al. 2020). An alternative solution involves identifying and concatenating relevant sections of the text into a sequence, which is then processed by the model. However, employing a two-stage pipeline can lead to discrepancies between the two modules. Therefore, modeling long texts remains an ongoing research problem that necessitates in-depth exploration.

Furthermore, computational efficiency cannot be overlooked. As the document’s length increases, the time and memory requirements needed to model the text with standard Transformers grow quadratically, adding a substantial burden for practical applications. This high computational cost originates from various factors, with the computation of self-attention being a major contributor. Given n the sequence length, h the number of attention heads, and d the dimension of the representation space, the computational complexity for a self-attention operation on a single sequence is $\mathcal{O}(dn^2)$. The memory complexity to compute the attention matrix is $\mathcal{O}(dn + hn^2)$, the first term being the memory required to store keys and queries, and the second term referring to the scalar attention values produced by each head. Hence, the \mathbf{QK}^\top matrix multiplication alone results in n^2 time and memory requirements, constraining the use of Transformers models to short sequences, typically capped at 512 or 1,024 tokens. Furthermore, the two feed-

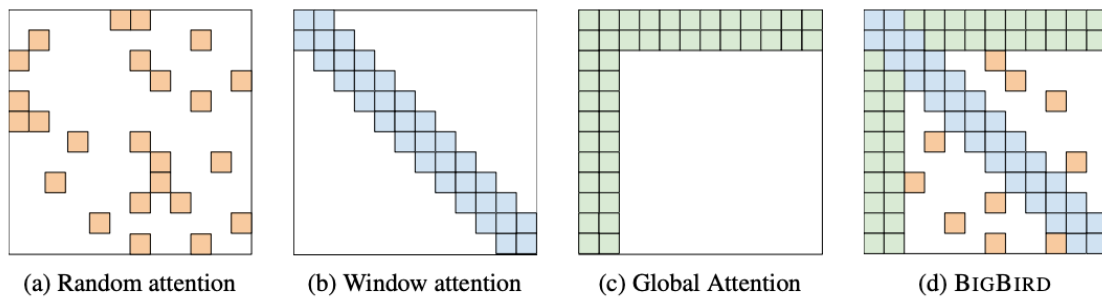


Figure 3.6 – Attention mechanisms used in BigBird. Illustration from Zaheer et al. (2020).

forward network components in each Transformer block significantly contribute to the cost of Transformers. While having a linear complexity with respect to sequence length, feed-forward networks are still, in practice, resource-intensive.

Additionally, long documents harbor distinct attributes when compared to shorter texts. As long texts are typically domain-specific documents with complex hierarchical structures, there is a need to consider long-range dependency (distant tokens may share semantic relationships with each other), inter-sentence relations (semantic connections between different sentences in the text), and discourse structure (long documents typically feature complex discourse structures, comprising sections and paragraphs) (Z. Dong et al. 2023).

We focus on modeling advances and architectural innovations that tackle the quadratic complexity issue of the self-attention mechanism. Furthermore, we explore benchmarks specifically designed for evaluating long-range modeling capabilities.

3.3.1 Long-range Models

To alleviate the cost of Transformers, a diversity of efficient self-attention model variants (Tay et al. 2020b) have been proposed over the past few years. Termed as *Long-range Transformers*, these variants play a vital role in applications that model long sequences. Based on their core techniques and primary use case, long-range Transformers can be grouped into three categories (Qin et al. 2022): *sparse patterns*, *recurrence*, and *low-rank and kernel* methods.

3.3.1.1 Sparse Patterns

The earliest modifications to self-attention involve applying pattern-based methods to sparsify the attention matrix. By computing attention solely on a limited

number of query-key pairs, this approach restricts the field of view to specific patterns. The key idea is to relax the constraint that a single layer needs to aggregate information from any two tokens. Although the attention of each layer is not dense, the receptive field can be increased as multiple layers are stacked.

As an initial attempt, the Sparse Transformer (Child et al. 2019) introduces a two-dimensional factorization of the attention matrix, where the network can attend to all positions through two steps of sparse attention. Half of the attention heads attend only to preceding elements in the sequence, while the other half attend to predesignated indices spread evenly throughout the sequence.

Longformer (Beltagy et al. 2020), a variant of Sparse Transformer, employs self-attention on both *local* and *global* contexts by introducing three attention patterns: *sliding window attention*, *dilated window attention*, and *global attention*. The key concept underlying the first two patterns is similar to convolution: the most important information is supposedly contained in the neighbourhoods of the tokens (Yixin Liu et al. 2022). *Sliding window attention* constrains the field of view for each token to a k -sized window centered on the token. Although a token can only attend to itself and its neighbours in a single layer, sliding window attention enables a multi-layer Transformer to achieve a receptive field covering the entire sequence. Due to the need for multiple layers to incorporate information from the complete sequence, the sliding window can be *dilated* to increase the receptive field without increasing computation. However, dilated sliding window attention alone does not suffice to produce task-specific representations: some tokens are so important that it is highly beneficial that each token is connected to them and conversely (e.g., through a single layer, the [CLS] token needs to have access to all input tokens for sequence classification tasks). *Global attention* addresses this issue by allowing s fixed, user-defined tokens to attend to every other token and vice-versa, enabling the model to learn task-specific representations.

BigBird (Zaheer et al. 2020) extends Longformer by adding *random pattern attention*, which allows every token to attend to r tokens chosen randomly, where r is a small constant number. The intuition behind this mechanism is that the path lengths in a randomly connected graph are on average logarithmic. The attention mechanisms employed in BigBird are depicted in Figure 3.6.

The Extended Transformer Construction (ETC) model (Ainslie et al. 2020) represents another iteration within the Sparse Transformer family. It introduces a novel *global-local* attention mechanism, encompassing four distinctive components: *global-to-global*, *global-to-local*, *local-to-global*, and *local-to-local* attentions. In addition to the original input, ETC integrates n_g auxiliary tokens at the beginning of the sequence, functioning as global tokens for participating in *global-to-** and **-to-global* attention processes. The *local-to-local* component uses a sliding window of size k . Notably, ETC's approach closely resembles that of Longformer in its

incorporation of global auxiliary tokens, which function as trainable parameters and can be interpreted as a form of model memory that pools across the sequence to collect global sequence information.

The Sparse Transformer has a loglinear time and memory complexity, while Longformer, BigBird, and ETC operates with a linear time and memory complexity. None of these models introduces new parameters beyond the Transformer model. Given the global attention mechanism, computing causal masks (Section 3.1.2.2) becomes unfeasible. As a result, the attention mechanisms employed by Longformer, BigBird, and ETC are unsuitable for use in an autoregressive setting. Nevertheless, they can be effectively applied in sequence-to-sequence modeling, employed in the encoder while standard self-attention is retained in the decoder. Both Longformer and BigBird, as well as ETC, have the capacity to handle up to 4,096 tokens.

3.3.1.2 Recurrence

Recurrence and compressed memory approaches incorporate *segment-level recurrence* into Transformer models to lengthen their attention span. The underlying concept of segment-based recurrence methods is to consider blocks of local receptive fields by chunking the input sequence into segments, and then process them in series via recurrence, as in RNNs.

Rather than attempting to reduce the cost of self-attention, Z. Dai et al. (2019) take inspiration from RNNs and propose Transformer-XL, a causal language model that introduces a segment-based recurrence mechanism to connect adjacent segments. In Transformer-XL, segments are sequentially fed to the model, and tokens within a segment attend to the rest of the segment *and* to the hidden states of the previous segment. Hence, after the first segment, tokens in subsequent segments will always have an immediate context size of n . By stacking multiple attention layers, the receptive field can be increased to multiple previous segments. In addition, this recurrence mechanism provides context for tokens in the beginning of a new segment.

XLNet (Z. Yang et al. 2019) leverages both autoregressive and bidirectional language modeling. Unlike traditional autoregressive models that rely on fixed forward/backward factorization orders, XLNet maximizes the expected log likelihood of a sequence across all possible permutations of factorization orders. This approach allows each position in the sequence to consider tokens from both left and right, creating a bidirectional context. Additionally, XLNet incorporates the segment recurrence mechanism and relative encoding scheme of Transformer-XL during pre-training. This integration empirically improves the model's performance, specifically for tasks involving long text sequences.

3.3.1.3 Approximating the Attention Mechanism

Another approach to improve the efficiency of Transformer models is to approximate the self-attention mechanism through techniques such as *learnable patterns*, *low-rank approximation*, or *kernelization*. The idea revolves around mathematically redefining the self-attention mechanism, which eliminates the need to explicitly compute the $n \times n$ matrix.

Reformer (Kitaev et al. 2020) uses learnable patterns that enable the model to learn the access pattern in a data-driven fashion. Learnable patterns facilitates a more global view of the sequence while maintaining the efficiency benefits of fixed patterns approaches. Reformer introduces Locality Sensitive Hashing (LSH) attention, a novel attention mechanism that consists in sharing parameters between Q and K , and clustering tokens into chunks. This concept is rooted in the idea that if the sequence is long, $\text{softmax}(QK^\top)$ only puts significant weight on very few key vectors for each query vector. Hence, given a query q , $\text{softmax}(qK)$ can be approximated by using only the keys that have a high cosine similarity with q . Using the LSH algorithm, query vectors are hashed into buckets of similar vectors. Attention is then computed among each bucket. If the bucket size is appropriately selected, the time and memory complexity of Reformer is $\mathcal{O}(n \log n)$.

Rather than approximating the softmax operator, alternative approaches have aimed to approximate the full attention mechanism using low-rank approximations. In Linformer (S. Wang et al. 2020), the keys and values are projected to a lower-dimensional representation $k \times d$, where $k < n$. As k does not depend on the sequence length n , the time and memory complexity of Linformer is linear. There is only a minimal parameter costs of the Linformer due to the extra nk length projections. If k is sufficiently small, there is negligible parameter costs incurred.

To estimate standard full-rank-attention Transformers without relying on any prior such as sparsity or low-rankness, Choromanski et al. (2020) generalize Linformer by proposing a kernel-based approach that uses a generalized attention framework to approximate any attention matrix.

Reformer can be used in both bidirectional and autoregressive settings. In Linformer, projection along the length dimension n induces the mixing of sequence information. Similarly, the randomized feature map and the approximations involved in the kernel-based approach might not inherently preserve the sequential order required for causal masking. Therefore, it is non-trivial to maintain causal masking and/or prevent mixing of past and future information when computing attention in both Linformer and Performer. Consequently, both Linformer's and Performer's attention approximations are unsuitable for deployment in an

autoregressive setting. Similar to Longformer, BigBird, and ETC, these attention mechanisms are restricted to the encoder of the Transformer.

3.3.2 Benchmarking Long-range Models

The broad array of efficient Transformers that has emerged to address the challenge of long-range modeling have been mostly evaluated using different sets of downstream tasks and datasets. Longformer and BigBird were evaluated on question answering (Z. Yang et al. 2018; Welbl et al. 2018) and text classification (Maas et al. 2011; Kiesel et al. 2019). ETC’s evaluation extends to information extraction tasks (L. Xiong et al. 2019). XLNet’s performance is assessed using machine reading comprehension (Lai et al. 2017) and document ranking². Linformer is evaluated on easier tasks such as sentiment classification (Maas et al. 2011; Socher et al. 2013), NLI (A. Wang et al. 2018) and textual similarity (Zhiguo Wang et al. 2017), where simpler models such as Convolutional Neural Networks (CNNs) perform well. On the other hand, the Sparse Transformer, Reformer, and Performer have their performance assessed on image generation tasks (Parmar et al. 2018). The large diversity of tasks and datasets used complicates the comparison of models and the assessment of their relative strengths and weaknesses.

Furthermore, intrinsic metrics such as perplexity or Bits-per-character (BPC) are widely used to assess the performance of efficient Transformers. However, an increasing amount of literature shows that predicting the next token is mostly a local task that does not require modeling long-range dependencies (Khandelwal et al. 2018; Sun et al. 2021).

In this section, we describe unified benchmarks designed to shed light on the ability of these architectures to reason in long-context scenarios, and study their performance in handling NLP tasks.

3.3.2.1 Long-Range Benchmarks

Tay et al. (2020a) introduce a systematic and unified benchmark, Long-Range Arena (LRA), designed to evaluate the ability of a model to reason in long-context scenarios. This benchmark is a suite of tasks with sequences ranging from 1,000 to 16K tokens, encompassing various data types and modalities (text, natural and synthetic images, mathematical expressions). This benchmark was created based on a set of specific requirements and criteria. First, LRA is general: all long-range Transformer models are applicable to the tasks. The tasks are intentionally designed to promote simple models rather than cumbersome pipelined approaches.

2. <https://lemurproject.org/clueweb09/>

Furthermore, the tasks are crafted to be challenging enough to ensure there is room for improvement. The input sequences are reasonably long, and the set of tasks allows to assess different capabilities of models. Finally, *LRA* is deliberately non-resource intensive and accessible. However, only two out of these five tasks involve natural language, restricting the relevance of *LRA* for *NLP*. Furthermore, *LRA* artificially increases the length of natural language sequences through character tokenization, and truncates each example at 4,000 characters, equivalent to less than 1,000 words. This exempts models from dealing with the complex long-range interactions present in long texts.

To extend evaluation beyond sentences and paragraphs, Shaham et al. (2022) propose Standardized CompaRison Over Long Language Sequences (*SCROLLS*), an *NLP* benchmark featuring datasets with long texts, ranging from an average of 1,700 words to over 51,000. *SCROLLS* consists of seven datasets covering various domains and tasks: summarization from government reports (*GovReport*), TV shows transcripts (*SummScreenFD*), and meetings (*QMSum*), question answering from scientific articles (*Qasper*), books (*NarrativeQA*), and stories (*QuALITY*), and *NLI* from non-disclosure agreements (*Contract NLI*). Every task is reformulated as a sequence-to-sequence problem for a simple unified format. *SCROLLS* presents a challenge not only due to the length of its inputs but also because it requires models to process long-range interactions across different sections.

3.3.2.2 On the Effectiveness of Long-range Models for *NLP* Tasks

To validate the effectiveness and long-range ability of long-range Transformers on language tasks and uncover the underlying factors behind model behaviors, Qin et al. (2022) benchmark different long-range Transformer models for *NLP* tasks characterized by long sequences. Five complex, long-text *NLP* tasks are considered, covering a wide spectrum of typical language scenarios: token/span-level prediction, sequence-level classification, and sequence-to-sequence generation.

Longformer and BigBird are used to assess the performance of sparse pattern approaches (Section 3.3.1.1). In coreference resolution, which consists in identifying mention spans and clustering them into entities, Qin et al. (2022) find that using larger sliding windows can be advantageous, but this advantage tends to level off or even decline after a certain point. In tasks where the quantity of guiding text, *i.e.*, specific textual information provided explicitly to guide the model's behavior, is limited—such as questions in question answering—using them as global tokens can enhance attention and substantially improve overall performance. Additionally, Qin et al. (2022) find a connection between long-range attention, global tokens, and the selectivity of sequence-to-sequence problems, which ultimately enhances the decoding process. Globally, the authors show that

pattern-based methods, despite being a widely adopted approach, do not effectively capture long distance information.

The effectiveness of recurrence-based methods (Section 3.3.1.2) is evaluated using XLNet. In various tasks, Qin et al. (2022) show that the memory of recurrence models tends to enhance performance, demonstrating the advantage of using past hidden states in Transformers. Nevertheless, XLNet falls short in maximizing the potential of past tokens, as it gives relatively less attention to distant information. This could be attributed to XLNet’s pretraining objective of predicting masked tokens, which does not consistently require long-range context (Sun et al. 2021). Moreover, the application of the stop-gradient technique might impede the model’s ability to efficiently focus on memories.

Performer is used as a kernel-based model (Section 3.3.1.3). It is found that the approximation technique of Performer demonstrates strong performance with shallow networks. However, when applied to deeply stacked Transformer layers, it encounters significant error accumulation issues. This leads to a notable drop in performance, which is considered unacceptable even for the base version of Transformer encoders.

Drawing from their discoveries, Qin et al. (2022) offer a few recommendations. For typical tasks like sequence classification or token-level prediction, it remains effective to divide inputs into chunks and use short-range Transformer models. In cases where explicit guiding text such as queries is available, models based on sparse patterns and featuring a global token mechanism are preferable. For sequence-to-sequence problems, leveraging long-range Transformers with pre-trained checkpoints yields superior performance.

3.3.3 Conclusion

The exploration of long-range modeling has been marked by continuous efforts to reduce the cost of Transformers to efficiently model long texts. The quest for ideal long-range models demands finding an equilibrium. These models should address the quadratic issue of Transformers, showcase universality by performing well across most tasks, and remain simple without unnecessary hard-coding or engineering complexities. There should be no compromise between speed and memory efficiency, and they should be able to seamlessly integrate with TPUs and accommodate causal masking.

The research conducted in this PhD thesis is in line with the continuous efforts to model long texts efficiently. Chapter 5 contributes to research on efficient long-range modeling by proposing a sparse attention pattern based on the layout of documents, introducing a novel approach to alleviate the computational

burden associated with long-range modeling. Acknowledging the importance of document structures in guiding summary generation, Chapter 7 integrates layout information into existing long-range models (Zaheer et al. 2020), thereby improving summarization of long documents and highlighting the significance of layout information in capturing long-range dependencies.

3.4 Conclusion

Transformer-based Pretrained Language Models have brought significant breakthroughs to NLP (Devlin et al. 2018; M. Lewis et al. 2019; T. Brown et al. 2020). Despite remarkable achievements, the quadratic complexity of the Transformer hinders their effective application to long texts. In tasks involving long texts, many works tend to apply the same approaches used for processing shorter ones (*e.g.* truncating or chunking) without addressing the challenges specific to long texts (*e.g.*, long-range dependencies and complex hierarchical structures). While various Transformer variants have been proposed to efficiently model long texts, there is currently no variant that can be confidently identified as having effectively addressed the efficiency challenges in Transformers while preserving performance comparable to that obtained with full self-attention.

The next chapter will address the applicative scope of this PhD thesis: Document Understanding. Notably, the prevailing approach in solving document understanding tasks involves leveraging Pre-trained Language Models to jointly learn text and multimodal information.

DOCUMENT UNDERSTANDING

Contents

| | | |
|-------|---|-----------|
| 4.1 | Document Understanding Tasks and Datasets | 61 |
| 4.1.1 | Document Layout Analysis | 62 |
| 4.1.2 | Document Image Classification | 63 |
| 4.1.3 | Visual Information Extraction | 63 |
| 4.1.4 | Document Visual Question Answering | 65 |
| 4.1.5 | Towards Advancing Document Understanding | 65 |
| 4.2 | Task-specific Deep Learning Models For Document Understanding | 65 |
| 4.2.1 | Document Image Classification | 66 |
| 4.2.2 | Document Layout Analysis | 66 |
| 4.2.3 | Visual Information Extraction | 67 |
| 4.3 | Deep Fusion of Modalities via General-purpose Multimodal Pre-training . . . | 68 |
| 4.3.1 | Representing Layout-rich Documents | 70 |
| 4.3.2 | Encoding Visual Elements | 76 |
| 4.3.3 | Pre-training Multimodal Transformers for Document Understanding . | 77 |
| 4.3.4 | Extending Layout/Visually-aware Transformers | 81 |
| 4.4 | Conclusion | 82 |

The majority of models, benchmarks, and tasks focus exclusively on a single source of information, namely plain text. However, disregarding the visual appearance of text is suboptimal in real-world scenarios. Documents, ranging from webpages to digital-born PDF files and scanned document images, exhibit a diverse array of formats. These documents, including business forms, scholarly and news articles, invoices, financial reports and emails, convey information not only through language but also through *visual* content (*e.g.*, figures, text formatting) and *layout* structure (*i.e.*, text positioning). Extracting information from documents becomes a challenging task due to the diversity in layouts and formats, the presence of low-quality scanned document images, and the complexity of template structures. Manually extracting information is a time-consuming and error-prone process with limited reusability. As such, Document Understanding, *i.e.*, the process of automatically understanding, classifying and extracting information from

visually-rich documents, has emerged as a key research area. As the foundation of digital transformation, Document Understanding holds significant economic value and has experienced a surge in industrial demand in recent years. To reduce the time and cost of document workflows, more and more companies are shifting from labor-intensive, rule-based algorithms to deep learning-based entity recognition, document classification, semantic extraction, *etc.* The study of Document Understanding spans multiple disciplines and involves developing models and techniques to comprehend the content and structure of complex, visually-rich documents. As such, Document Understanding is also crucial from an academic standpoint as it opens up avenues for various research and advancements in Natural Language Processing (NLP) and Computer Vision.

Over the last thirty years, the development of Document Understanding has undergone various stages—starting from rule-based heuristics to the rise of neural network approaches. In the early 1990s, researchers relied on rule-based heuristic methods constructed by manually observing document layout information (Wong et al. 1982; Fisher et al. 1990; Lebourgeois et al. 1992). However, these handcrafted rules proved to be non-scalable, and the adoption of rule-based approaches often resulted in high labor costs. As Machine Learning technology rose in the 2000s, models based on annotated data (Baechler and Ingold 2011; H. Wei et al. 2013) became the predominant approach for document processing, marking a transition towards more scalable and data-driven approaches in Document Understanding. While offering a certain degree of performance enhancement, their general usability often falls short due to the lack of customized rules and limited training samples. Moreover, the adaptation costs for various document types are relatively high, rendering previous approaches impractical for widespread commercial use. In recent years, the advent of Deep Learning and the accumulation of vast amounts of unlabeled electronic documents, have propelled Document Understanding into a new era. This era embraces the "pre-training then fine-tuning" paradigm, resulting in a significant breakthrough in the field (Yiheng Xu et al. 2020; Peng et al. 2022).

In this chapter, we begin by providing an overview of the representative tasks and datasets in Document Understanding. We then explore the most recent and significant advancements in the field driven by Deep Learning, focusing on two research directions. The first direction involves task-specific approaches that employ shallow fusion between textual and visual/layout information. On the other hand, the second axis explores the application of pre-training techniques for deep fusion of modalities, significantly advancing analysis performance and accuracy.

4.1 Document Understanding Tasks and Datasets

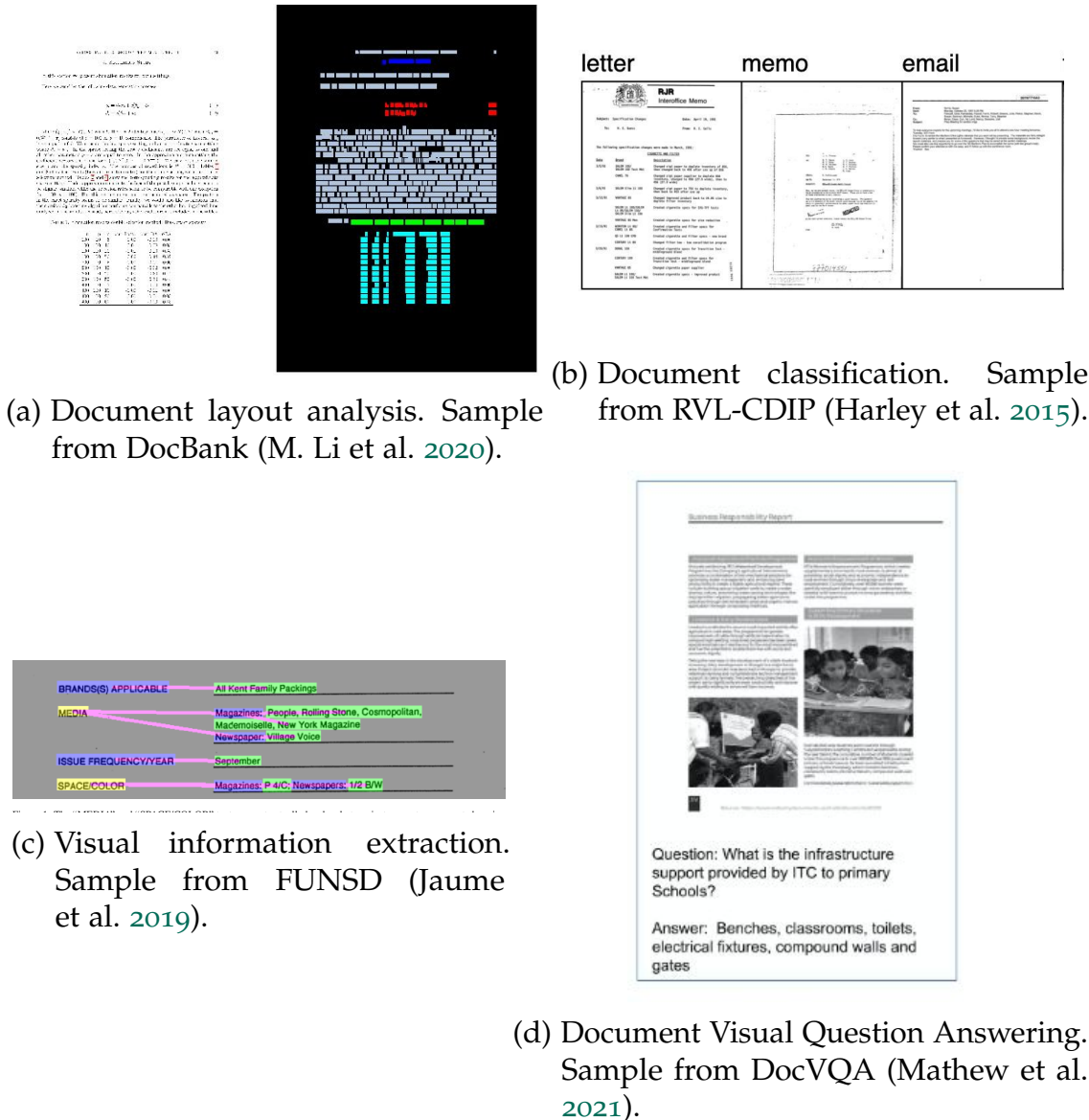


Figure 4.1 – Document Understanding tasks and examples.

The field of Document Understanding covers problems that involve understanding visually-rich documents (in contrast to plain texts), requiring comprehending the conveyed multimodal information. Real-world application scenarios encompass a diverse set of tasks spanning various domains and paradigms. These tasks can be broadly classified into four categories:

- Document Layout Analysis, which involves identifying and categorizing document components;
- Document Image Classification, which consists in classifying document images into various categories;
- Visual Information Extraction, which requires extracting semantic entities and their relationships from visually-rich documents;
- Document Visual Question Answering, which consists in supplying the answer to a question related to a visually-rich document.

While document layout analysis and document image classification are more centered around processing visual data (*image-centric*), visual information extraction and document visual question answering put more focus on textual data (*text-centric*).

4.1.1 Document Layout Analysis

Document layout analysis consists in automatically locating and categorizing the components (*e.g.*, text, tables, figures) of a document. This task includes two primary subtasks: *page segmentation* and *logical structural analysis* (Binmakhashen and Mahmoud 2019). Page segmentation consists in detecting the structure of the document and establishing a partition into distinct regions such as text, figures, images, and tables. On the other hand, logical structural analysis focuses on providing finer-grained semantic classifications within the previously detected regions, *e.g.* identifying a region of text that is a paragraph. Document layout analysis plays a crucial role in parsing semi-structured documents into structured, machine-readable formats for downstream applications, such as Optical Character Recognition (OCR). This task is challenging due to the varying layouts and formats of the documents. Several benchmark datasets have emerged for document layout analysis. The International Conference on Document Analysis and Recognition (ICDAR) have produced several gold-standard datasets from their annual competitions (Antonacopoulos et al. 2013; L. Gao et al. 2017). PubLayNet (Zhong et al. 2019) consists of biomedical and life sciences articles and involves detecting and classifying page regions into categories including caption, list, and paragraph. DocBank (M. Li et al. 2020) comprises scientific research papers with fine-grained token-level annotations for better applicability to NLP methods.

Table understanding is a crucial and challenging subtask of document layout analysis. Tables serve as a concise and effective means of summarizing and presenting information in various types of documents. Unlike other document elements

such as headings and paragraphs, tables display greater variability in format and a more complex structure. Consequently, significant research has been conducted on tables, focusing on two key subtasks: 1) Table detection, which aims to determine the boundaries of tables in the document, and 2) Table structure recognition, where the objective is to extract the layout structure of tables, including information about rows, columns, and cells. ICDAR held several competitions to evaluate both aspects of table understanding, using both modern and archival documents (Göbel et al. 2013; L. Gao et al. 2019). To address the need for larger datasets, M. Li et al. (2019) proposed TableBank, a large-scale dataset built from Office Word and \LaTeX documents using weak supervision. With PubTabNet, Zhong et al. (2020) offer additional information on table structure and cell content to assist in table recognition.

4.1.2 Document Image Classification

Document image classification refers to the process of classifying document images into various categories such as emails, invoices, scientific papers, and more. Unlike natural images, document images primarily consist of textual content displayed in diverse styles and layouts. Therefore, Document Image Classification is a special subtask of image classification that requires understanding both visual and textual aspects of documents. The Tobacco-3482 dataset (Kumar and Doermann 2013) consists of 3,482 document images classified into 10 classes. Widely used for document image classification, the large-scale RVL-CDIP dataset (Harley et al. 2015) is a representative dataset for this task and encompasses 400,000 images distributed across 16 categories.

4.1.3 Visual Information Extraction

Visual information extraction consists in extracting semantic *entities* (*entity recognition*) and their relationships (*relation extraction*) from visually-rich documents, based on a set of pre-defined *keys*. In contrast to traditional text-only information extraction, the two-dimensional spatial arrangement of text requires an understanding of page layouts for complete comprehension.

Notable public datasets for this task include Form Understanding in Noisy Scanned Documents (FUNSD) (Jaume et al. 2019), a form understanding dataset consisting of 199 real, noisy, and fully annotated scanned forms. These forms are organized as a list of interlinked form entities. Each semantic entity is characterized by a unique identifier, an entity type (*i.e.*, *key*), a list of links with other entities, and a list of words and their corresponding bounding boxes. The dataset

features three keys for which values need to be extracted: question, answer, and header. For receipt understanding, Scanned Receipts OCR And Key Information Extraction (SROIE) (Zheng Huang et al. 2019) (973 English documents) and Consolidated Receipt Dataset (CORD) (Park et al. 2019) (1000 Indonesian documents) are widely used datasets. In SROIE, every receipt is formatted as a list of text lines, with each line being associated with word bounding boxes. The receipts are classified into four entity types: company, date, address, and total. In CORD, receipts are arranged as lists of entities, each identified by a unique identifier, an entity type, and a list of words along with their respective bounding boxes. CORD encompasses 30 entity types, further categorized into four superclasses: menu, subtotal, total, and void. In Chapter 6, we employ these datasets to evaluate our work for visual information extraction tasks.

Several datasets have been proposed to advance entity extraction from more complex documents. Kleister (Graliński et al. 2020) comprises non-disclosure agreements and financial reports, typically characterized by their extensive length. The DeepForm dataset provided by Borchmann et al. (2021) is an improved version of the original dataset¹ featuring advertising disclosure forms with manually corrected invalid data points. Additionally, the PWC dataset, also proposed by Borchmann et al. (2021), reformulates the original PWC Leaderboards dataset (Kardas et al. 2020) into a visual information extraction task, using digital-born scientific papers as input instead of tables. VRDU-Registration Forms and VRDU-Ad-buy Forms (Zilong Wang et al. 2023) contain public documents downloaded from the Foreign Agents Registration Act² and the Federal Communications Commission,³ respectively. Three settings of increasing difficulty have been introduced to evaluate the model’s performance across diverse scenarios. These scenarios range from handling a single template across training, validation, and testing, to the conventional setting where all three sets contain documents from the same template set, and finally, to the challenging scenario where documents in the sets are drawn from distinct template sets.

To prompt further research into multilingual document understanding systems, datasets for visual information extraction containing documents in languages other than English have been proposed. EPHOIE (Jiapeng Wang et al. 2021) contains Chinese document images collected from scanned examination papers. XFUND is a multilingual extension of FUNSD (Yiheng Xu et al. 2022) and contains manually-labeled forms in 7 languages.

1. https://wandb.ai/stacey/deepform_v1
2. <https://www.justice.gov/nsd-fara>
3. <https://publicfiles.fcc.gov/>

4.1.4 Document Visual Question Answering

Document visual question answering is a high-level understanding task that requires providing the correct answer to a question related to a visually-rich document, posed in natural language. This is achieved by jointly reasoning over the document layout (page structure, forms, tables), textual content (handwritten or typewritten), and visual elements (marks, tick boxes, diagrams). Unlike traditional visual question answering tasks, textual information holds a pivotal role in document visual question answering. The DocVQA dataset (Mathew et al. 2021) contains more than 12,000 industry documents and 5,000 corresponding questions. InfographicsVQA (Mathew et al. 2022) comprises infographic images with questions that require elementary reasoning and basic arithmetic skills. To spur further research on developing generation abilities, VisualMRC (Tanaka et al. 2021) was built from multiple domains of webpages and requires producing long and abstractive answers. For question answering over tables, Borchmann et al. (2021) converted the WikiTableQuestions dataset (Pasupat and Liang 2015) into a document visual question answering dataset by using HTML tables sourced from Wikipedia as input, in place of the original semi-structured HTML tables.

4.1.5 Towards Advancing Document Understanding

The number of open-sourced benchmark datasets proposed for these tasks has significantly facilitated the development of new techniques and models. Particularly noteworthy is the recent surge in deep learning-based models that have achieved state-of-the-art performance across these tasks. In line with the aforementioned works, we introduce novel datasets for layout-aware summarization—a task that has received limited attention in the document understanding field—in Chapter 7.

4.2 Task-specific Deep Learning Models For Document Understanding

Deep Learning marked a paradigm shift by enabling significant performance leaps across various research areas. In particular, the fields of NLP and Computer Vision have undergone substantial advancements due to the emergence of Deep Learning. The development of Document Understanding also reflects a similar

trend, where methodologies from [NLP](#) and Computer Vision are integrated into modern document understanding systems.

The initial approach to enhancing document understanding system with Deep Learning involves the use of task-specific methods. These methods leverage pre-trained Computer Vision and/or [NLP](#) models to gather knowledge from the corresponding modality. Features are extracted from either a single modality or through a simple combination of modality features, and used for a specific downstream task.

4.2.1 Document Image Classification

Document image classification constitutes a subtask of image classification. Consequently, classification models originally designed for natural images can be applied to tackle the challenges associated with document image classification. Afzal et al. (2015) used a deep Convolutional Neural Network ([CNN](#)) trained on the millions of examples from ImageNet (Deng et al. 2009). Das et al. (2018) introduced a range of deep [CNNs](#) designed to classify specific regions within a document. These classifiers are combined using an effective ensembling technique for document image classification. Dauphinee et al. (2019) developed a model consisting of two distinct components — one dedicated to processing text and the other to handling images. This modular approach uses both visual information and textual content within a page to classify document images.

4.2.2 Document Layout Analysis

Document layout analysis can be framed as an instance segmentation task for document images, where units such as text, figures, headings, paragraphs, and captions are the objects that need to be detected and recognized. For this task, the model has to predict per-pixel labels to categorize regions of interest within the document. Framing document layout analysis as an instance segmentation task offers flexibility and can adapt to both the coarser-grained task of page segmentation and the finer-grained task of logical structural analysis. Most deep learning based document layout analysis borrow various [CNN](#)-based object detection and segmentation frameworks Ren et al. (2015) and Cai and Vasconcelos (2018). X. Yang et al. (2017) proposed an end-to-end [CNN](#) that combines both text and visual features within an encoder-decoder architecture for pixel classification. The encoder-decoder architecture ensures that visual feature information at various levels of resolution is considered throughout the encoding and decoding process (Burt and Adelson 1987). In addition to the resulting visual representation, text

embeddings learned from a pre-trained NLP model are supplied at the final decoding layer. Similarly, Oliveira et al. (2018) introduced a multi-task pixel-by-pixel prediction CNN-based model to perform layout analysis on historical documents. Soto and Yoo (2019) view layout analysis of scientific articles as an object detection task. While instance segmentation requires delineating the boundaries of individual instances at the pixel level, object detection primarily focuses on identifying and locating objects using bounding boxes. Soto and Yoo (2019) find that integrating contextual information into Faster R-CNN (Ren et al. 2015), a prominent object detection framework, leverages local invariance of article elements and improves region detection performance.

Faster R-CNN has been particularly successful when directly applied to table detection, a task often framed as an object detection task. CascadeTabNet (Prasad et al. 2020) leverages the Cascade R-CNN (Cai and Vasconcelos 2018) model for simultaneous table detection and table structure recognition. Another notable contribution is TableSense (H. Dong et al. 2019), which significantly enhances table detection by introducing cell features and employing advanced sampling algorithms.

Most deep learning based document layout analysis and table understanding models mainly focus on processing visual features of layout components using CNN-based models. Visual features alone may fall short in solving document layout analysis tasks due to their limited ability to capture underlying semantics, structural relationships, and textual content. For better layout analysis, it is crucial to integrate both visual and semantic information while capturing and encoding relationships between layout components (S. Luo et al. 2022). We explore this direction in Chapter 5.

4.2.3 Visual Information Extraction

For information extraction from visually-rich documents, many researchers and practitioners have framed the problem as a segmentation task. In this approach, semantically meaningful regions are identified through object detection and labeled via semantic segmentation. Given the pivotal role of text in visual information extraction, directly embedding textual information into the image simplifies handling of 2D textual relationships. The conventional framework considers document images as a pixel grid. Each pixel corresponds to a text embedding vector, which can be a one-hot encoding vector (Katti et al. 2018), a word embedding (Kerroumi et al. 2021), or a contextualized subword embedding (Denk and Reisswig 2019). This approach preserves the layout of documents and captures details such as positioning, size, and alignment for textual components. As an initial attempt, Chargrid (Katti et al. 2018) operates at the character level, using a one-hot encod-

ing for each character. A 2D grid of characters is constructed by mapping each pixel intersecting with a character bounding box to the corresponding one-hot encoding. A convolution-based encoder-decoder model is then applied to perform instance-level segmentation. Specifically, the model predicts a segmentation mask where each pixel is assigned a class label, and generates object bounding boxes to assign characters from the same segmentation class to distinct instances. Expanding on Chargrid, VisualWordgrid (Kerroumi et al. 2021) operates at the word level, incorporating word embeddings from Word2Vec or fastText. To improve the end-to-end accuracy, BERTgrid (Denk and Reisswig 2019) constructs a grid at the word-piece level, embedding each word piece with dense contextualized vectors obtained from Bidirectional Encoder Representations from Transformers (BERT). ViBERTgrid (W. Lin et al. 2021) takes a step further by concatenating a BERTgrid to an intermediate layer of a CNN model, resulting in a more powerful grid-based document representation.

Documents can also be seen as graphs, wherein nodes correspond to textual segments, and relationships between text fragments are modeled as edges. Xiaojing Liu et al. (2019) introduce a model based on Graph Convolutional Networks (GCNs) to integrate both textual and visual information. In this model, each node comprises information about the position of the segment and the text it contains, while edges represent the relative distances between the corresponding segments and their aspect ratio. Graph convolution is employed to calculate graph embeddings for each text segment, which are then combined with text embeddings. The resulting embeddings are fed into a bidirectional Long Short Term Memory (LSTM) for information extraction from in-house invoices and receipts. This graph-based approach ensures that both local and global information can be learned. Hwang et al. (2020) model a document as a directed graph, extracting information through dependency analysis. W. Yu et al. (2021) combines graph learning with graph convolution to achieve richer semantic representations.

In Chapter 6, we tackle visual information extraction tasks through a distinct approach, leveraging general-purpose multimodal pre-training methods for a broader and more versatile perspective.

4.3 Deep Fusion of Modalities via General-purpose Multimodal Pre-training

While the aforementioned methods demonstrate good performance across document understanding tasks, they still face significant limitations. The majority of these models are all designed for specific tasks and document types. As such, these approaches rely on labeled data; however, most datasets related to Docu-

ment Understanding are label-scarce. This scarcity arises from the labor-intensive and time-consuming nature of the human annotation process. Driven by data limitations, these task-specific models often rely solely on pre-trained Computer vision models and/or NLP models, each trained independently. A common practice involves combining the knowledge gained from each modality through a shallow fusion of features, such as concatenation. However, this approach makes it challenging to easily transfer domain knowledge from one document type to another. This stems from the need to re-train models from scratch when the document type is changed.

Visually-rich documents encompass three modalities that inherently align: text, layout, and visual information. Layout, *i.e.*, the spatial relationship of text blocks within a document, facilitates reading comprehension and information searching (Meyer et al. 1980; Guthrie et al. 1991; Wright 1999). For instance, the arrangement of key-value pairs in forms typically follows a left-right or top-down order. In addition to spatial information, the visual elements presented with the text can offer global structural information (*e.g.* there is a clear visual distinction between different document types) and help with downstream tasks (*e.g.*, the title of documents is usually enlarged). Hence, it becomes vital to jointly learn text with layout and visual information.

As visually-rich documents are widely used in real-world applications, there is a substantial volume of unlabeled documents. This provides an opportunity for leveraging self-supervised pre-training methods. The widespread success and popularity of pre-training techniques, notably those employing the Transformer architecture (Vaswani et al. 2017), emphasize the critical role of deep contextualization for sequence modeling in both NLP and Computer Vision problems. Following the current research trend, a general-purpose framework that can learn from unlabeled documents through pre-training and perform model fine-tuning for different types of downstream applications is preferred over ones that are task-specific and require fully-annotated training data. This trend has prompted a shift in Document Understanding towards the "pre-training then fine-tuning" paradigm, establishing pre-training techniques as the *de facto* approach in the field over years. In particular, researchers and practitioners have been leveraging the Transformer architecture to attain cross-modal alignment via joint pre-training of text, layout, and images from large amounts of unlabeled data. This approach enables pre-trained models to absorb cross-modal knowledge across various document types. Consequently, when the model is applied to a different domain with different document formats, only a small number of labeled samples are required to fine-tune the generalized model effectively.

Next, we discuss general-purpose, multimodal pre-training methods for Document Understanding. We review these methods from several viewpoints, consid-

ering the various challenges in the field. This includes the integration of layout, image encoding, the pre-training tasks used, the incorporation of positional information, model initialization, and considerations for multilingual and long-range aspects.

The models discussed are summarized in Table 4.1. Through pre-training, these models have learned general representations that can be fine-tuned for a wide range of tasks. Notably, these pre-trained models demonstrate remarkable performance, surpassing state-of-the-art task-specific models across document image classification (Yang Xu et al. 2020), information extraction (Peng et al. 2022), and document layout analysis (M. Li et al. 2020) tasks. In parallel with the development of datasets for document visual question answering, the use of self-supervised pre-training methods has yielded notable achievements in the corresponding tasks (Appalaraju et al. 2021; Tanaka et al. 2021).

| | Architecture | Visual Encoding | Bounding Box Granularity | Layout Encoding | Relative Bias |
|------------------------------------|----------------------|--------------------|--------------------------|----------------------|---------------|
| LayoutLM (Yiheng Xu et al. 2020) | Encoder | ✗ | Word | Embedding Tables | ✗ |
| LayoutLMv2 (Yang Xu et al. 2020) | Encoder | ResNeXt-FPN | Word | Embedding Tables | ✓ |
| LayoutXLM (Yiheng Xu et al. 2022) | Encoder | ResNeXt-FPN | Word | Embedding Tables | ✓ |
| LayoutLMv3 (Y. Huang et al. 2022) | Encoder | Embedding Tables | Block | Embedding Tables | ✓ |
| Pramanik et al. (2020) | Encoder | ResNet50-FPN | Word | Embedding Tables | ✗ |
| DocFormer (Appalaraju et al. 2021) | Encoder | ResNet50 | Word | Embedding Tables | ✗ |
| ERNIE-Layout (Peng et al. 2022) | Encoder | ResNeXt-FPN | Word | Embedding Tables | ✓ |
| FormNet (C.-Y. Lee et al. 2022) | Encoder | ✗ | Word | ✗ | ✓ |
| BROS (Hong et al. 2020) | Encoder | ✗ | Word | Sinusoidal functions | ✗ |
| StructuralLM (C. Li et al. 2021) | Encoder | ✗ | Block | Embedding Tables | ✗ |
| SelfDoc (P. Li et al. 2021) | Dual-stream encoder | Faster R-CNN | Block | ✗ | ✗ |
| LiLT (Jiapeng Wang et al. 2022) | Dual-stream encoder | ✗ | Word | Embedding Tables | ✗ |
| H-VILA (Z. Shen et al. 2022) | Hierarchical encoder | ✗ | Block | Embedding Tables | ✗ |
| LAMPReT (T.-L. Wu et al. 2021) | Hierarchical encoder | CNN + Linear Layer | Block | ✗ | ✗ |
| TILT (Powalski et al. 2021) | Encoder-decoder | U-Net | Word | ✗ | ✓ |
| LayoutT5 (Tanaka et al. 2021) | Encoder-decoder | Faster R-CNN | Word | Embedding Tables | ✗ |
| LayoutBART (Tanaka et al. 2021) | Encoder-decoder | Faster R-CNN | Word | Embedding Tables | ✗ |
| Donut (Kim et al. 2022) | Encoder-decoder | Swin Transformer | ✗ | ✗ | ✗ |
| UDOP (Z. Tang et al. 2023) | Encoder-decoder | Embedding Tables | Word | Embedding Tables | ✓ |

Table 4.1 – Summary of general-purpose, multimodal pre-training document understanding models.

4.3.1 Representing Layout-rich Documents

The layout of a document defines its visual structure. Effectively incorporating layout information provides valuable cues for accurate interpretation and organization of information, addressing the challenges posed by complex and varied document layouts in document understanding tasks. We explore strategies for incorporating layout information into Transformer models, delving into layout

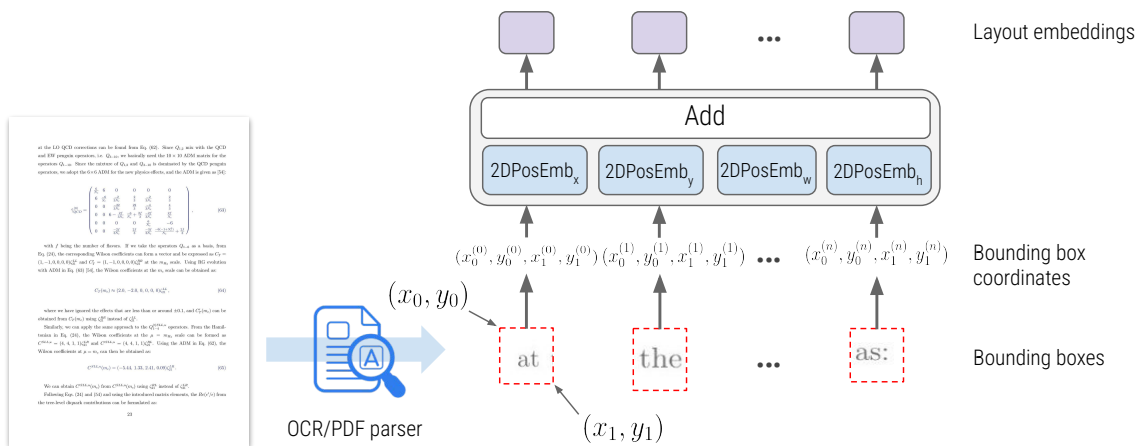


Figure 4.2 – Layout encoding process in LayoutLM (Yiheng Xu et al. 2020).

encodings and layout-aware model architectures. Then, we focus on modifications to the attention mechanism that explicitly include layout information. Finally, we examine the role of position encodings in the context of document understanding tasks, where complex layouts and potential reading order inaccuracies pose significant challenges.

4.3.1.1 Incorporating Layout Information

As the first work to jointly learn text and layout information, LayoutLM (Yiheng Xu et al. 2020) stands out as the pioneer work in multimodal pre-training for document understanding tasks. Over time, it has become the building block for designing more complex document understanding systems. LayoutLM encodes layout information with learned 2D position embeddings, as illustrated in Figure 4.2. A 2D position embedding carries information about the spatial position of a token within the document page. The spatial position of a token is represented by its bounding box (x_0, y_0, x_1, y_1) obtained by an OCR engine (Kay 2007), where (x_0, y_0) and (x_1, y_1) respectively denote the upper-left and lower-right corners. The coordinates are discretized and normalized to integers in $\{0, \dots, 1000\}$. Four embedding tables are used to encode spatial positions: two for the coordinates axes (x and y) and the other two for the bounding box size (width and height). The final layout embedding $\ell \in \mathbb{R}^{d_\ell}$, for a token located at position (x_0, y_0, x_1, y_1) , is defined by:

$$\begin{aligned}
\ell = & 2\text{DPosEmb}_x(x_0) + 2\text{DPosEmb}_y(y_0) \\
& + 2\text{DPosEmb}_x(x_1) + 2\text{DPosEmb}_y(y_1) \\
& + 2\text{DPosEmb}_w(x_1 - x_0) \\
& + 2\text{DPosEmb}_h(y_1 - y_0)
\end{aligned} \tag{4.1}$$

These 2D position embeddings are added to the sequential position and text embeddings from [BERT](#). The resulting input sequence is passed through an encoder similar to [BERT](#). Via the self-attention mechanism, encoding 2D position features into the language model improves alignment between layout information and semantic representation.

Building on the groundwork laid by LayoutLM, many works have introduced alternative approaches to encode layout information. BROS (Hong et al. 2020) employs sinusoidal functions as an alternative to linear embedding layers to encode continuous values for the spatial positions of tokens on the page. In addition to spatial positions, DocFormer (Appalaraju et al. 2021) incorporates the Euclidean distance from each corner of a bounding box to the corresponding corner in the bounding box to its right, as well as the distance between centroids of the bounding boxes. DocFormer, as well as ERNIE-Layout (Peng et al. 2022), create separate layout features for visual and language modalities. Driven by the consideration that 2D dependencies might differ across layers, DocFormer introduces layout features as residual connections to each layer of a Transformer encoder, as opposed to directly adding them to language features. In addition, layout features are shared across modalities to enforce feature correlation across modalities.

Based on the assumption that words in a block typically convey the same semantic meaning, certain studies favor leveraging information from content blocks (such as header, paragraph, figure, detected with segmentation algorithms from Section 4.2.2) rather than operating at the word level, as contextualization between every word may be redundant and overlook localized context. StructuralLM (C. Li et al. 2021) departs from word-level 2D positions and leverages block-level 2D positions derived from the bounding boxes obtained through OCR. As such, words that belong to the same block share the same 2D position embeddings. This approach allows the model to discern which words belong to the same block, thereby enhancing contextual representations of cells. Similarly, LayoutLMv3 (Y. Huang et al. 2022) models the layout information of blocks by adopting block-level 2D positions.

Rather than treating layout information as an additional feature, some works have modified model architectures to align with the inherent structure of docu-

ments. In Layout-Aware Multimodal PreTraining (LAMPReT) (T.-L. Wu et al. 2021), the layout is obtained by parsing a document into content blocks, which encompass text and possibly images, using PDF parsing tools. The layout is then processed by two cascaded Transformers. The lower-level Transformer processes content blocks as input, while the higher-level model aggregates the block-level representations obtained by the lower-level Transformer, focusing on how these blocks are spatially structured. SelfDoc (P. Li et al. 2021) uses the visual features of content blocks obtained with an object detection model, Faster R-CNN (Ren et al. 2015), trained on semantically meaningful components such as text blocks, titles, lists, tables, and figures. H-VILA (Z. Shen et al. 2022) parses the document to extract group of tokens, which are then encoded using a hierarchical Transformer. For visual information extraction tasks, Token Path Prediction (Chong Zhang et al. 2023) models the document layout as a complete directed graph of tokens. UDOP (Z. Tang et al. 2023) fuses image pixels and text tokens based on layout information, *i.e.*, the input representation of a token is the sum of its text representation and the image feature of the patch to which it belongs. Our approach in Chapter 5 is in line with the concept of adapting model architectures to suit the inherent structure of documents. Building on prior works in cognitive sciences, we introduce a novel attention mechanism that relies solely on the spatial positions of tokens in the page. This approach mirrors the human process of skimming through a document to extract its inherent structure.

4.3.1.2 Layout-aware Attention Mechanisms

The relative order of two tokens, how many tokens separate them, or how many pixels apart they are, are often relevant to the decision of how strongly a token should attend to another one. However, 2D position embeddings can only implicitly capture the relationship between tokens within a document. To address this limitation and effectively model local invariance in document layout, various research works have introduced advanced approaches to integrate layout information by incorporating the spatial relationships between tokens directly into the attention calculation process.

Extending the concept of 1D relative position bias (Raffel et al. 2020) to the 2D scenario, LayoutLMv2 (Yang Xu et al. 2020) builds upon LayoutLM by incorporating *bias* terms that encode the 2D relative position of tokens with respect to each other. These bias terms are added to the attention scores to explicitly capture the relationship between tokens, defining a *spatial-aware attention mechanism*. Formally, the pre-softmax attention scores are defined as follows:

$$\alpha_{i,j} = \frac{1}{\sqrt{d}} \mathbf{q}_i \cdot \mathbf{k}_j + \mathbf{b}_{j-i}^{(1D)} + \mathbf{b}_{x_j-x_i}^{(2D_x)} + \mathbf{b}_{y_j-y_i}^{(2D_y)}, \quad (4.2)$$

where $\mathbf{b}^{(1D)}$, $\mathbf{b}^{(2D_x)}$, and $\mathbf{b}^{(2D_y)}$ correspond to the sequential, horizontal, and vertical relative position biases, respectively. Likewise, TILT (Powalski et al. 2021) incorporates 2D relative positions while entirely discarding the use of absolute 2D positions in its encoding strategy.

To introduce a spatial perspective to the computation of semantic similarity within attention, both DocFormer (Appalaraju et al. 2021) and ERNIE-Layout (Peng et al. 2022) decouple attention into distinct components. ERNIE-Layout uses a disentangled attention mechanism (P. He et al. 2020). In this mechanism, each token is represented using four sets of query-key-value, with each set encoding the token’s content (*i.e.*, text and image), relative sequential positions (1D), relative horizontal positions ($2D_x$), or relative vertical positions ($2D_y$). Attention⁴ of token i on token j is computed as the sum of standard (*i.e.*, content-based) attention and the following components:

$$\alpha_{ij}^* = \mathbf{q}_i \cdot \mathbf{k}_{\delta_*(i,j)}^*{}^\top + \mathbf{k}_j \cdot \mathbf{q}_{\delta_*(j,i)}^*{}^\top, \quad \forall \star \in \{1D, 2D_x, 2D_y\} \quad (4.3)$$

where relative positions $\delta_*(i, j)$ are first mapped to relative position embeddings using embedding tables, then projected into query, key, and value vectors. Rather than using query-key-value projections, DocFormer directly incorporate relative position embeddings $\mathbf{b}^{(1D)}$ into its attention mechanism. Dismissing relative 2D positions, DocFormer defines attention as the sum of standard attention with 1D relative attention and layout-based attention, expressed as follows:

$$\begin{aligned} \alpha_{ij}^{(1D)} &= \mathbf{q}_i \cdot \mathbf{b}_{j-i}^{(1D)} + \mathbf{q}_j \cdot \mathbf{b}_{j-i}^{(1D)} \\ \alpha_{ij}^{(2D)} &= \mathbf{q}_i^{(2D)} \cdot \mathbf{k}_j^{(2D)}, \end{aligned} \quad (4.4)$$

where $\mathbf{q}_i^{(2D)}$ and $\mathbf{k}_j^{(2D)}$ are computed from the tokens’ layout embeddings.

In Language-independent Layout Transformer (LiLT) (Jiapeng Wang et al. 2022), text embeddings and layout embeddings are processed independently through separate encoders. To consider the cross-modal interactions between text and layout across the entire pipeline, a *bidirectional attention complementation mechanism* is introduced. In each layer of the textual encoder, the attention scores are summed with those obtained by the layout encoder at the same layer level, and vice-versa.

Rather than using relative embeddings, FormNet (C.-Y. Lee et al. 2022) incorporate features about relative positions by using trainable parametric functions. For each feature type (*e.g.*, the order of and log-distance between pairs of tokens), the observed feature is compared against the "ideal" feature, which is obtained using the associated parametric function. The resulting penalty score is then

4. For the sake of clarity, we omit the scaling and softmax operations.

added to the attention score. This process penalizes pairs whose predicted feature significantly deviates from the observed one.

4.3.1.3 Incorporating Sequential Position Information

All document pre-training techniques operate on serialized text. An OCR engine or PDF parser extracts text from a document and serializes it according to a raster-scan order, which arranges tokens in a sequence from the top-left to the bottom-right corner. However, this arrangement does not always conform to human reading patterns, particularly for documents with complex layouts such as multicolumn texts, tables, and forms. This misalignment with human reading habits (*i.e.*, *serialization errors*) can result in suboptimal performance in document understanding tasks.

To address this problem, automatic word reordering techniques can be employed. ERNIE-Layout (Peng et al. 2022) uses an in-house document layout analysis toolkit that provides an appropriate reading order based on the spatial distribution of words, pictures, and tables. Enhanced with this knowledge, the token sequence can be rearranged in a way that yields a lower perplexity compared to the raster-scan order. This translates into a serialization that aligns better with human reading patterns. LayoutReader (Zilong Wang et al. 2021) was designed to detect the reading order of documents and improve the text line ordering of OCR engines. This sequence-to-sequence model employs LayoutLM (Yiheng Xu et al. 2020) as its encoder, generating the reading order sequence for documents. Another strategy to mitigate reading order errors involves using more robust position encodings, as it has been shown that the performance of document Transformer models with learned positional embeddings significantly deteriorates on noisy data with incorrect reading order information (Hong et al. 2020). G. Wang et al. (2022) propose a Learnable Sinusoidal Positional Encoding (LSPE) method based on feed-forward networks. Using sinusoidal functions enables the model to extrapolate to longer lengths not encountered during training. Simultaneously, the learnable feed-forward network component enhances the learnability and flexibility of positional representation, particularly for spatial information. LSPE can be integrated into any Transformer-based model and demonstrates improved performance and robustness on noisy data with unreliable reading order information. XYLayoutLM (Gu et al. 2022) leverages both strategies by introducing 1) an augmentation algorithm based on XY Cut (Ha et al. 1995) to generate a series of proper reading orders for training, and 2) a Dilated Conditional Position Encoding as the position embedding generator to create position embeddings of varying lengths with local layout information. Another strategy to mitigate serialization errors involves graph construction. Prior to serialization, FormNet (C.-Y. Lee et al. 2022) leverages inductive biases about the spatial relationships

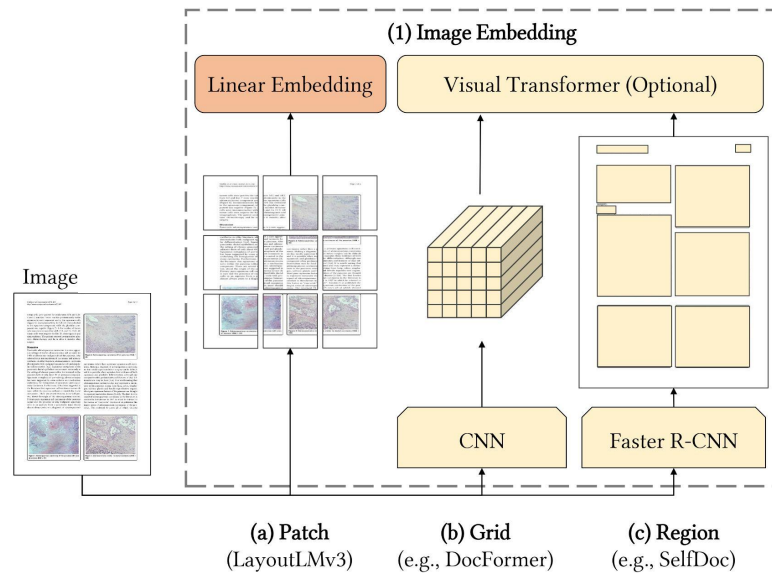


Figure 4.3 – Comparisons of LayoutLMv3 (Y. Huang et al. 2022), DocFormer (Appalaraju et al. 2021) and SelfDoc (P. Li et al. 2021) on image embedding construction. Adapted from Y. Huang et al. (2022).

between tokens to build a graph. For each token in the document, it then creates a contextualized Super-Token by embedding representations from its neighboring tokens through graph convolutions.

However, all the models presented still depend on a reading order, which might introduce noise and may not be relevant in the case of complex layouts. Our work in Chapter 6 avoids serialization errors by entirely discarding sequential position information. Instead, position embeddings are created from the document layout by learning to reconstruct the relative order between tokens.

4.3.2 Encoding Visual Elements

To capture appearance features, many research works have leveraged vision models to integrate visual elements into document understanding systems.

In the fine-tuning stage of LayoutLM (Yiheng Xu et al. 2020), optional token visual embeddings can be added to capture appearance features, *e.g.*, fonts, types, colors. Token visual embeddings are obtained by splitting the document image according to the bounding boxes obtained through OCR, and feeding the resulting pieces to Faster-RCNN (Ren et al. 2015). LayoutLMv2 (Yang Xu et al. 2020) extends LayoutLM by integrating visual embeddings in the pre-training stage. Following contextualized word embeddings, contextualized image embeddings are expected to capture each image region semantics in the context of its entire

visual neighborhood. In addition to the text (w_1, \dots, w_n) extracted from a document page image via OCR, the model introduces visual tokens (v_1, \dots, v_{WH}) . These visual tokens are prepended to the text sequence, creating an input sequence $(v_1, \dots, v_{WH}, w_1, \dots, w_n)$ that combines visual and textual tokens. Visual tokens are obtained by feeding the document image to a visual encoder, namely ResNeXt-FPN (Xie et al. 2017; T.-Y. Lin et al. 2017). Finally, LayoutLMv2 integrates the embedding of each textual and visual token with its corresponding layout embedding.

Unlike LayoutLMv2, where visual and text features are concatenated into a single sequence, DocFormer (Appalaraju et al. 2021) introduces an alternative approach. Similarly to layout features, visual features are disentangled and introduced as residual connections to each layer of a Transformer encoder. This design choice aims to enforce the correlation between language and vision modalities.

LAMPReT (T.-L. Wu et al. 2021) exploits block-level visual features by feeding the image contents corresponding to each text block to a pre-trained CNN. In addition, the model encodes the visual presentation of the text, such as font size and text formatting, within each block. Similarly, SelfDoc (P. Li et al. 2021) uses block-level visual features obtained with Faster R-CNN, processing them separately from language features through a visual encoder.

Completing the shift towards using raw image data, LayoutLMv3 (Y. Huang et al. 2022) and UDOP (Z. Tang et al. 2023) take a different approach to visual feature extraction. Instead of relying on a pre-trained CNN or Faster R-CNN backbone, both models use linear patches obtained by resizing the document image, uniformly splitting it, and encoding each patch using linear embeddings. This approach not only saves parameters but also eliminates the need for region annotations in training object detectors. Figure 4.3 depicts the image encoding process of LayoutLMv3 in comparison to DocFormer and SelfDoc.

4.3.3 Pre-training Multimodal Transformers for Document Understanding

Efficient pre-training allows models to exploit cross-modal interactions and generalize across different document types and tasks. We explore initialization strategies, model architectures, pre-training datasets, and novel pre-training tasks that specifically address the challenges posed by document understanding.

4.3.3.1 Model Initialization

Several models take advantage of existing powerful Pre-trained Language Models and adapt them to document understanding tasks. LayoutLM (Yiheng Xu et al. 2020) is initialized with the weights of BERT (Devlin et al. 2018), LayoutLMv2 (Yang Xu et al. 2020) leverages UniLMv2 (Bao et al. 2020), while ERNIE-Layout (Peng et al. 2022) and StructuralLM (C. Li et al. 2021) are initialized from RoBERTa (Yinhan Liu et al. 2019).

4.3.3.2 Model Architecture

The majority of document understanding models typically are typically Transformer encoders. This choice is motivated by treating document understanding tasks as natural language understanding tasks, emphasizing comprehension and representation over generation. However, certain studies unify document understanding tasks under one framework by casting them as sequence-to-sequence problems, thereby expanding the language generation capabilities of the models and removing the need for additional task-specific layers. Donut (Kim et al. 2022) is an OCR-free encoder-decoder model that consists of a visual encoder initialized with Swin Transformer (Ze Liu et al. 2021), and a decoder that uses the weights of mBART (Yinhan Liu et al. 2020). TILT (Powalski et al. 2021) adds layout and visual information into Text-to-Text Transfer Transformer (T₅) (Raffel et al. 2020). UDOP (Z. Tang et al. 2023) unifies text, layout, and image modalities through an encoder, and generates all three modalities using a decoder. The text encoder-decoder is initialized with T₅, whereas the visual decoder employs the weights of MAE (K. He et al. 2022). LayoutT₅ and LayoutBART (Tanaka et al. 2021) add 2D position embeddings and visual features to T₅ and Bidirectional and Auto-Regressive Transformers (BART) (M. Lewis et al. 2019) in the fine-tuning stage. Similarly, we enhance the architecture of the long-range BigBird encoder-decoder (Zaheer et al. 2020) in Chapter 7 by incorporating layout information.

4.3.3.3 Large-scale Pre-training Datasets

LayoutLM (Yiheng Xu et al. 2020) pioneered the use of the IIT-CDIP Test Collection 1.0 (D. Lewis et al. 2006) as a prominent dataset for pre-training layout-aware language models for document understanding tasks. This large-scale dataset, extracted from the Legacy Tobacco Documents Library, features 6 million documents from lawsuits against American tobacco industries, spanning over 11 million scanned document images. IIT-CDIP contains document page images of diverse types and layouts, including news articles, scientific reports, handwritten materials, and more. The collection presents a spectrum of challenges attributed

to variations in image quality, resolution, and the potential presence of artifacts (e.g., stains) in the scanned document images. Alternatively, SelfDoc (P. Li et al. 2021) was pre-trained using RVL-CDIP (Harley et al. 2015), a subset of IIT-CDIP containing 400,000 grayscale images commonly employed for image classification, while TILT (Powalski et al. 2021) was pre-trained on a custom dataset containing documents from RVL-CDIP, the UCSF Industry Documents Library,⁵ and PDF files extracted from Common Crawl.⁶

4.3.3.4 Pre-training Tasks for Cross-modal Alignment

A plethora of pre-training strategies have emerged to learn cross-modal alignment. We categorize them into token masking strategies, image-based objectives, and approaches that directly leverage layout information.

Token Masking To pre-train LayoutLM, Yiheng Xu et al. (2020) introduced *Masked Visual-Language Modeling (MVLN)*, a self-supervised pre-training task that has become widely adopted in the document understanding field. Drawing inspiration from the Masked Language Modeling (MLM) strategy, MVLN aims to learn language representations by considering both semantics and spatial clues. The task consists in randomly masking some tokens while retaining their layout information, as well as layout information and semantics from the remaining tokens. The model is then trained to predict the masked tokens based on the contextual information. Therefore, not only does the model "understand" language contexts, but it also uses the corresponding 2D information, thereby bridging the gap between layout and language modalities. Rather than predicting individual tokens, another approach consists in predicting spans of tokens. Inspired by SpanBERT (Joshi et al. 2020), BROS (Hong et al. 2020) extends spans of a one-dimensional text to consecutive text bounding boxes in a two-dimensional space. This approach involves selecting a few regions in the document layout, masking all tokens within bounding boxes in the selected regions, and predicting the masked tokens.

Image-based Strategies Several strategies have been proposed to enforce alignment between language and image modalities. DocFormer (Appalaraju et al. 2021) and LayoutLMv2 (Yang Xu et al. 2020) predict whether a document image and a text are from the same document page. However, instances where the image and text are completely unrelated tend to be too easy for the model. Therefore, in LAMPReT (T.-L. Wu et al. 2021) and ERNIE-Layout (Peng et al. 2022), a set of randomly selected image patches is replaced with patches from other images, and the

5. <https://www.industrydocuments.ucsf.edu/>

6. <https://commoncrawl.org/>

model is tasked to predict which patches have been replaced. LayoutLMv2 uses another fine-grained task, wherein the image regions corresponding to randomly selected text tokens are masked, and the model is tasked with predicting whether the image region of each text token is masked. LayoutLMv3 (Y. Huang et al. 2022) employs a similar strategy but based on words instead of tokens. Rather than predicting whether an image region is masked, visual reconstruction strategies can be used. For instance, SelfDoc (P. Li et al. 2021) is tasked with regressing visual features for a set of randomly masked image regions, UDOP (Z. Tang et al. 2023) learns to reconstruct masked image patches, while DocFormer (Appalaraju et al. 2021) is trained to reconstruct the entire document image by passing its output features through a CNN-based image decoder. However, regressing region features can be noisy and challenging to optimize (J. Cho et al. 2020), while predicting raw pixels often leads to learning noisy details rather than high-level structures (Ramesh et al. 2021). Consequently, LayoutLMv3 learns to predict the (discrete) labels of masked image patches. These labels come from an image tokenizer that transforms dense image pixels into discrete tokens according to a visual vocabulary. In addition to image reconstruction, UDOP learns image-text correspondence by generating the text tokens at given locations in the page.

In general, there is a trend towards unifying image-based strategies and other modality-specific objectives. LayoutLMv3 addresses the disparity between text and image multimodal representation learning through unified discrete token reconstructive objectives, while UDOP unifies all document tasks to the sequence-to-sequence generation framework using a universal generative task format with task prompt.

Exploiting Layout Information Other pre-training strategies involve direct use of layout information. In addition to predicting masked tokens, UDOP (Z. Tang et al. 2023) is trained to locate them by generating their corresponding bounding boxes. Furthermore, the model is tasked to output positions of groups of text tokens, given both the document image and contextual text. To exploit the structural interactions among blocks, LAMPReT predicts whether two blocks are swapped, allowing it to learn the spatial order of blocks. In ERNIE-Layout’s architecture, the lack of explicit boundary between blocks requires learning the relationship between layout and reading order. Therefore, the model is tasked to learn the reading order by predicting whether two tokens are consecutive. In Chapter 6, our approach employs the same approach to ensure that the model effectively captures the correlation between spatial arrangement of tokens and reading order.

Supervised Training As the IIT-CDIP Test Collection includes multiple tags for each document image, LayoutLM is also pre-trained using *Multi-Label Docu-*

ment Classification, a task that supervises the pre-training process using document tags such as advertisement, letter, and form. The goal is to enable the model to aggregate knowledge from various domains, thereby enhancing document-level representations. TILT is also pre-trained on a set of supervised tasks, including visual information extraction and visual document question answering. However, given that the majority of datasets lack annotations, most document understanding models rely on unsupervised pre-training alone. Moreover, larger models trained for longer derive advantages from processing a larger share of unlabeled data (Raffel et al. 2020).

4.3.4 Extending Layout/Visually-aware Transformers

Document understanding models have primarily been designed for short documents, such as receipts, forms, and invoices, typically written in English. We discuss the extension of multimodal Transformers to non-English languages and longer documents, addressing challenges and strategies for broader applicability.

4.3.4.1 Multilinguality

Document understanding models have demonstrated successful application on English Documents. However, visually-rich documents typically exhibit varying formats and layouts based on the country, and this diversity even extends to different regions within the same country. To bridge the language barriers for real-world document understanding, LayoutXLM (Yiheng Xu et al. 2021) carries out multilingual pre-training by expanding the language support of LayoutLMv2 to a total of 53 languages.

4.3.4.2 Capturing Long-range Dependencies

The majority of multimodal pre-trained models focus on short documents due to the computational requirements and memory limitations of the Transformer. Furthermore, incorporating layout and visual information renders them more resource-intensive compared to pre-trained models that exclusively deal with text. As such, there has been limited research dedicated to long document understanding. However, real-world documents, such as business documents, can be very dense and long. The conventional approach for handling long documents involves splitting them into short segments and processing each segment independently. This approach presents a significant challenge for multi-page and cross-page understanding of long documents, where useful information is usually distributed across their lengths. While the component-level formulation (C. Li et al. 2021; P.

Li et al. 2021) can reduce the input sequence length for a document, it does not allow capturing long-range dependencies. Therefore, processing long documents requires a suitable method to connect information across pages.

In the context of Document Understanding, models can leverage the structure of documents (*e.g.*, page or block information), to sparsify attention. To handle multi-page documents, Pramanik et al. (2020)'s model encodes the page number into token representations and uses the Longformer architecture as its backbone. H. Pham et al. (2022)'s approach introduces a plug-able approach to integrating spatial input into self-attention, removing the need for extra embeddings. Drawing inspiration from the sliding window approach in Longformer (Beltagy et al. 2020), the model introduces attention masks based on spatial information to limit the context of each token to its neighbors in the 2D page. In this approach, each context window for a bounding box is defined by calculating its spatial neighbors, rather than relying on neighboring words determined by the sequential order obtained from an OCR engine. This approach is similar to our contribution in Chapter 5, where attention is sparsified using a layout-based masking approach.

While progress has been made in addressing the challenges of handling dense and lengthy real-world documents, the exploration of effective and efficient methods for long document understanding remains an ongoing and still largely under-explored area of research.

4.4 Conclusion

Research on understanding documents with complex interplay of text, layout and visual elements has garnered significant attention for document understanding tasks. Notably, general-purpose multimodal Pre-trained Language Models have experienced a surge in usage due to their remarkable performance. Yet, the academic literature on methodologies addressing Document Understanding remains relatively scarce when compared to fields with abundant publicly available data, such as image classification and translation. While advancements in Deep Learning has led to significant improvements in document understanding tasks, the field faces several challenges in practical applications.

Firstly, the limited input length of Pre-trained Language Models poses difficulties in processing long documents. Secondly, despite the increasing availability of digital-born documents, a significant portion of real-world documents, often originating from scanning equipment and affected by issues such as crumpled paper, deviate in quality from annotated training data. This can lead to suboptimal performance in document understanding models. Addressing this issue involves data synthesis and augmentation techniques. Thirdly, existing docu-

ment understanding tasks are often treated independently from each other, and there is a lack of effective leveraging of correlations between different tasks. This contrasts with recent advancements in foundation models like ChatGPT, which have demonstrated the benefits of leveraging inter-task correlations. Fourthly, pre-trained document understanding models encounter challenges due to insufficient computing resources and labeled training samples in practical applications. This emphasizes the importance of research directions such as model compression, few-shot learning, and zero-shot learning. Finally, the relationship between OCR and document understanding tasks is crucial. Given that document understanding systems typically receive input from OCR engines, the accuracy of text recognition and the ordering of words play pivotal roles in downstream tasks. Addressing these challenges and research directions is essential for advancing the field of Document Understanding in practical scenarios.

Our contributions aim to address key challenges in the field of Document Understanding. In Chapter 5, we introduce *Skim-Attention*, a novel attention mechanism that leverages the structure of documents to mimic human reading strategies, exploiting layout in a computationally efficient manner. In Chapter 6, we present an attention-based module, *Layout2Pos*, that learns position embeddings from the spatial position of tokens in the page, providing a solution to serialization errors by completely discarding sequential position information in downstream tasks. Finally, in Chapter 7, we introduce *LoRaLay*, a multilingual collection of datasets designed for long-range summarization. These datasets come with accompanying visual and layout information, as well as baselines merging layout-aware and long-range models, providing novel resources for researchers to explore the integration of multimodal information in long document modeling. Together, these contributions stand as innovative solutions to several challenges in the field of Document Understanding, enhancing the practical applicability of document understanding models.

LEVERAGING LAYOUT FOR SPARSE ATTENTION

Chapter abstract

While multimodal pre-training techniques prove effective, their development is not driven by efficiency considerations. Built upon the Transformer architecture, these models suffer from the quadratic complexity of self-attention, hindering their applicability to long documents. In this chapter, our focus lies in exploiting layout in a computationally efficient manner. In particular, we explore two research questions:

- 1. Is it possible to determine attention from layout only?*
- 2. Can layout help reduce the complexity of self-attention?*

Motivated by human reading strategies, we present Skim-Attention, a new attention mechanism that takes advantage of the structure of the document and its layout. Skim-Attention only depends on the 2-dimensional position of the words in a document. To exploit this mechanism, we introduce Skimformer and Skimming Mask, two frameworks for integrating Skim-Attention into the Transformer architecture. Our experiments show that Skim-Attention obtains a lower perplexity than prior works, while being more computationally efficient. We also show how Skim-Attention can be used off-the-shelf as a mask for any Pre-trained Language Model, allowing to improve their performance while restricting attention. Finally, we show the emergence of a document structure representation in Skim-Attention.

The work in this chapter has led to the publication of a conference paper:

- Laura Nguyen, Thomas Scialom, Jacopo Staiano, and Benjamin Piwowarski (Nov. 2021). “Skim-Attention: Learning to Focus via Document Layout”. In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 2413–2427. URL: <https://aclanthology.org/2021.findings-emnlp.207>.

Contents

| | | |
|-------|--|-----|
| 5.1 | Preliminary Experiments: Human Evaluation | 87 |
| 5.2 | Skim-Attention: A Novel Layout-Aware Attention Mechanism | 88 |
| 5.2.1 | Skim-Attention Overview | 89 |
| 5.2.2 | Skim-Attention in Transformers | 90 |
| 5.3 | Experiments | 94 |
| 5.3.1 | Data | 94 |
| 5.3.2 | Experimental Settings | 96 |
| 5.4 | Results | 97 |
| 5.4.1 | Language Modeling Evaluation | 97 |
| 5.4.2 | Document Layout Analysis Evaluation | 99 |
| 5.4.3 | Attention Visualization | 101 |
| 5.5 | Conclusion | 102 |

As seen in Chapter 4, Transformer-based joint pre-training of text, layout and images has allowed models to reach state-of-the-art performance in a number of document understanding tasks. However, multimodal pre-trained models suffer from very high computational and memory costs due to the quadratic complexity inherent to Transformers, rendering them unable to process long documents. In most approaches, layout is considered an additional feature (*e.g.*, integrating 2D coordinates as an extension of 1D positions) that enhances performance, and each token is still contextualized with respect to the entire input sequence. Yet, layout stands as a distinct modality alongside language. The conventional approach of treating layout as a special positional feature fails to fully exploit the strong correlation between modalities in documents, leading to a lack of cross-modal interaction between layout and text. This limitation has the potential to impede the model’s understanding of the role that layout plays in semantic expression.

Considering cognitive aspects, it has been shown that a well-designed layout results in less cognitive effort (Kieras 1978; Britton et al. 1982; Olive and Barbier 2017) and facilitates comprehension of the conveyed information by helping identify the document type and its constituents, as well as providing cues regarding relationships between elements (Meyer et al. 1980; Wright 1999; Lemarié et al. 2008). Semiotic research hypothesizes that readers scan the document before taking a closer look at certain units (Kress, Van Leeuwen, et al. 1996), a claim supported by eye-tracking experiments on newspapers (Leckner 2012). Inspired by these research findings, we claim that one does not need to have read each word in a document page to be able to understand a specific paragraph. Therefore, we argue that, to efficiently process long and structurally-complex documents, it is a waste of effort and computation to contextualize a token with respect to the entire

input sequence. To shift towards processing long documents with awareness of their structure, we propose to leverage layout in a more intuitive and efficient way, resembling human cognition. We argue that this approach can be key to a model coping with long and complex documents.

We delve into two research questions:

1. Can attention be effectively learned using layout information exclusively?
2. Can layout be used to reduce the complexity of self-attention?

In addressing these questions, we propose Skim-Attention, a new self-attention mechanism that relies solely on the 2D position of tokens in the page, independently of their semantics. To exploit this mechanism, we introduce Skimformer and Skimming Mask, two frameworks for integrating Skim-Attention into Transformer models. Skimformer is an end-to-end Transformer language model that computes the Skim-Attention scores just once and uses them across each layer of a text-based Transformer encoder. This design mirrors human reading strategies, where the model skims through the document to extract its structure, and reads the contents based on the prior structural understanding. Moreover, Skimformer can be adapted for long-range Transformers to model longer documents. On the other hand, Skimming Mask uses Skim-Attention as a masking mechanism to sparsify attention within any Transformer language model. Each token's attention is restricted to at most k tokens, as identified by Skim-Attention. This approach allows for a smaller context length, thereby leading to more efficient computation.

In this chapter, we first present a simple cognitive experiment that highlights the fundamental role of layout in humans' comprehension of documents. Then, we introduce the Skim-Attention mechanism and its integration into Transformer language models. Finally, we showcase the quantitative and qualitative benefits of our approach.

5.1 Preliminary Experiments: Human Evaluation

How much does the document layout help in comprehending long textual contents? How faster is it for humans to find information in documents when layout is provided? To answer these questions, we conduct a simple cognitive experiment wherein we measure the amount of time needed for human annotators to extract information from both formatted and plain-text documents. We hand-pick four document pages from the DocBank dataset (M. Li et al. 2020), and create a plain-text version out of each of these documents by serializing them. We create two basic questions for each document, and ask four annotators to answer them.

| | Average | Standard Deviation |
|------------|---------|--------------------|
| Formatted | 6.05 | 1.73 |
| Plain-text | 15.18 | 9.06 |

Table 5.1 – Average (std) time (in seconds) required to answer questions from documents, depending on whether layout is provided.

Half of the time, annotators are given access to the full layout, while the other half, they are limited to plain text only (i.e., no layout nor formatting).

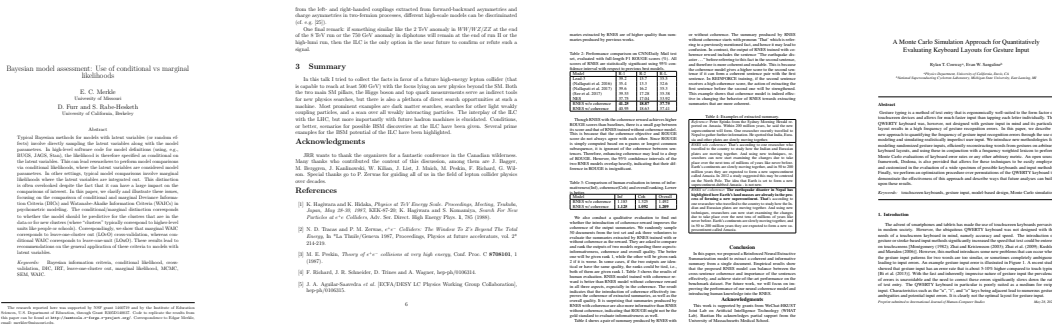


Figure 5.1 – Documents selected for our preliminary cognitive experiment.

Table 5.1 reports the average time needed to retrieve information from the documents. We find that it is 2.5 times faster to answer questions from the formatted documents, and that the variability in the results is much lower in this case. These results support the hypothesis that *less cognitive effort* is spent when the document is formatted, emphasizing the importance of layout information in reading comprehension.

We make the hypothesis that machines could benefit from the the document layout, just like humans, as a strategy to retrieve information faster while expending less effort. In particular, layout information could be of great help in reducing the cost of self-attention in Transformer models (Section 5.4.1.3) and facilitating the emergence of a document structure representation (Section 5.4.3).

5.2 Skim-Attention: A Novel Layout-Aware Attention Mechanism

In light of the aforementioned cognitive experiment, it is clear that layout is of utmost importance for humans to understand long documents. We propose to take layout into consideration by introducing *Skim-Attention*, a self-attention

module that computes attention solely based on the spatial positions of tokens. To process long and layout-rich documents, we present different ways of integrating this mechanism into Transformer architectures.

5.2.1 Skim-Attention Overview

Our novel attention mechanism, Skim-Attention, views documents as collections of word bounding boxes distributed over a two-dimensional space, *i.e.*, the page. In the following, we provide details on how to encode spatial positions into layout embeddings, followed by a detailed description of our attention module.

5.2.1.1 Layout Embeddings

Layout embeddings carry information about the spatial position of the tokens. Following LayoutLM (Yiheng Xu et al. 2020), the spatial position of a token is represented by its bounding box in the document page image, (x_0, y_0, x_1, y_1) , where (x_0, y_0) and (x_1, y_1) respectively denote the coordinates of the top-left and bottom-right corners. We discretize and normalize them to integers in $\{0, \dots, 1000\}$. Four embedding tables are employed to encode spatial positions: LE_x and LE_y for the coordinate axes (x and y), and LE_w and LE_h for the bounding box size (width and height). The final layout embedding of a token, $\ell \in \mathbb{R}^{d_\ell}$, located at position (x_0, y_0, x_1, y_1) is defined by:

$$\begin{aligned} \ell = & \text{LE}_x(x_0) + \text{LE}_y(y_0) \\ & + \text{LE}_x(x_1) + \text{LE}_y(y_1) \\ & + \text{LE}_w(x_1 - x_0) \\ & + \text{LE}_h(y_1 - y_0) \end{aligned} \tag{5.1}$$

5.2.1.2 Skim-Attention Mechanism

A standard self-attention mechanism works by comparing every token in the sequence to every other token in the sequence, and reweighing the embeddings of each token to include contextual relevance (Section 3.1.1.3). Skim-Attention diverges from standard self-attention as it operates independently from text semantics (*i.e.*, token representations). Instead, Skim-Attention computes attention using *only* the spatial positions of tokens, *i.e.*, their layout embeddings ℓ .

Formally, let $\mathbf{X}^\ell = \{\ell_0, \ell_1, \dots, \ell_n\}$ be an input sequence of layout embeddings, and $\mathbf{Q}^\ell = \mathbf{W}_q^\ell \mathbf{X}^\ell$, $\mathbf{K}^\ell = \mathbf{W}_k^\ell \mathbf{X}^\ell$. The queries \mathbf{Q}^ℓ and keys \mathbf{K}^ℓ are obtained by linear transformations of the layout embeddings, following the standard attention

mechanism. As in Equation 3.7, Skim-Attention is defined as a scaled-dot product between queries and keys, passed through a softmax operator:

$$\mathbf{A}^\ell = \text{Softmax} \left(\frac{\mathbf{Q}^\ell (\mathbf{K}^\ell)^\top}{\sqrt{d^\ell}} \right) \quad (5.2)$$

Similar to multi-head self-attention in Transformers, Skim-Attention is decomposed in multiple heads.

Intuitively, \mathbf{A}^ℓ captures the correlation between two tokens based on their spatial positions: the more similar two tokens are in terms of layout embeddings, the more they should attend to each other.

Because attention is calculated only once, using non-contextualized layout embeddings may limit Skim-Attention’s ability to capture complex relationships within the document structure. Therefore, to enhance the quality of layout representations, we contextualize them by adding a small Transformer prior to computing Skim-Attention. The contextualization process allows for a more refined understanding of the relationships and dependencies within the layout.

The integration of Skim-Attention with any long-range Transformer is entirely feasible, as these approaches operate independently. We tailor our methodology by performing the long-range attention calculation once, using layout information instead of text semantics.

5.2.2 Skim-Attention in Transformers

We now present two approaches to exploit Skim-Attention: *i) Skimformer*, wherein self-attention is replaced by Skim-Attention; and *ii) Skimming Mask*, where an attention mask, built from Skim-Attention, is plugged into a Transformer language model to restrict its attention.

5.2.2.1 Skimformer

Skimformer is designed to answer the first research question: *Is it possible to determine attention from layout only?* *Skimformer* is a two-stage Transformer that replaces multiple standard self-attention layers with a single layer of Skim-Attention. Drawing inspiration from insights in cognitive science, the intuition behind this approach is to mimic how humans process a document by *i) skimming* through the document to extract its structure, and *ii) reading* the contents based on the prior structural understanding.

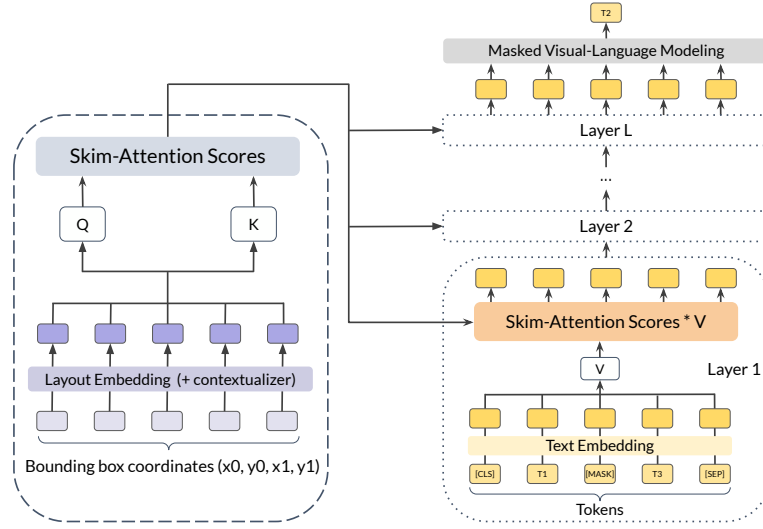


Figure 5.2 – Skimformer model architecture. The input consists of two components: a sequence of tokens (right-hand side) and a sequence of token bounding box coordinates (left-hand side). Only the layout embeddings (left) are used to compute Skim-Attention. L denotes the number of Transformer encoder layers. Q and K are the queries and keys obtained by projecting the layout embeddings. V represents the values produced by projecting the encoder layers’ textual inputs. The attention is solely based on token spatial positions and computed only once. The attention scores are then distributed to each layer of a Transformer encoder.

Skimformer is fed with a sequence of token embeddings and the corresponding sequence of layout embeddings. Because layout information implicitly reflects the reading order of documents, we do not encode the sequential positions of tokens. The model adopts a two-step approach: first, it computes the skim-attention scores (once and only once) using layout information alone. It then uses these attention scores across all layers of a Transformer encoder. The architecture of Skimformer is depicted in Figure 5.2.

For a given encoder layer and a single head, the output representations Z of the standard self-attention operation becomes:

$$Z = A^\ell V^t \quad (5.3)$$

where A^ℓ is the skim-attention matrix obtained through Eq. 5.2, and $V^t = W_v X^t$ is the value matrix obtained by projecting the layer input.

More intuitively, computing skim-attention scores (Eq. 5.2) can be interpreted as *skimming through* the document to grasp its structural aspects. Information about the semantics (contained in V) is then routed based on these similarity scores. This is done via Eq. 5.3 and can be seen as *reading* the contents of the document, focusing on the most relevant parts guided by the skim-attention scores.

Similarly to LayoutLM, we pre-train Skimformer using Masked Visual-Language Modeling (MVLM). This involves randomly masking some of the input tokens, while retaining their corresponding layout embeddings. The model is then trained to recover the masked tokens given the contexts.

While we experimented with a standard Transformer encoder-only model, it is worth noting that any language model can be used as the backbone of Skimformer.

Albeit remaining quadratic, the time and memory cost of Skim-Attention is much lower than vanilla self-attention. Let n be the maximum sequence length, L the number of encoder layers, L' the number of layers needed to contextualize layout representations, d the dimension of the text embeddings, and d' the dimension of the layout embeddings. The computational complexity is reduced from $\mathcal{O}(2Ldn^2)$ to $\mathcal{O}(L'd'n^2 + Ldn^2)$, the first term being the time required to calculate the skim-attention scores, and the second term referring to the time needed to compute the residual value for each token in the input. The memory complexity for vanilla self-attention is $\mathcal{O}(Ldn + Ln^2)$, where the first term is the memory required to store keys, queries and values, while the second represents the attention scores produced. These requirements are reduced to $\mathcal{O}((d'n + n^2) + Ldn)$, with the first term corresponding to the keys and queries, the second term representing the attention scores, and the last one corresponding to the values.

In conclusion, Skimformer is a two-stage Transformer architecture designed to determine attention from layout information alone, inspired by human reading strategies that efficiently process document structure. While enhancing efficiency, Skimformer has suboptimal aspects. One limitation lies in the fixed nature of Skim-Attention scores across all layers, which may oversimplify the model’s comprehension, potentially missing complex relationships within the document. Furthermore, a token’s attention is determined solely by layout information, potentially overlooking important semantic clues that are essential to understand complex relationships.

5.2.2.2 Skimming Mask

One drawback of Skimformer is that it does not use attention based on semantics. However, for each token in a sequence, Skim-Attention provides a ranking of every other token in accordance with their layout-based similarity. Drawing from this observation, *Skimming Mask* answers the second question: *Can layout*

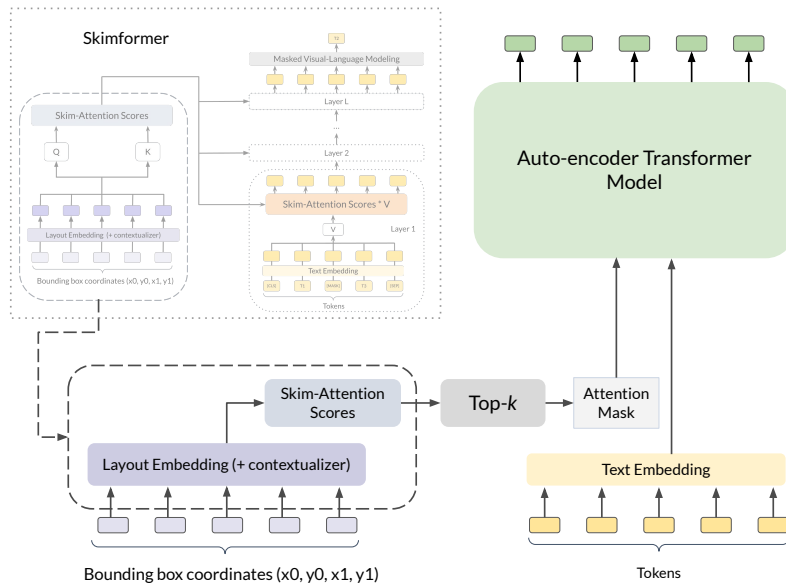


Figure 5.3 – Skimming Mask model architecture. The layout embeddings, Key and Query projections are initialized from an already pre-trained Skimformer model. By filtering the k most attended tokens for each token, the Skim-Attention scores are then converted to an attention mask and given as input to a text-based Transformer model.

help reduce the complexity of self-attention? by using Skim-Attention as a *mask* to restrict the computation of self-attention to a smaller number of elements for each token. In this setting, Skim-Attention is viewed as an independent, complementary module that can be plugged into any Transformer-based language model. Given a sequence of layout embeddings, the corresponding skim-attention matrix is converted to an attention mask: based on the similarity scores provided in the attention matrix, each token only attends to at most k tokens. The mask is given as input to a text-based Transformer language model with vanilla self-attention, and is used to restrict self-attention for each element in the input text sequence. This can be viewed as *sparsifying* the standard self-attention matrix. In contrast to the sparse pattern methods discussed in Section 3.3.1.1, this sparsity is determined by the document’s layout.

It is worth nothing that Skimming Mask introduces a new way to cluster tokens: tokens within the same group have a high similarity to each other in terms of their respective *layout positions*. This characteristic positions Skimming Mask as a concurrent approach to Reformer (Kitaev et al. 2020), which reduces the cost of self-attention by clustering tokens into chunks. As opposed to the latter, the concept of similarity is not derived from text semantics but rather from the document structure. Furthermore, Skimming Mask does not require an understanding of the semantic content; it solely relies on their layout features. Because each token

is viewed as a bounding box whose characteristics are only its size and position, the representation space of layout features is much smaller than that of the text, which spans a vocabulary of more than 30k sub-words. As a consequence, computing attention based on layout could require a smaller latent space dimension than for text, corresponding to less computational efforts. This is also the case for humans: as demonstrated in section 5.1, it is much easier to retrieve information from documents when the layout is provided.

Ideally, the Skimming Mask pattern should be learned in an end-to-end fashion alongside the Transformer model. However, achieving this end-to-end training in practice is challenging due to the non-differentiable nature of generating an attention mask from an attention matrix. Consequently, to train this model, the weights for Skim-Attention must have already undergone training. In this context, we naturally use the pre-trained Skimformer weights. The overall architecture of the model is illustrated in Figure 5.3.

The computational complexity is reduced to $\mathcal{O}(L'd'n^2 + Ldk^2 + Ldn^2)$, where the first term represents the time required to compute the skim-attention scores, the second term corresponds to the time needed to calculate the sparse attention matrix, and the third term pertains to the time required to compute the residual value for each token in the input. Meanwhile, the memory complexity for Skimming Mask is reduced to $\mathcal{O}((d'n + n^2) + Ldk + Lk^2)$, encompassing the memory requirements for Skim-Attention and those for the sparsified attention mechanism.

5.3 Experiments

We first present the data used to pre-train and evaluate our models, and provide details on the experimental settings.

5.3.1 Data

5.3.1.1 Pre-training Data

To pre-train our models on a wide variety of document formats, we select three datasets with various non-trivial document layouts: DocBank (M. Li et al. 2020), RVL-CDIP (Harley et al. 2015) and PubLayNet (Zhong et al. 2019). We combine them by randomly selecting 25k documents from each dataset, for a total of 75K documents. We discard the provided labels and consider these data

as unannotated. The resulting dataset is referred to as *MIX*. As a first evaluation metric, we can compare the perplexity for the different language models on *MIX*.

DocBank (M. Li et al. 2020) is a large-scale dataset that contains 500k English document pages from papers extracted from the arXiv repository. These articles span a variety of disciplines (*e.g.*, Physics, Mathematics, and Computer Science), which is beneficial to train more robust models. Pages are split into a training set, validation set and test set with a ratio of 8:1:1. As the authors already extracted the text and bounding boxes using PDFPlumber,¹ there is no need for an Optical Character Recognition (OCR) system or a PDF parser. To build our subset, we extract 25k document pages: 20k from the full training set, 2,500 from the validation set and 2,500 from the test set.

RVL-CDIP (Harley et al. 2015) is a large collection of 400k scanned document images from various categories (*e.g.*, letter, form, advertisement, invoice). The wide range of layouts, as well as the low image quality, allows to train more robust models. We select 25k documents from the RVL-CDIP dataset available on Kaggle,² which amounts to half of the training images from the full dataset (160k images). The text and word bounding boxes are extracted using Tesseract (Kay 2007). We split the data into 80% for training, 10% for validation and 10% for test.

PubLayNet (Zhong et al. 2019) comprises over 360k document images from PubMed Central™ Open Access. The medical publications contained in the collection have similar layouts, but the text density coupled with the small image size allows to train more robust models. We arbitrarily extract the first training split among the 7 available on IBM Data Asset eXchange³ and use the first 20k images as our training set. For the validation and test sets, we keep the first 2,500 images in each split. Because Tesseract’s accuracy is too low without any pre-processing, we apply a few image processing operations (*i.e.*, rescaling, converting to grayscale, applying dilation and erosion) on each image to improve text extraction.

5.3.1.2 Dataset for Document Layout Analysis

In addition to perplexity, we evaluate our approach on a downstream task, document layout analysis, which consists in associating each token with its corresponding category. We use a subset of the full DocBank dataset, where the

1. <https://github.com/jsvine/pdfplumber>

2. <https://www.kaggle.com/nbhativp/first-half-training>

3. <https://developer.ibm.com/exchanges/data/all/publaynet/>

categories are: abstract, author, caption, date, equation, footer, list, paragraph, reference, section, table, title and figure.⁴

The subset is created by selecting 10k document pages (distinct from the ones used for pre-training): 8,000 from the full training set, 1,000 from the validation set and 1,000 from the test set. We refer to this dataset as *DocBank-LA*. Each document page is organized as a list of words with bounding boxes, colors, fonts and labels. We use the precision, recall and F1 score defined by M. Li et al. (2020).

5.3.2 Experimental Settings

For reproducibility purposes, the code and data pre-processing scripts are made publicly available.⁵

5.3.2.1 Baselines

We compare our models with three baselines: i) the text-only Bidirectional Encoder Representations from Transformers (**BERT**) (Devlin et al. 2018), ii) the multi-modal LayoutLM (Yiheng Xu et al. 2020), and iii) the text-only Longformer (Beltagy et al. 2020) for long documents. Note that the LayoutLM architecture is based on **BERT**, with additional layout components. For fair comparison, all our models designed for short sequences are based on **BERT** as well, as detailed below. All the models are trained from scratch.

5.3.2.2 Pre-training

For **BERT**, LayoutLM and Longformer, we use their base architecture. Following the **BERT** base model, Skimformer consists of a 12-layer Transformer encoder with 12 attention heads and a hidden size set to 768 for both text and layout embeddings, amounting to 99M parameters. We further add a 2-layer Transformer encoder to contextualize the layout embeddings, which increases the number of parameters to 113M. To test Skim-Attention on longer documents, we build LongSkimformer, a combination of Skim-Attention and Longformer. Every model is trained from scratch on the MIX dataset for 10k steps. We set the maximum sequence length to $n = 512$ for every model except for Longformer and LongSkimformer, for which $n = 2,048$. While **BERT** and Longformer are pre-trained with Masked Language Modeling (**MLM**), Skimformer, LongSkimformer and LayoutLM

4. We actually discard the *Figure* label, as 1) our models do not take image features into account, and 2) the text associated with such elements is always the same, making the task trivial.

5. <https://github.com/recitalAI/skim-attention>

| Model | Test Perplexity |
|----------------------------------|-----------------|
| BERT (Devlin et al. 2018) | 357.11 |
| LayoutLM (Yiheng Xu et al. 2020) | 45.86 |
| Skimformer | 33.77 |
| Longformer (Beltagy et al. 2020) | 333.28 |
| LongSkimformer | 32.02 |

Table 5.2 – Test perplexity on the MIX dataset after 10k optimization steps. Each model was trained from scratch. Bold denotes the best score.

are pre-trained using *MVLM*, an extension of *MLM* that incorporates layout information into the pre-training task.

5.3.2.3 Document Layout Analysis

As DocBank contains fine-grained token-level annotations, we consider the document layout analysis task as a sequence labeling task. Each model pre-trained on MIX is fine-tuned on this downstream task for 10 epochs. For the Skimming Mask models, we selected the hyperparameter k —the number of tokens that can be attended—to on the validation set. We tested $k \in \{512, 384, 256, 128\}$.

5.4 Results

We first discuss the results obtained on language modeling and document layout analysis, before exploring the attention maps obtained by Skimformer.

5.4.1 Language Modeling Evaluation

5.4.1.1 Perplexity

In Table 5.2, we report the perplexity on the MIX dataset. We observe that Skimformer and LongSkimformer outperform both BERT and Longformer by a huge margin, while improving perplexity by more than 10 points over LayoutLM. In addition, Figure 5.4 demonstrates that Skimformer converges much faster than BERT, and slightly more than LayoutLM, indicating its efficiency in learning from the training data. However, it is worth acknowledging certain limitations. Firstly, the models are pre-trained for a relatively brief duration on a rather small dataset, which may impact their generalizability. Additionally, achieving better perplexity does not always guarantee improved performance in downstream tasks.

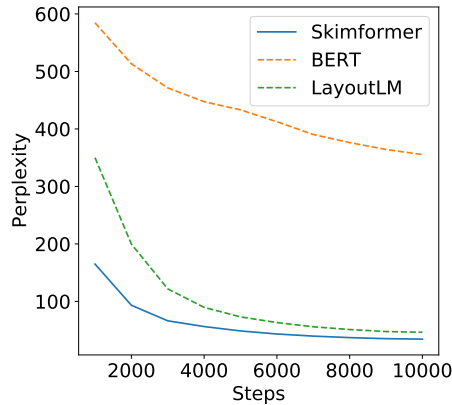


Figure 5.4 – Model perplexity on the MIX validation set with respect to the number of optimization steps. All models are trained from scratch.

| Skim-Attention Input | Test Perplexity |
|---------------------------|-----------------|
| Non-contextualized Layout | 36.41 |
| 1D position | 54.39 |
| Uniform layout | 421.97 |
| Degraded layout | 103.39 |
| Full model | 33.77 |

Table 5.3 – Ablation study on the MIX dataset, where perplexity on the test set is reported. All models were trained from scratch. Bold denotes the best score.

5.4.1.2 Ablation Study

We further conduct an ablation study about the influence of the Skim-Attention inputs on Skimformer’s performance. The results are listed in Table 5.3. To estimate the impact of the input type, we consider a Skimformer model i) wherein layout representations are not contextualized (*Non-contextualized Layout*), ii) Skim-Attention is based on sequential positions (*1D position*), iii) the bounding boxes are all set to the same fixed value, preventing the model to gather any information about the true location (*Uniform layout*), and iv) they are replaced by their centers (*Degraded layout*). We also provide the perplexity obtained by the full Skimformer model (*Full model*).

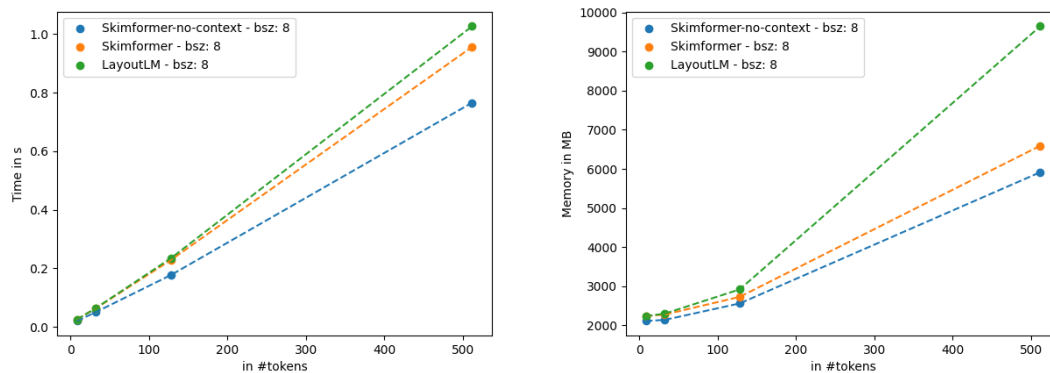
We find that computing Skim-Attention directly on the layout embeddings (*Non-contextualized Layout*) results in higher perplexity, indicating that depending solely on non-contextualized layout information results in increased uncertainty regarding the predictions made. Additionally, substituting spatial positions with sequential positions (*1D position*) leads to an increase in perplexity, indicating that layout information is crucial for the language model. We also observe that assigning the same bounding box to every token (*Uniform Layout*) leads to a severe

drop in performance. Coupled with the perplexity obtained with a degraded layout, this shows that the model’s performance is greatly impacted by the layout input quality.

5.4.1.3 Training Speed and Memory Usage

Using Hugging Face’s Transformers benchmarking tools (Wolf et al. 2019), we benchmark Skimformer and LayoutLM on both speed and required memory for pre-training. We consider the base variant of LayoutLM, and use the implementation from the Transformers library. In addition to Skimformer, we evaluate a variant in which the small Transformer contextualizing layout embeddings is removed (*Skimformer-no-context*). The batch size is fixed to 8, and memory and time performance is evaluated for the following sequence lengths: 8, 32, 128 and 512. All experiments were conducted on one Tesla T4 with 15GB of RAM.

Figures 5.5a and 5.5b report the time and peak memory consumption, respectively, with respect to the sequence length. Results confirm that Skimformer is more time and memory efficient than LayoutLM.



(a) Time usage for pre-training.

(b) Memory usage for pre-training.

Figure 5.5 – Comparison of time and memory usage for LayoutLM (green), Skimformer with layout contextualizer (orange) and without (blue). Results are plotted against sequence length.

5.4.2 Document Layout Analysis Evaluation

We now evaluate the performance of our models on a document layout analysis task using the DocBank-LA dataset. Table 5.4 reports the performance on DocBank-LA, the sequence length processed, the number of times attention is computed and the ratio of the total calculation unit ($n^2 \times \text{Nb Skim-Attn} + \text{Seq. Len}^2 \times$

| Model | Skimming Mask | Seq. Len | Nb Attention | | Total Compute | Rec. | Prec. | F1 |
|----------------------------------|---------------|----------|--------------|-----------|---------------|--------------|--------------|--------------|
| | | | Original | Skim-Attn | | | | |
| BERT (Devlin et al. 2018) | ✗ | 512 | 12 | 0 | 100.00% | 67.21 | 59.28 | 60.98 |
| LayoutLM (Yiheng Xu et al. 2020) | ✗ | 512 | 12 | 0 | 100.00% | 81.60 | 77.96 | 79.28 |
| Skimformer | ✗ | 512 | 0 | 3** | 25.00% | 78.80 | 74.35 | 75.86 |
| BERT+SkimEmbeddings | ✗ | 512 | 12 | 0 | 100.00% | 82.42 | 77.06 | 79.16 |
| BERT+SkimmingMask | ✓ | 128 | 12 | 3** | 31.25% | 72.32 | 64.39 | 67.36 |
| LayoutLM+SkimmingMask | ✓ | 128 | 12 | 3** | 31.25% | 81.15 | 78.30 | 79.26 |
| Longformer (Beltagy et al. 2020) | ✗ | 2,048 | 12 | 0 | 100% | 74.88 | 69.29 | 71.17 |
| LongSkimformer | ✗ | 2,048 | 0 | 3** | 25% | 81.22 | 73.45 | 76.61 |

*Standard self-attention for Skimformer, BERT-based and LayoutLM-based models. Longformer self-attention for Longformer and LongSkimformer. ** Attention is computed twice (by a 2-layer Transformer) during layout contextualization, then once by Skim-Attention.

Table 5.4 – Model performance (in %) on the DocBank-LA dataset. *Seq. Len* indicates the number of tokens attended with either standard attention (for Skimformer, BERT-based and LayoutLM-based models), or Longformer attention (for Longformer and LongSkimformer). *Nb Attention* represents the number of times attention (original and Skim-Attention) is computed and stored. *Total Compute* specifies the ratio of the final computational cost (# operations needed to compute attention) w.r.t. BERT/LayoutLM or Longformer. Each model was pre-trained from scratch on the MIX dataset, then fine-tuned on DocBank-LA.

Nb Standard Attn, where n is the length of the initial sequence on which Skim-Attention is applied; and *Seq. Len* is the length obtained after applying Skimming Mask) to that of BERT/LayoutLM and Longformer. All models were pre-trained from scratch on MIX.

Skimformer is substantially superior to BERT, improving the F1 score by 15% while reducing the number of attentions computed by four. We experimented with plugging the layout embeddings learnt by Skimformer in a BERT model. The resulting model, BERT+SkimEmbeddings, resembles LayoutLM in terms of architecture.⁶ Results show that BERT+SkimEmbeddings performs on par with LayoutLM despite simply combining separately pre-trained modalities, as opposed to the latter which requires an extensive joint training.

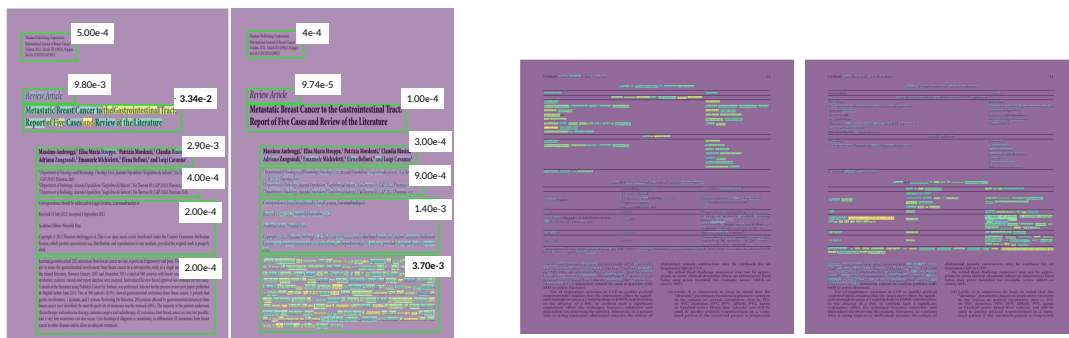
For the Skimming Mask models, the models attend only to the top-128 tokens. Compared to LayoutLM, reducing the number of tokens attended to from 512 to 128 allows to obtain the same downstream results with only 31.25% of the computational burden. Compared to BERT, it even obtains an absolute improvement of more than 6% in term of F1 score.

6. In BERT+SkimEmbeddings, the layout embeddings are first projected into the same dimensional space as the text embeddings. In this way, we can plug the layout embeddings from any Skimformer model, in particular smaller ones.

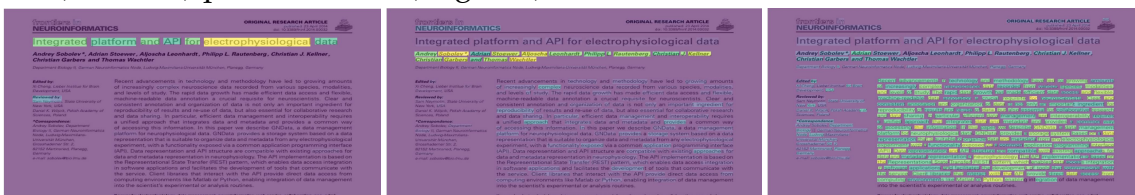
LongSkimformer benefits from both Skim-Attention and Longformer’s gain in efficiency. It outperforms Longformer by 5% while requiring four times less attention operations, and the use of Longformer’s linear attention allows LongSkimformer to process sequences four times larger than Skimformer can.

Overall, the competitive results achieved by Skimformer show that it is possible to learn attention solely from layout information. Additionally, the efficiency gains and effectiveness demonstrated by the Skimming Mask approach emphasize the potential of leveraging layout information to reduce the complexity of self-attention.

5.4.3 Attention Visualization



(a) Skim-attention maps corresponding to the title (left) and the abstract (right), with average attention score (in white) per text block (in green). (b) Skim-attention maps corresponding to the top table (left) and the bottom table (right).



(c) Skim-attention maps corresponding to the title (left), the authors (center) and the abstract (right).

Figure 5.6 – Skim-Attention maps obtained on three sample documents. We consider the skim-attention matrix averaged over all the attention heads. Given a semantic unit, we plot the average attention score for each token.

We explore the distribution of attention probabilities obtained by Skim-Attention. Figure 5.6 shows the attention maps produced by Skimformer on three randomly sampled documents. Given a semantic unit (either title or abstract in our example),

we select the corresponding tokens and compute their average attention over the whole document. We observe, both qualitatively and quantitatively, that tokens attend mainly to other elements in the same semantic unit, thereby creating clusters of tokens that are relevant to each other. This shows that the model has grasped the concept of semantic unit with only self-supervision, enabling the emergence of a document structure representation. We argue that these structure-aware clusters could pave the way for long text encoding and unsupervised document segmentation.

5.5 Conclusion

We have presented Skim-Attention, a novel structure-aware attention mechanism. Distinct from prior works in layout-aware pre-training, Skim-Attention is inspired by human reading strategies: rather than considering every single token in the document to compute attention, our approach exploits 2D positions. We conduct extensive experiments to show the effectiveness of Skim-Attention, both as an end-to-end model (Skimformer) and as a mask for any language model (Skimming Mask).

Potential extensions of this work include full-scale pre-training, integrating image features into Skim-Attention to leverage information across all modalities, as well as exploring tasks that require capturing longer-range dependencies.

Most pre-training techniques, including Skim-Attention, depend on serialized text obtained through OCR or PDF parsing. Yet, accurately detecting the *reading order* of visually-rich documents poses a significant challenge. With the aim of advancing models that capture document structure more intuitively, the next chapter will focus on addressing reading order issues by leveraging document layout and entirely discarding sequential position information.

LEVERAGING LAYOUT TO AVOID READING ORDER ISSUES

Chapter abstract

Due to their remarkable performance, general-purpose multimodal Pre-trained Language Models have gained widespread adoption for Document Understanding tasks. The majority of Pre-trained Language Models rely on serialized text, extracted using either Optical Character Recognition (OCR) or PDF parsing. However, accurately determining the reading order of visually-rich documents (VrDs) is challenging, potentially affecting the accuracy of the extracted text and leading to suboptimal performance in downstream tasks. For information extraction tasks, where entity recognition is commonly framed as a sequence-labeling task, incorrect reading order can hinder entity labeling. In this work, we avoid reading order issues by discarding sequential position information. Based on the intuition that layout contains the information to supplement reading order, we present Layout2Pos—a shallow Transformer designed to generate position embeddings from layout. Incorporated into a BART architecture, our approach demonstrates competitiveness with models dependent on reading order across three benchmark datasets for information extraction. We also show that evaluating models using a reading order different from the one seen during training can result in substantial performance drops, thereby highlighting the importance of not relying on the reading order of documents.

The work in this chapter has led to the submission of a paper currently under review at NAACL 2024.

Contents

| | | |
|-------|---|-----|
| 6.1 | Reconstructing Positional Information from 2D Positions | 107 |
| 6.1.1 | Preliminary Experiments: OCR Serialization Errors | 108 |
| 6.1.2 | Layout2Pos Module | 109 |
| 6.2 | Experiments | 115 |
| 6.2.1 | Data | 115 |
| 6.2.2 | Experimental Settings | 118 |
| 6.3 | Results and Discussion | 121 |
| 6.3.1 | Next Token Position Prediction | 121 |
| 6.3.2 | Visual Information Extraction | 122 |
| 6.4 | Conclusion | 124 |

The organization of textual content in a specific layout is crucial for conveying meaning and context, holding significant importance across various written materials, including business documents, scholarly papers, and news articles. In particular, layout determines the sequence in which text is intended to be read or processed within a document, *i.e.*, the *reading order*. A well-designed reading order ensures that readers can follow the logical flow and structure of information and comprehend the intended meaning of the text. However, defining a proper reading order is non-trivial due to the complexity of document layouts, which may include elements such as tables and multiple columns.

When models are trained with a reading order that aligns with human understanding, they learn to capture the relationships between words, sentences, and paragraphs. Hence, reading order is crucial for models to perform well. Most pre-training methods for Document Understanding rely on serialized text, where either an Optical Character Recognition (OCR) engine or a PDF parser is used to extract text. However, due to the variety of layout formats, most OCR engines and PDF parsers struggle to provide accurate reading orders, resulting in misalignments between the order of the extracted text and the order of the original visual content (*i.e.*, *serialization errors*).

Most of the time, the text extracted is re-arranged in a raster-scan order, aligning tokens from the top-left to the bottom-right corner (Clausner et al. 2013). However, this linear organization does not always align with human reading patterns, particularly in documents with complex layouts such as multicolumn texts, tables, and forms. Serialization errors impact the accuracy of the extracted text and, therefore, affect the entire text processing pipeline. Without an accurate reading order, models may misinterpret the relationships between different parts of the text, resulting in suboptimal performance in downstream tasks. This poses

a substantial challenge in various applications, notably in the field of Document Understanding, where document layouts can be complex.

Furthermore, in real-world scenarios, documents may be processed by various OCR engines for different reasons such as cost, availability, or integration with existing workflows. Each OCR engine may have unique characteristics related to layout, font handling, or language support. These differences can introduce variations in the quality and accuracy of the OCR output for the same document, impacting, most importantly, the reading order. This variability may result in significant fluctuations in downstream performance. However, it is crucial for organizations to be able to choose OCR engines based on their specific requirements without compromising performance in downstream tasks.

In particular, in visual information extraction tasks, the primary goal¹ is to identify entities of predefined semantic types (*e.g.*, names, dates, addresses). Following the classic settings of Natural Language Processing (NLP), the task is commonly framed as a sequence-labeling problem. This approach involves labeling each token using a tagging scheme, such as BIO-tagging (Ramshaw and M. P. Marcus 1999), and leveraging these tags to identify entities. A sequence labeling-based approach operates under the assumption that each identified segment of an entity forms a continuous sequence of words within the input. While this assumption is valid for plain texts, it may not hold for real-world documents, where OCR systems or PDF parsers might not correctly organize text. In this context, performance is notably impacted by serialization errors. For instance, an entity might be split into non-continuous fragments. Such disordered input disrupts the BIO-tagging scheme, preventing the models from accurately identifying entities. Moreover, for training sequence labeling approaches, word-level annotations are essential. However, they are not always available; some datasets only provide the text corresponding to each information type (Graliński et al. 2020). Obtaining word-level annotations involves matching these texts with the words in the document, which can be time-consuming if done manually or susceptible to errors if performed automatically.

On the other hand, layout inherently encapsulates the correct reading order of documents by visually organizing content in a structured manner. A well-designed layout guides the reader’s natural progression from one section to another, ensuring coherent and logical information flow. Therefore, understanding the layout provides essential cues for determining the correct reading order, as it aligns with the visual hierarchy and structure intended by document creators.

Yet, existing pre-training methods for Document Understanding often neglect this aspect, opting to oversimplify the integration of layout. For instance, LayoutLM

1. Additionally, the task extends to classifying the relationships between these recognized entities (*relation extraction*). In this work, we do not focus on this task.

(Yiheng Xu et al. 2020) incorporates layout information as an extra embedding in the input layer, while LayoutLMv2 (Yang Xu et al. 2020) adds it as a bias term in the attention layer. Although ERNIE-Layout (Peng et al. 2022) learns the relationship between layout and reading order through a pre-training strategy involving reading order prediction, it continues to rely on sequential position embeddings derived from the reading order obtained via OCR. However, by investigating the position-awareness in causal language models with no explicit positional encodings, Haviv et al. (2022) show that these models develop an implicit understanding of absolute positions to compensate for missing information. It is suggested that causal attention, a mechanism in which each token attends only to its preceding positions in a sequence, allows the model to estimate the number of predecessors each token can attend to, thereby approximating its absolute position. Our objective is to demonstrate that the capability of a model to approximate its absolute position is not restricted to causal models, but holds true across various scenarios when layout information is provided.

Unlike sequence labeling approaches that entirely depend on the content extracted via OCR, generative models can bypass the need for word-level annotations and generate text without being restricted by the document's content or its reading order, enabling them to potentially correct OCR-induced errors. Sage et al. (2020) represent the information to be extracted as a sequence of tokens in XML. They employ a recurrent encoder-decoder architecture to generate XML representations, using pointer-generator networks (See et al. 2017) to allow the model to dynamically decide whether to generate a word from its vocabulary or copy it directly from the document. Townsend et al. (2021) use a Transformer (Vaswani et al. 2017) language model trained on database records to generate JSON-like representation of the extracted information.

Close to our work, TILT (Powalski et al. 2021) is a Transformer encoder-decoder model enhanced with layout and visual information, specifically designed for visual information extraction tasks. Unlike TILT, our work does not depend on sequential position information. Going further, Donut (Kim et al. 2022) uses a Transformer visual encoder to extract features from a document image, eliminating the reliance on OCR for text extraction. A textual Transformer decoder is then used to map these visual features to a desired structured format, such as JSON, for visual information extraction tasks. In contrast to Donut, our method does not rely on visual features, providing better computational efficiency when various information extraction tasks are conducted or/and complex documents are processed.

In this chapter, we focus on mitigating serialization errors by *entirely discarding sequential position information*. We introduce *Layout2Pos*, a shallow Transformer

model designed to generate position embeddings from the document layout. Our endeavor is twofold: from a practical standpoint, we aim to enhance the robustness of models to reading order changes, crucial for real-world applications; from a theoretical perspective, we demonstrate that it is feasible to discard sequential position information without compromising overall performance. We integrate this module into a sequence-to-sequence framework. To train the model, the language modeling task is coupled with a pre-training strategy designed to instill the model with the ability to learn the reading order from layout information. This integration eliminates the reliance on reading order and enables the generation of values that are not explicitly present in the input. We demonstrate the benefits of our approach for visual information extraction tasks, showcasing competitive performance to models that depend on reading order.

6.1 Reconstructing Positional Information from 2D Positions

In this section, we present preliminary experiments that underscore the limitations inherent to OCR processing in terms of accurately preserving the reading order of tokens. In response to the challenges posed by OCR-induced serialization errors, we then introduce Layout2Pos, a novel approach that goes beyond conventional approaches by discarding the reliance on reading order information. Instead, Layout2Pos leverages the inherent 2D positional information of tokens within the document page to construct positional information, thereby enhancing the robustness and reliability of downstream tasks.



(a) Plain (b) List (c) Multicolumn (d) Table

Figure 6.1 – Examples of documents for each layout category, arranged from the simplest to the most complex.

6.1.1 Preliminary Experiments: OCR Serialization Errors

| | Layout Type | | | |
|--------------------------|-------------|-------|-------------|--------|
| | Plain | Lists | Multicolumn | Tables |
| Tesseract OCR (Kay 2007) | 78.71 | 72.75 | 61.43 | 36.97 |
| DocTR (Mindee 2021) | 86.83 | 77.83 | 82.54 | 66.11 |

Table 6.1 – Accuracy (in %) obtained by each OCR engine, for each document layout type.

To gauge the extent of OCR-induced serialization errors, we conduct preliminary experiments comparing the annotated ground-truth reading order against the reading orders produced by 1) Tesseract OCR (Kay 2007), a widely-used OCR engine, and 2) DocTR (Mindee 2021), an OCR engine based on deep learning models. The goal is to assess the alignment of the reading order produced via OCR with actual human reading patterns.

We use a subset of 100 documents from ReadingBank (Zilong Wang et al. 2021), a benchmark dataset for reading order detection that includes high-quality reading order annotations extracted from Word documents. These annotations capture the correct sequence of words as visually presented in the documents. Upon examination of the samples and discerning patterns that appear most frequently, we have identified four prevalent document layout types: *plain* layout, *lists*, *multi-column* layout, and *tables*. We provide examples in Figure 6.1.

Tesseract OCR and DocTR are employed to extract and serialize text from the documents. The reading orders produced are compared against the ground-truth for discrepancies. Specifically, we evaluate accuracy by comparing, for each word in the ground-truth sequence, the actual next word with the one predicted by each OCR engine. We compute the accuracy obtained by each system for each specific layout type. Results are reported in Table 6.1. Additionally, we include the results obtained by our approach, Layout2Pos. Our findings indicate that both OCR engines face increased difficulty in reconstructing the correct reading order as the document layout becomes more complex.² Our approach exhibits a similar trend, although to a significantly lesser degree. Furthermore, it demonstrates higher accuracy for each document type compared to both OCR engines.

Additionally, we provide the ground-truth reading order and the reading order generated by Tesseract OCR for a sample table (Figure 6.2) and a document with a multi-column layout (Figure 6.3). In the case of the table, a comparison with the ground-truth reading order reveals that Tesseract OCR lacks knowledge about

². This difficulty in accurately predicting the next word is further attributed to the OCR engines' misinterpretation of certain words.

| | | | | |
|----|--|----|--|--|
| | 0 1 | | 2 3 4 5 6 7 8 | 9 10 11 12 13 14 15 |
| 16 | 17 18 19 20 21 | 22 | 23 24 25 26 | 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 |
| 43 | Sequences— Infinite Geometric 44 | 45 | 46 47 48 | |
| 49 | 50 51 52 | 53 | 54 55 56 57 58 59 60 61 62 | |
| 63 | 64 65 66 | | | |

(a) Ground-truth reading order

| | | | | |
|----|--|----|--|----------------------------|
| | 0 10 | | 1 2 3 4 5 6 7 8 9 | 11 12 13 14 15 |
| 16 | 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 | 49 | 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 | |
| 72 | 73 74 | | | |

(b) Reading order obtained through Tesseract

Figure 6.2 – Ground-truth reading order (a) compared to the reading order generated by Tesseract (b) for a sample table. Arrows emphasize the differences in reading order in the first row. Non-highlighted text indicates that it does not appear in the serialized sequence.

cells, as it organizes text in a top-left to bottom-right manner. Regarding the document with two columns, Tesseract OCR reads the text column by column, despite the document being horizontally divided into subgroups. This observation highlights the limited understanding of structure by OCR engines.

These preliminary experiments provide the groundwork for investigating into novel approaches designed to alleviate serialization errors, ultimately enhancing the performance of document understanding models.

6.1.2 Layout2Pos Module

Building on the insights gained from the previous experiment, accurately retrieving the correct reading order poses a significant challenge for OCR engines.

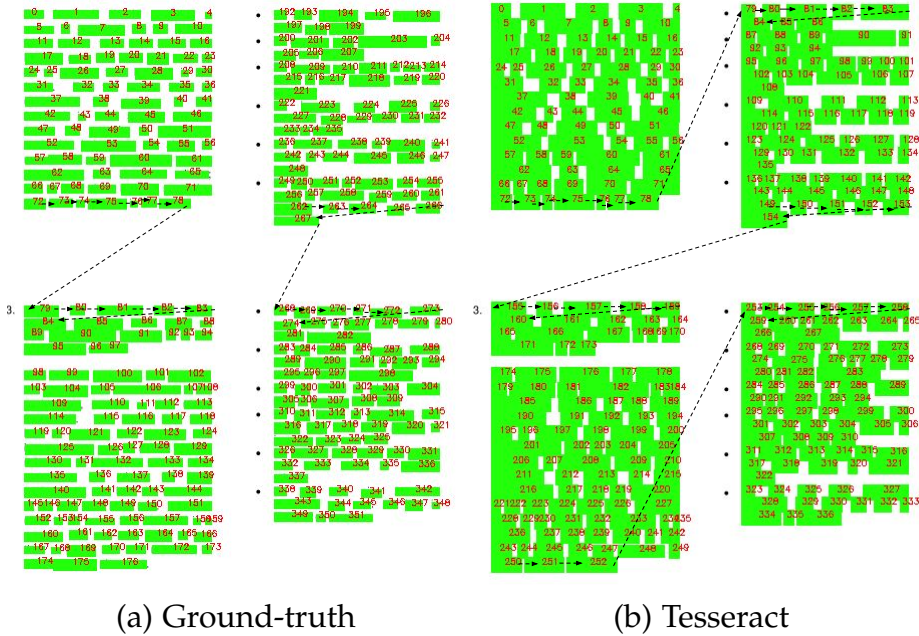


Figure 6.3 – Ground-truth reading order (a) compared to the reading order generated by Tesseract (b) for a document with a two-column layout. The document was cropped for better visibility. Arrows emphasize the differences in reading order. Non-highlighted text indicates that it does not appear in the serialized sequence.

We argue that it is possible to directly leverage document layout. To address this, we propose a novel approach, *Layout2Pos*, a transformer-based module that does not rely on the reading order generated by OCR, and learns position embeddings solely from the spatial positions of tokens. In the following, we elaborate on the process of encoding spatial information into layout embeddings, followed by an in-depth description of our *Layout2Pos* module.

6.1.2.1 Encoding Layout Information

To encode layout information, we use 1) bounding box information, 2) 2D relative positions, and 3) a novel method based on line and column relative positions. This approach distinguishes itself from *Skimformer* (Section 5), which relies solely on bounding box information. The inclusion of richer layout information in our method is crucial for effectively retrieving the reading order of documents.

Encoding Bounding Box Information The spatial position of a token is represented by its bounding box in the document page image, denoted as (x_0, y_0, x_1, y_1) , where (x_0, y_0) and (x_1, y_1) correspond to the coordinates of the top-left and bottom-right corners, respectively. Following *LayoutLMv2*, we discretize and normalize

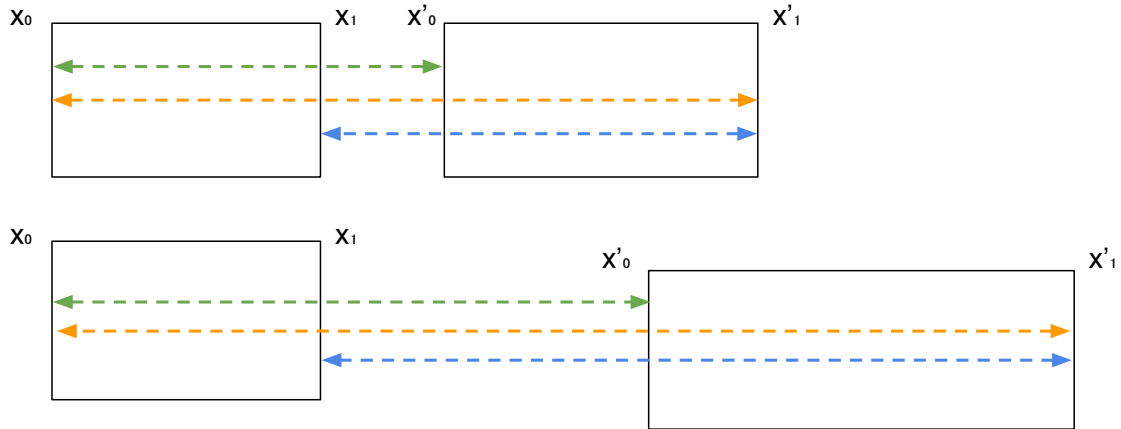


Figure 6.4 – Distances between: the left edge of each box ($x'_0 - x_0$, in green), the right edge of the second box and the left edge of the first ($x'_1 - x_0$, in orange), and the right edge of each box ($x'_1 - x_1$, in blue).

these coordinates to integers within the range of $\{0, \dots, 1000\}$. Four embedding tables are employed to encode spatial positions: LE_x and LE_y for the coordinate axes (x and y), and LE_w and LE_h for the bounding box size (width and height). In line with LayoutLMv2, the final layout embedding $\ell \in \mathbb{R}^d$ of a token, whose bounding box is (x_0, y_0, x_1, y_1) , is defined as follows (\parallel denotes concatenation):

$$\begin{aligned} \ell = & \text{LE}_x(x_0) \parallel \text{LE}_y(y_0) \\ & \parallel \text{LE}_x(x_1) \parallel \text{LE}_y(y_1) \\ & \parallel \text{LE}_w(x_1 - x_0) \\ & \parallel \text{LE}_h(y_1 - y_0), \end{aligned} \tag{6.1}$$

Leveraging 2D Relative Positions LayoutLMv2 encodes spatial relative positions as bias terms added to the attention scores to explicitly capture the spatial relationship between tokens (see Section 4.3.1.2). Following LayoutLMv2, for each pair of bounding boxes $((x_0, y_0, x_1, y_1), (x'_0, y'_0, x'_1, y'_1))$, we compute the vertical distance $y'_1 - y_1$ between the bottom edge of each box and the horizontal distance $x'_0 - x_0$ between the left edge of each box. The latter measurement, depicted in green in Figure 6.4, provides insights into how much space exists between the starting points of the two tokens.

Furthermore, we provide additional perspectives into the spatial relationships of tokens. We compute the horizontal distance $x'_1 - x_0$ between the right edge of the second box and the left edge of the first (illustrated in orange in Figure 6.4),

providing information about the gap between the beginning of the first token and the ending of the second one. In addition, we calculate the horizontal distance $x'_1 - x_1$ between the right edge of each box (shown in blue in Figure 6.4), indicating how far apart the ending points of the two tokens are.

Incorporating Line and Column Relative Positions Understanding the relative positions within columns provides information about the sequential structure of the document, aiding in distinguishing between different parts of the document. On the other hand, the relative positions within lines is valuable for documents with multicolumn layouts, offering insights into the spatial arrangement of text across columns. Hence, for each bounding box, we identify other bounding boxes that share the same line/column. This is determined by whether the horizontal/vertical line passing through the center of the box intersects with the other bounding boxes. If there is an intersection, the boxes are considered to be on the same line/column. For each token t_i , we determine its positions $p^{(l)}(i)$ and $p^{(c)}(i)$ within its corresponding line and column, using a left-to-right order for lines and a top-to-bottom order for columns. Then, we compute the relative sequential distance δ_{ij}^l and δ_{ij}^c between elements within each line and column. If they do not belong to the same line or column, the distance is set to ∞ .

Let us summarize formally how attention is computed. Suppose \mathbf{q}_i^ℓ and \mathbf{k}_i^ℓ denote the query and key projections obtained from the layout embedding ℓ_i of token i . Let $\mathbf{b}^{(2D_x)}$, $\mathbf{b}^{(2D_y)}$, $\mathbf{b}^{(l)}$, and $\mathbf{b}^{(c)}$ be the horizontal, vertical, line, and column relative position biases, respectively. In Layout2Pos, attention is re-defined as:

$$\begin{aligned} \alpha_{ij} = & \frac{1}{\sqrt{d}} (\mathbf{q}_i^\ell \cdot \mathbf{k}_j^\ell) + \mathbf{b}_{x_0^{(j)} - x_0^{(i)}}^{(2D_x)} + \mathbf{b}_{y_1^{(j)} - y_1^{(i)}}^{(2D_y)} \\ & + \mathbf{b}_{x_1^{(j)} - x_0^{(i)}}^{(2D_x)} + \mathbf{b}_{x_1^{(j)} - x_1^{(i)}}^{(2D_x)} + \mathbf{b}_{\delta_{ij}^l}^{(l)} + \mathbf{b}_{\delta_{ij}^c}^{(c)} \end{aligned} \quad (6.2)$$

6.1.2.2 Learning Position Embeddings from Layout Information

Given a sequence of layout embeddings derived from token bounding box coordinates, as defined by Equation 6.1, Layout2Pos employs a stack of Transformer layers to contextualize the sequence. The outputs of the last layer, $\bar{\ell}_i$, serve as position embeddings, *i.e.*, $\mathbf{p}_i = \bar{\ell}_i$. The objective is for these embeddings $(\mathbf{p}_1, \dots, \mathbf{p}_n)$ to carry information regarding the reading order. To accomplish this, we build a simple classifier on top of these embeddings, designed to compute alignment scores between each token:

$$A_{ij} = (\mathbf{p}_i \mathbf{W}^q) (\mathbf{p}_j \mathbf{W}^k). \quad (6.3)$$

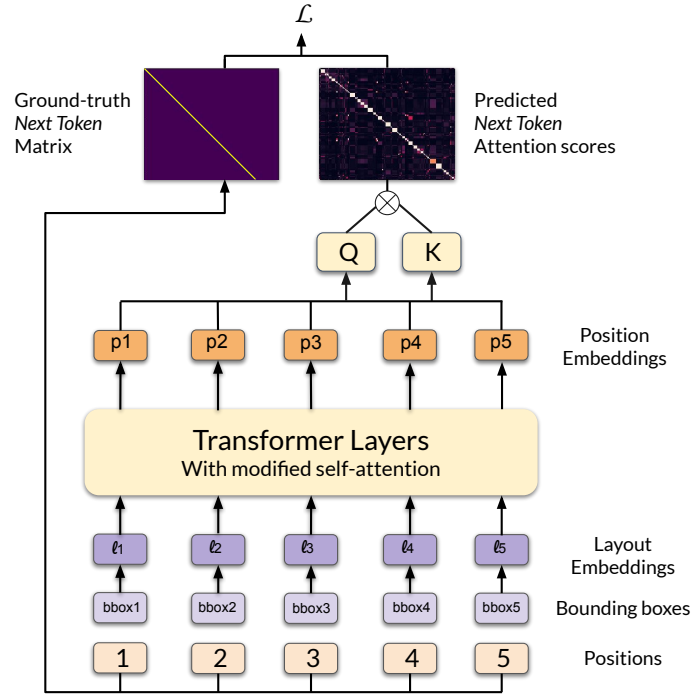


Figure 6.5 – Layout2Pos Architecture.

To learn the relationship between layout and reading order, we employ the same strategy as ERNIE-Layout (Peng et al. 2022), which consists in learning to predict whether two tokens are consecutive (Section 4.3.3). We assume that the attention matrix \mathbf{A} carries information about the reading order, *i.e.*, A_{ij} represents the probability that the j -th token follows the i -th token. Let N denote the ground-truth binary matrix obtained from the ground-truth reading order, where N_{ij} equals 1 if token at position j is the *next* token in the sequence after token at position i , and 0 otherwise. We define the *Next Token Position Prediction* strategy, which consists in using the attention matrix \mathbf{A} to predict the next token of each token in the sequence (*next token matrix*). The corresponding cross-entropy loss is defined as follows:

$$\mathcal{L}_{NTPP} = -\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n N_{ij} \log(\text{softmax}_i(\mathbf{A}_{.j})) \quad (6.4)$$

As such, Layout2Pos can be trained to capture the relationship between layout and reading order³ by ensuring that the attention matrix \mathbf{A} derived from the computed position embeddings (p_1, \dots, p_n) carries information about the next

3. It is noteworthy that a global reading order is unnecessary; there is no requirement to establish an order between two words that belong to segments that have no relation to each other.

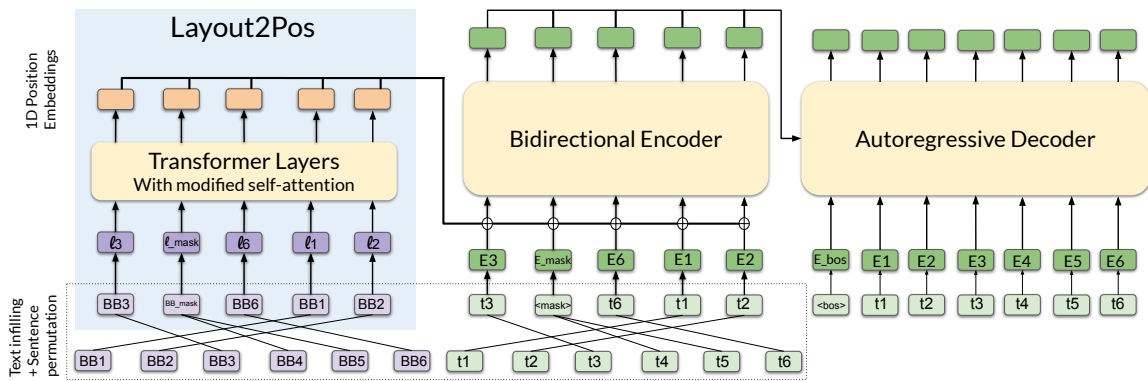


Figure 6.6 – Architecture of Layout2Pos integrated into a BART model, *i.e.*, BART+Layout2Pos. The input consists of two components: a sequence of tokens (middle) and a sequence of token bounding box coordinates (left).

token for each token in the sequence. The architecture of Layout2Pos is depicted in Figure 6.5.

6.1.2.3 Integrating Layout2Pos into a Sequence-to-Sequence Framework

Layout2Pos can be integrated into any language model, removing the reliance on sequential position information. This is achieved by substituting the traditional position encodings derived from OCR by Layout2Pos’ position embeddings. Specifically, we integrate Layout2Pos into a Transformer encoder-decoder architecture, as illustrated in Figure 6.6. The model takes as input a sequence of tokens and their associated bounding boxes, both embedded using embedding tables (Section 3.2.1.1). The sequence of position embeddings, obtained by Layout2Pos, is added to the sequence of token embeddings. The resulting sequence is input to the bidirectional encoder. The output sequence of contextualized embeddings is fed to the autoregressive decoder to generate the target sequence.

Corruption Loss Layout2Pos is trained together with the encoder-decoder model. While the module learns to predict the subsequent token of each token based on layout information, the encoder-decoder follows a pre-training approach similar to Bidirectional and Auto-Regressive Transformers (BART) (M. Lewis et al. 2019). The model is trained to reconstruct the original input sequence from a corrupted version (*denoising*). Sequences are corrupted by randomly replacing text spans with a single mask token (*text infilling*) and permuting sentences (*sequence permutation*). The corrupted sequence is encoded using the bidirectional encoder, and the autoregressive decoder is trained to reconstruct the original sequence. The final loss is expressed as follows:

$$\mathcal{L} = \mathcal{L}_{NTPP} + \mathcal{L}_{Denoising}. \quad (6.5)$$

We refer to the overall model as BART+Layout2Pos.

Inference for Information Extraction Tasks To determine how the model predicts the next token of the sequence for information extraction tasks, we employ a customized variant of beam search to generate tokens while minimizing repetitions, therefore enhancing the coherence of the generated sequences. In this modified version, the generated tokens, if present in the source sequence, are constrained not to occur more frequently than in the original source. This constraint is enforced by keeping count of the number of occurrences of each token in the source sequence within the target sequence, masking the corresponding logit when the maximum occurrence is reached and redistributing the probability mass over the valid tokens.

6.2 Experiments

In this section, we provide an overview of the datasets used for pre-training our models and conducting visual information extraction tasks. Furthermore, we provide details on the experimental setup, covering baselines, pre-training methodologies, and fine-tuning protocols. For reproducibility purposes, we make the models' implementation, along with the fine-tuning and evaluation scripts, publicly available.

6.2.1 Data

6.2.1.1 Pre-training Data

Following a common practice in the field of Document Understanding, we collect data from the IIT-CDIP collection (D. Lewis et al. 2006) to build our pre-training dataset (Section 4.3.3). IIT-CDIP consists of around 11 million document page images of various types and layouts, including news articles, scientific reports, handwritten materials, and more. The collection contains scanned images of documents, introducing challenges related to image quality, resolution, and potential artifacts. As such, we leverage IIT-CDIP to pre-train models under realistic conditions. We select over 7 million document images from the collection to build our pre-training dataset, allocating over 18k for validation, another 18k for testing, and the remaining images for training. To extract text and bound-

ing boxes from the documents, we use DocTR (Mindee 2021). Due to potential serialization errors induced by DocTR, and given that the Next Token Position Prediction task requires documents with proper reading orders, IIT-CDIP is only used for training models in language modeling tasks (*i.e.*, denoising).

To enable Layout2Pos to effectively learn the correct reading order of documents, we use the 500k documents from ReadingBank (Zilong Wang et al. 2021). These documents are serialized and annotated with high-quality reading order annotations, serving as the training data for both Next Token Position Prediction and language modeling tasks.

Our final pre-training corpus is referred to as IIT-CDIP+ReadingBank. While permutation is used for both datasets, text infilling is not applied to ReadingBank. This choice is made to prevent potential alignment issues between Next Token Position Prediction and language modeling, which could arise from replacing spans of tokens with a single mask token.

Note that, in cases where a word is split into multiple tokens, earlier approaches based on word-level bounding boxes typically assign the word’s bounding box to all the tokens within that word. However, we experimentally found that this approach is inefficient for Next Token Position Prediction, given that tokens within the same word would share identical layout embeddings, hence hindering accurate predictions of the next token. Therefore, we approximate token-level bounding boxes by dividing each word-level bounding box by the number of characters in the word.

6.2.1.2 Data for Visual Information Extraction

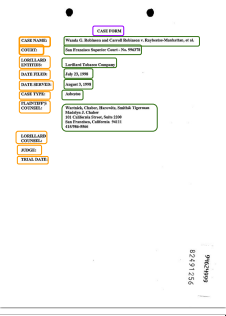
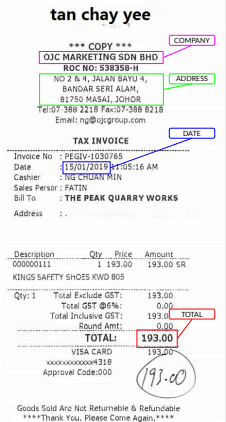
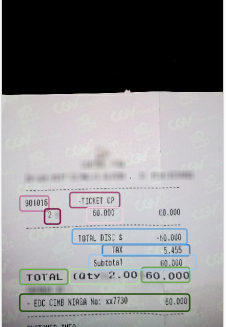
| Dataset | Document | Targets |
|---------|---|---|
| FUNSD |  | <p>COURT: QUESTION</p> <p>JUDGE: QUESTION</p> <p>Address: ANSWER</p> <p>CASE FORM: HEADER</p> <p>CASE NAME: QUESTION</p> <p>LORILLARD ENTITIES: QUESTION</p> <p>DATE FILED: QUESTION</p> <p>DATE SERVED: QUESTION</p> <p>CASE TYPE: QUESTION</p> <p>PLAINTIFF COUNSEL: QUESTION</p> <p>LORILLARD COUNSEL: QUESTION</p> <p>TRIAL DATE: QUESTION</p> <p>Warwick, Chaber, Harowitz, Smith & Tigerman Madelyn J. Chaber 101 California Street, Suite 2200 San Francisco, California 94111 415 986-5566 : ANSWER</p> <p>August 3, 1998 : ANSWER</p> <p>July 23, 1998 : ANSWER</p> <p>Lorillard Tobacco Company : ANSWER</p> <p>San Francisco Superior Court - No. 996378 : ANSWER</p> <p>Wanda G. Robinson and Carroll Robinson v Raybestos-Manhattan, et al. : ANSWER</p> |
| SROIE |  | <p>OJC MARKETING SDN BHD : COMPANY</p> <p>NO 2 & 4, JALAN BAYU 4, BANDAR SERI ALAM, 81750 MASAI, JOHOR : ADDRESS</p> <p>15/01/2019 : DATE</p> <p>193.00 : TOTAL</p> |
| CORD |  | <p>TAX \$ 455 : SUB_TOTAL TAX_PRICE</p> <p>TOTAL 60.000 : TOTAL TOTAL_PRICE</p> <p>Qty 2.00 : TOTAL MENUQTY_CNT</p> <p>EDC CIMB NAGA No: xx7730 60.000 : TOTAL CREDITCARDPRICE</p> <p>601016 : MENU NUM</p> <p>TICKET CP : MENU NUM</p> <p>2 : MENU CNT</p> <p>60.000 : MENU PRICE</p> <p>Subtotal 60.000 : SUB_TOTAL SUBTOTAL_PRICE</p> <p>TOTAL DISC \$ 40.000 : SUB_TOTAL DISCOUNT_PRICE</p> |

Table 6.2 – Example document from FUNSD, SROIE, and CORD, accompanied by their corresponding target sequences that include the entities to be extracted paired with their corresponding keys. Best viewed in color.

We evaluate our approach on visual information extraction tasks, where the goal is to extract semantic entities from visually-rich documents, based on a set of pre-defined keys. This evaluation is conducted using three benchmark datasets for visual information extraction (Section 4.1.3), each covering different document types: FUNSD (Jaume et al. 2019), SROIE (Zheng Huang et al. 2019), and CORD (Park et al. 2019). For each dataset, we use the reading order provided. To maintain consistency with pre-training data, we employ approximated token-level

bounding boxes. In Figure 6.2, we provide an example of source documents from FUNSD, SROIE, and CORD, along with their corresponding target sequences.

FUNSD (Jaume et al. 2019) is a form understanding dataset consisting of 199 real, noisy and scanned forms where each sample is a list of form entities. There are three keys for which values have to be extracted: *question*, *answer*, and *header*. The dataset is split into 149 samples for training and 50 for test.

SROIE (Zheng Huang et al. 2019) is a receipt understanding dataset comprising 973 scanned receipts written in English. The task involves extracting entities for four keys: *total*, *date*, *company*, and *address*. The dataset is partitioned into 626 samples for training and 347 for test.

CORD (v1) (Park et al. 2019) is another receipt understanding dataset containing 1,000 scanned Indonesian receipts with 30 keys categorized into four superclasses: *menu*, *subtotal*, *total*, and *void*. Following the katanaml/cord⁴ dataset repository, we exclude keys with very few occurrences, resulting in 22 keys grouped into three superclasses. The dataset is divided into 800 examples for training, 100 for validation, and 100 for test.

6.2.2 Experimental Settings

Models were implemented in Python using PyTorch (Paszke et al. 2017) and Hugging Face (Wolf et al. 2019) libraries.

6.2.2.1 Baselines

We compare our approach with BART+2D, a layout-augmented BART model which relies on position embeddings derived from OCR-induced positions. These position embeddings are calculated using embedding tables (Devlin et al. 2018) and are subsequently added to textual features. Layout embeddings, computed from bounding boxes using Equation 6.1, are incorporated to the resulting embeddings to construct the input embeddings. Following LayoutLMv2, BART+2D encodes spatial relative positions as bias terms added to the attention scores:

$$\alpha_{i,j} = \frac{1}{\sqrt{d}} \mathbf{q}_i \cdot \mathbf{k}_j + \mathbf{b}_{j-i}^{(1D)} + \mathbf{b}_{x_0^{(j)}-x_0^{(i)}}^{(2D_x)} + \mathbf{b}_{y_1^{(j)}-y_1^{(i)}}^{(2D_y)}, \quad (6.6)$$

4. <https://huggingface.co/datasets/katanaml/cord>

where $\mathbf{b}^{(1D)}$, $\mathbf{b}^{(2D_x)}$, and $\mathbf{b}^{(2D_y)}$ denote the sequential, horizontal, and vertical relative position biases, respectively. BART+2D follows the same training and inference procedures as BART+Layout2Pos.

Additionally, we report the performance of two layout-aware encoder-only models: 1) LayoutLM and 2) LayoutLMv2-no-visual, a variant of LayoutLMv2 that discards visual information to ensure a fair comparison with our approach. In the datasets used for visual information extraction, the assumption that each identified segment of an entity constitutes a contiguous sequence of words within the input holds. This simplifies the learning process for sequence labeling methods, as the order of tokens in the input aligns with the entities to identify, facilitating the association of tokens with their specific label. Consequently, sequence-labeling approaches might be seen as having an unfair advantage.

6.2.2.2 Pre-training

Encoder-decoder Models Layout2Pos is composed of 2 layers with 12 attention heads and a hidden size of 768 (a choice we validate in Section 6.3.1). The final attention calculation, responsible for computing the next token matrix, involves a single attention head. Following the BART base model, both the encoder and decoder in BART+Layout2Pos and BART+2D are comprised of 6 layers, each with 12 attention heads and a representation space dimensionality of 768. BART+Layout2Pos comprises a total of 156M parameters, whereas BART+2D consists of approximately 140M parameters, making them roughly equivalent in size.

Both models are trained from scratch on IIT-CDIP+ReadingBank. The documents are tokenized using the tokenizer of the base variant of BART (bart-base) shared through the Hugging Face Model Hub. The training spans 10 epochs, amounting to 500k optimization steps, including 59k steps for warmup. For each model, we select the checkpoint with the best validation loss. We use a maximum sequence length of 512, a batch size of 80, and a learning rate of $1e^{-4}$. Following BART, we mask 30% of tokens in each sequence (with span lengths drawn from a Poisson distribution where $\lambda = 3$) and permute all sentences. Experiments were ran using Nvidia Titan RTX with 25GB.

Encoder-only baselines For LayoutLM, we use the microsoft/layoutlm-base-uncased checkpoint with 113M parameters, without any additional pre-training. Following the base architecture of LayoutLMv2, LayoutLMv2-no-visual is composed of a 12-layer Transformer encoder with 12 attention heads and a hidden size of 768, amounting to 110M parameters. The model is also pre-trained from scratch for 10 epochs on IIT-CDIP+ReadingBank, using Masked Visual-Language Model-

ing (MVLM), a pre-training task that extends Masked Language Modeling (MLM) with layout information. The documents are tokenized using the tokenizer of microsoft/layoutlm-base-uncased. We use a maximum sequence length of 512, a batch size of 80, and a learning rate of $1e^{-4}$.

6.2.2.3 Visual Information Extraction

Sequence-labeling Approaches To compare our sequence-to-sequence approach with the traditional sequence labeling method on visual information extraction tasks, we employ LayoutLM and LayoutLMv2-no-visual. We use the BIO (Beginning, Inside, Outside) tagging format (Ramshaw and M. P. Marcus 1999) as the labeling scheme to tag tokens based on both their entity and their position within that entity. For every dataset, the maximum sequence length is set to 512. Both models are fine-tuned for 100 epochs on FUNSD, and 20 epochs on SROIE and CORD. The learning rate is set to $5e^{-5}$ for all models and datasets.

Sequence-to-sequence Models We frame visual information extraction as a sequence-to-sequence problem, wherein the document serves as the input, and the output consists of a series of extracted entities paired with their corresponding keys. For all three datasets, we set the maximum source sequence length to 512. Documents that exceed this length are split into contiguous sequences of 512 tokens each. For each input sequence, we formulate a target sequence containing the pairs of entities-keys to be extracted from the input sequence. The structure of the target sequences is defined such that each entity is followed by a colon and its corresponding key, with pairs separated by a line break. The arrangement of the pairs aligns with the order in which the corresponding entities appear in the document, *i.e.*, the provided reading order.

Entities and their corresponding keys are extracted from both the generated sequences and the ground-truth sequences. The pairs of generated and ground-truth (*key, entities*) are then compared to compute precision, recall, and F1 score. To provide further insights into a sequence-to-sequence model's errors, additional metrics are defined. To measure how often the model produces content that is not grounded in the input, the *hallucination rate* is defined as the percentage of entities generated by the model that do not match with any text in the input sequence. The *repetition rate* is the percentage of generated entities that are part of the ground-truth entities but are repeated more frequently than their occurrences in the ground-truth target sequence, quantifying the frequency with which the model repeats entities. The *wrong label rate* represents the proportion of generated entities present in the ground-truth but mislabeled by the model, and measures how often the model generates the right entities but mislabels them. The *omission*

rate denotes the proportion of ground-truth entities that were not generated by the model, providing insights into how often the model omits entities. Lastly, the *non-entity rate* is the percentage of generated entities that, in the ground-truth, correspond to the category "Other". This metric assesses the frequency with which the model categorizes a text as an entity when it should not be considered as such (discarding hallucinations).

We compute statistics on the lengths of target sequences and establish the maximum target length to be greater than the 3rd quartile. In the case of FUNSD, we truncate target sequences at 768 tokens. As for SROIE and CORD, the maximum target sequence length is set to 96 and 512 tokens, respectively. BART+Layout2Pos (BART+2D) is fine-tuned for 100 (100), 40 (40), and 20 (50) epochs on FUNSD, SROIE, and CORD, respectively. The learning rate is set to $5e^{-5}$ for all models and datasets. During inference, we set the number of beams to 8. Precision, recall, and F1 scores are computed using the seqeval package (Nakayama 2018).

Considering that the pre-training phase has allowed BART+Layout2Pos to compute meaningful position embeddings, and acknowledging the potential discrepancies in the reading order within visual information extraction datasets, we opt not to use the Next Token Position Prediction strategy during fine-tuning.

6.3 Results and Discussion

We first provide an ablation study of Layout2Pos on the Next Token Position Prediction task, before discussing the results obtained on visual information extraction.

6.3.1 Next Token Position Prediction

| Number of Layers | Pre-training Dataset | Accuracy |
|------------------|----------------------|---------------|
| 1 | IIT-CDIP | 67.10% |
| 1 | ReadingBank | 89.37% |
| 2 | ReadingBank | 95.86% |

Table 6.3 – Accuracy in predicting the next token for pairs sourced from ReadingBank, which were not used for pre-training. Selected pairs are considered "difficult", meaning that the tokens are positioned on different lines.

We first evaluate the performance of the Layout2Pos module by computing the accuracy of Next Token Position Prediction. Recalling that the IIT-CDIP dataset is not considered reliable, this evaluation is conducted on a set of pairs of consecutive tokens derived from 100 examples from ReadingBank, which were not used in the pre-training phase. Specifically, we curated pairs categorized as "difficult", where the tokens are positioned on different lines, making a raster-scan approach ineffective. This choice demands the model to leverage layout information to accurately predict the next token in these scenarios.

In these experiments, we exclusively train and evaluate Layout2Pos, omitting the encoder-decoder architecture. For each token, we compute accuracy by comparing the position of its subsequent token with the position of the token associated with the highest logit according to Layout2Pos. We vary the number of layers and the pre-training dataset used.

Performance is reported in Table 6.3. Notably, pre-training Layout2Pos on ReadingBank compared to IIT-CDIP yields an increase of over 22% in accuracy. Additionally, augmenting the number of layers in Layout2Pos results in a notable increase of over 6% in accuracy, reaching an accuracy score of 95% for challenging pairs. These results highlight the significance of using documents with accurate reading orders and contextualizing layout information to produce position embeddings able to capture the reading order of documents.

6.3.2 Visual Information Extraction

Table 6.4 reports the performance of all four models on FUNSD, SROIE, and CORD. For each dataset, the additional rates are calculated for each document and averaged across all documents. We find that our sequence-to-sequence models achieve performance that is comparable or even superior to sequence-labeling approaches. This suggests that the sequence-to-sequence approach can match the effectiveness of traditional sequence labeling methods, offering an alternative that is not constrained by the document's content.

On SROIE, BART+Layout2Pos performs on par with its counterpart fed with sequential position information—BART+2D. This suggests that Layout2Pos effectively leverages layout information to generate meaningful position embeddings on SROIE, implying that the reading order provided by OCR is no longer necessary. However, on the other two datasets, BART+Layout2Pos demonstrates lower performance than BART+2D, with a slight underperformance on CORD and a more notable disparity on FUNSD.

Additionally, we find that the majority of errors arise from either omitted or mislabeled entities. On FUNSD, for BART+2D (BART+Layout2Pos), mislabeled

| Dataset | Reading Order | Model | Rate | | | | | | | |
|---------|---------------|----------------------------------|-------|-------|--------------|------------|---------------|-------------|----------|------------|
| | | | Prec. | Rec. | F1 | Repetition | Hallucination | Wrong Label | Omission | Non-entity |
| FUNSD | Original | LayoutLM (Yiheng Xu et al. 2020) | 75.91 | 80.54 | 78.16 | | | | | |
| | | LayoutLMv2-no-visual | 78.58 | 81.49 | 80.01 | | | | | |
| | | BART+2D | 83.74 | 86.55 | 85.12 | 2.67 | 1.32 | 45.76 | 39.06 | 1.19 |
| | | BART+Layout2Pos | 80.62 | 80.10 | 80.36 | 2.56 | 5.50 | 22.88 | 57.11 | 3.96 |
| | Shuffled | BART+2D | 77.82 | 82.16 | 79.93 | 2.89 | 2.15 | 48.37 | 34.68 | 3.25 |
| | | BART+Layout2Pos | 80.84 | 80.98 | 81.13 | 2.37 | 5.31 | 22.30 | 58.05 | 3.97 |
| SROIE | Original | LayoutLM (Yiheng Xu et al. 2020) | 90.74 | 93.95 | 92.32 | | | | | |
| | | LayoutLMv2-no-visual | 93.20 | 93.88 | 93.54 | | | | | |
| | | BART+2D | 93.46 | 93.73 | 93.60 | 0.00 | 0.29 | 0.00 | 18.11 | 2.64 |
| | | BART+Layout2Pos | 93.20 | 93.80 | 93.50 | 0.00 | 0.58 | 0.29 | 17.03 | 3.43 |
| | Shuffled | BART+2D | 80.58 | 66.33 | 73.13 | 2.66 | 1.46 | 0.41 | 73.34 | 5.72 |
| | | BART+Layout2Pos | 93.13 | 93.80 | 93.45 | 0.0 | 0.58 | 0.29 | 17.03 | 3.43 |
| CORD | Original | LayoutLM (Yiheng Xu et al. 2020) | 93.91 | 95.11 | 94.51 | | | | | |
| | | LayoutLMv2-no-visual | 93.14 | 94.89 | 94.00 | | | | | |
| | | BART+2D | 95.97 | 94.81 | 95.39 | 2.33 | 0.00 | 5.28 | 19.06 | 0.33 |
| | | BART+Layout2Pos | 94.56 | 92.71 | 93.62 | 0.99 | 4.37 | 5.40 | 22.83 | 0.40 |
| | Shuffled | BART+2D | 91.46 | 87.54 | 89.46 | 4.53 | 0.83 | 26.5 | 35.72 | 0.42 |
| | | BART+Layout2Pos | 94.45 | 92.61 | 93.51 | 1.10 | 4.37 | 5.38 | 22.75 | 0.40 |

Table 6.4 – Model performance (in %) on FUNSD, SROIE, and CORD, reported for 1) the original reading order and 2) three shuffled orders (averaged). Best F1 scores for each dataset/reading order are reported in bold.

entities account for an average of 45.76% (22.88%) of errors in a document, while 39.06% (57.11%) of errors are on average attributed to omissions. Overall, both models rarely hallucinate (on FUNSD, 1.32% and 5.50% of errors made by BART+2D and BART+Layout2Pos, respectively, are attributed to hallucinations), repeat entities (2.67% and 2.56%), or identify a text as an entity when it should not be considered as such (1.19% and 3.96%). This is also a result of our constrained decoding approach.

To measure the impact of reading order on models dependent on it, we evaluate BART+2D and BART+Layout2Pos on test documents with shuffled reading orders. For every test dataset, the reading order of each document is shuffled such that words belonging to the same entity remain grouped together. This process is repeated three times, generating three shuffled test sets for every original test dataset. BART+2D and BART+Layout2Pos, fine-tuned using the reading order provided by the dataset, are then evaluated on each of the shuffled test sets. The resulting scores are then averaged and reported in Table 6.4. Results show that altering the reading order, even while ensuring that words belonging to the same entities are kept together, leads to a significant performance decline for BART+2D. Specifically, there is a F1-score drop of 5.19 and 5.93 for FUNSD and

CORD, respectively, and a notable decrease of 20.84 for SROIE. In contrast, such variations have no effect on BART+Layout2Pos, as the model is not reliant on reading order.⁵ This highlights the significance of developing methods robust to variations in reading order.

6.4 Conclusion

To derive position embeddings solely from layout information and avoid reading order issues, we propose Layout2Pos—a Transformer-based module that learns the sequential relationships between tokens in a document. We conduct experiments on three benchmarks datasets for visual information extraction, demonstrating the effectiveness of our approach in leveraging layout information to produce meaningful position embeddings. Furthermore, we showcase the significant impact of variations in reading order on models that rely on sequential position information, encouraging research on reading order-independent methods for document understanding tasks. In addition, we introduce novel metrics that offer additional insights into the performance of sequence-to-sequence models for visual information extraction.

Acknowledging the limitations of our study, our sequence-to-sequence evaluation framework considers any arrangement of key-value pairs as valid. However, language models trained with teacher forcing tend to favor a single correct output, potentially penalizing valid responses with different entity orders. For future work, we will investigate permutation invariant losses to foster robustness to variation in entity orders. Furthermore, our current model evaluations are limited by their focus on relatively simple datasets and documents of shorter length. Additionally, our analyses have been confined to English language texts. Recognizing these limitations, generalizability could be enhanced by including more complex datasets, particularly those featuring longer documents (Graliński et al. 2020).

5. While we do observe marginal variations for BART+Layout2Pos, we attribute these differences to the fact that documents longer than the maximum sequence length may be split into sequences different from those obtained with the original reading order.

LEVERAGING LAYOUT TO DEAL WITH LONG AND LAYOUT-RICH DOCUMENTS

Chapter abstract

Building upon the groundwork laid in Chapter 5, which introduced a layout-based self-attention mechanism employed for Document Layout Analysis, and Chapter 6, which presented a layout-based Transformer model applied to Visual Information Extraction, this chapter further advances the exploration by investigating how to leverage layout when processing long documents, specifically in the context of Long Text Summarization. Text Summarization is a popular task and an active area of research for the Natural Language Processing (NLP) community. It requires accounting for long input texts, a characteristic which poses computational challenges for neural models. Moreover, real-world documents come in a variety of complex, visually-rich, layouts. This information is of great relevance, whether to highlight salient content or to encode long-range interactions between textual passages. Yet, all publicly available summarization datasets only provide plain text content. To facilitate research on how to exploit visual/layout information to better capture long-range dependencies in summarization models, we present LoRaLay, a collection of datasets for long-range summarization with accompanying visual/layout information. We extend existing and popular English datasets (arXiv and PubMed) with visual/layout information and propose four novel datasets—consistently built from scholar resources—covering French, Spanish, Portuguese, and Korean languages. Further, we propose new baselines merging layout-aware and long-range models—two orthogonal approaches—and obtain state-of-the-art results, showing the importance of combining both lines of research. In addition, we develop an annotation interface for human evaluation of summaries and introduce a novel metric to provide insights into the preservation of information flow in generated summaries.

The work in this chapter has led to the publication of a conference paper:

- Laura Nguyen, Thomas Scialom, Benjamin Piwowarski, and Jacopo Staiano (May 2023). “LoRaLay: A Multilingual and Multimodal Dataset for

Long Range and Layout-Aware Summarization". In: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Dubrovnik, Croatia: Association for Computational Linguistics, pp. 636–651. URL: <https://aclanthology.org/2023.eacl-main.46>.

- Award: EACL Outstanding Paper.

Contents

| | | |
|-------|----------------------------------|-----|
| 7.1 | Datasets Construction | 129 |
| 7.1.1 | Collecting the Data | 130 |
| 7.1.2 | Data Pre-processing | 132 |
| 7.1.3 | Datasets Statistics | 134 |
| 7.2 | Experiments | 136 |
| 7.2.1 | Models | 136 |
| 7.2.2 | Implementation Details | 137 |
| 7.3 | Results and Discussion | 138 |
| 7.3.1 | General Results | 138 |
| 7.3.2 | Human Evaluation | 141 |
| 7.3.3 | Case Studies | 145 |
| 7.4 | Conclusion | 145 |

Long document understanding presents several significant challenges. Firstly, contemporary document understanding approaches, based on the Transformer architecture (Vaswani et al. 2017), suffer from the quadratic complexity of self-attention, as discussed in Chapter 3. Furthermore, integrating layout information increases resource demands compared to models that exclusively deal with text. This challenge hinders the ability of pre-trained document understanding models to capture long-range dependencies, constraining their use to short sequences. Dividing long documents into shorter segments which are independently processed is a straightforward solution, but it falls short for documents where crucial information is distributed across their entire length.

Chapter 5 showcased the potential of leveraging layout information to reduce the complexity of Transformers, emphasizing the significance of layout in facilitating comprehension of information conveyed in long documents. Therefore, processing long documents requires the ability to effectively and efficiently model layout information. To connect information across pages, Pramanik et al. (2020) encode the page number into token representations and use the Longformer architecture (Beltagy et al. 2020) to process long documents. Inspired by Longformer, H. Pham et al. (2022) introduce spatial-based attention masks to restrict each token’s attention to its neighbors in the 2D page. Despite advancements in tackling the challenges related to long documents, efficiently leveraging layout information to process long documents remains an unresolved and under-explored problem.

This contribution aims at spurring further research on how to incorporate multi-modal information to better capture long-range dependencies. To develop models capable of exploiting multimodal information from long documents, there is a need for tasks and datasets that cover every modality. Focusing on compressing

the most relevant information from long texts to short summaries, the Text Summarization task naturally lends itself to benefit from a global context. Notice that, in practice, the limitations linked to sequence length are also amplified by the lack of visual/layout information in the existing datasets.

Yet, the vast majority of publicly available summarization datasets only provide plain text content. Hermann et al. (2015) proposed the CNN/DailyMail dataset, a collection of English articles extracted from the CNN and The Daily Mail portals. Each news article is associated with multi-sentence highlights which serve as reference summaries. Scialom et al. (2020) bridge the gap between English and non-English resources for text summarization by introducing MLSum, a large-scale multilingual summarization corpus providing news articles written in French, German, Spanish, Turkish and Russian. Going towards more challenging scenarios involving significantly longer documents and aiming to encourage a shift towards building more abstractive summarization models with global content understanding, Sharma et al. (2019) introduce BIGPATENT. This large-scale dataset comprises U.S. patent filings, where invention descriptions serve as reference summaries.

Guidelines for manually summarizing texts—especially long ones—often recommend roughly previewing them to break them down into their major sections (Toprak and Almacioğlu 2009; L. Luo et al. 2019). Recognizing the significance of document structures, such as sections and paragraphs, in guiding summary generation, Cohan et al. (2018) introduce the arXiv and PubMed datasets. These datasets consist of scientific articles collected from academic repositories, with discourse information (*i.e.*, sections), where the paper abstracts are used as summaries. Leveraging these section structures, Cohan et al. (2018) introduce a section-level encoder based on the output of a word-level encoder for long document summarization. Similarly, S. Cao and L. Wang (2022) leverage section levels to build a document structure tree. To account for the relative positions of tokens within the document structure, learnable hierarchical biases, derived from the distance in the structure tree between the corresponding sections, are added to the attention scores. In addition, S. Cao and L. Wang (2022) curate a new dataset for long and structure-aware document summarization. This dataset comprises 21k documents written in English, sourced from WikiProject Biography, and includes section structures. However, not all documents are explicitly organized into clearly defined sections, and extracting discourse structure may not be straightforward, especially in cases where only PDF files or document images are available.

Although not every document is explicitly arranged into well-defined sections, the great majority contains layout and visual clues (*e.g.*, a physical organization into paragraphs, bigger headings/subheadings) which help structure their textual contents and facilitate reading for humans. Therefore, we argue that layout might

be crucial to summarize long documents. To investigate this hypothesis, we construct *LoRaLay*, a collection of datasets designed for *long-range* and *layout-aware* summarization. LoRaLay is a large-scale corpus of research papers that extends two popular datasets, arXiv and PubMed, with layout and visual information, and introduces 4 novel datasets covering French, Spanish, Portuguese and Korean languages. Finally, we compare the performance of Transformer-based models on LoRaLay, and show that combining long-range and layout-aware models results in enhanced performance for long document summarization.

In this chapter, we first detail the dataset construction process. Then, we introduce novel long-range and layout-aware baselines and offer a detailed explanation of the experimental setup. Lastly, we demonstrate the importance of combining layout-aware and long-range modeling through qualitative and quantitative evaluations.

7.1 Datasets Construction

Inspired by the way the arXiv and PubMed datasets were built (Cohan et al. 2018), we construct our corpus from research papers, with abstracts as ground-truth summaries. As the PDF format allows simultaneous access to textual, visual and layout information, we collect PDF files to construct our datasets, and provide their URLs.¹

For each language, we select a repository that contains a high number of academic articles (in the order of hundreds of thousands) and provides easy access to abstracts. More precisely, we chose the following repositories:

- Archives Ouverte HAL (French),² an open archive of scholarly documents from all academic fields. As HAL is primarily directed towards French academics, a great proportion of articles are written in French;
- SciELO (Spanish and Portuguese),³ an open access database of academic articles published in journal collections from Latin America, Iberian Peninsula and South Africa, and covering a broad range of topics (*e.g.*, agricultural sciences, engineering, health sciences, letters and arts). Languages include English, Spanish, and Portuguese.

1. We make the corpus-construction code publicly available at <https://github.com/recitalAI/loralay-datasets>.

2. <https://hal.archives-ouvertes.fr/>

3. <https://www.scielo.org/>

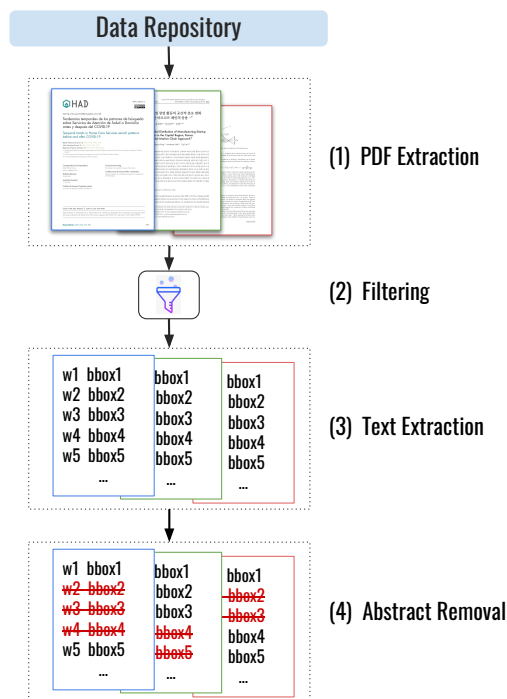


Figure 7.1 – Dataset Construction Process.

- KoreaScience (Korean),⁴ an open archive of Korean scholarly publications in the fields of natural sciences, life sciences, engineering, and humanities and social sciences. Articles are written in English or Korean.

Further, we provide enhanced versions of the arXiv and PubMed datasets, respectively denoted as arXiv-Lay and PubMed-Lay, for which layout information is provided. The dataset construction process is illustrated in Figure 7.1, and is composed of the following stages: (1) PDF Extraction, (2) document filtering, (3) text extraction, and (4) abstract removal.

7.1.1 Collecting the Data

7.1.1.1 Extended Datasets

The arXiv and PubMed datasets (Cohan et al. 2018) contain long scientific research papers extracted from the arXiv and PubMed repositories. We augment them by providing their PDFs, allowing access to layout and visual information. As the abstracts contained in the original datasets are all lowercased, we do not reuse them, but rather extract the raw abstracts using the corresponding APIs.

4. <http://www.koreascience.or.kr>

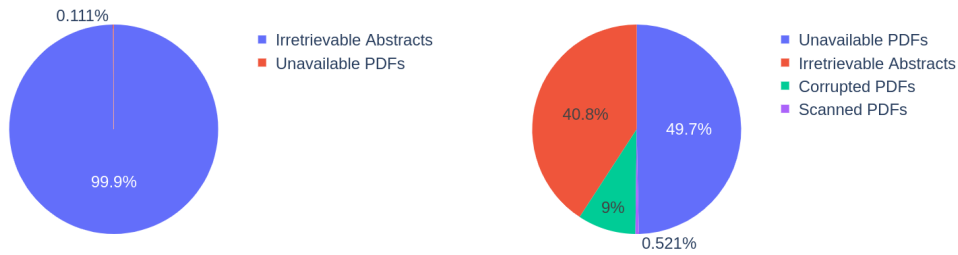


Figure 7.2 – Distribution of failure types in arXiv-Lay (top) and PubMed-Lay (bottom).

Note that we were unable to retrieve all the documents contained in the original datasets. For the most part, we failed to retrieve the corresponding abstracts, as they did not necessarily match the ones contained in the PDF files (due to *e.g.* PDF-parsing errors). We also found that some PDF files were unavailable, while others were corrupted or scanned documents. Figure 7.2 provides details on the amount of original documents lost in the process of augmenting arXiv and PubMed with layout/visual information. We observe four types of failures, and provide numbers for each type:

- The link to the document’s PDF file is not provided (*Unavailable PDF*);
- The PDF file is corrupted (*i.e.*, cannot be opened) (*Corrupted PDF*);
- The document is not digital-born, making it impossible to parse it with PDF parsing tools (*Scanned PDF*);
- The document’s abstract cannot be found in the PDF (*Irretrievable Abstract*).

In total, about 39% (35%) of the documents contained in the original arXiv dataset (PubMed) were lost.

arXiv-Lay The original arXiv dataset (Cohan et al. 2018) was constructed by converting the \LaTeX files to plain text. To be consistent with the other datasets—for which \LaTeX files are not available—we instead use the PDF files to extract both text and layout elements. For each document contained in the original dataset, we fetch (when possible) the corresponding PDF file using Google Cloud Storage buckets. As opposed to the original procedure, we do not remove tables nor discard sections that follow the conclusion. We retrieve the corresponding abstracts from a metadata file provided by Kaggle.⁵

5. <https://www.kaggle.com/Cornell-University/arxiv>

PubMed-Lay For PubMed, we use the PMC OAI Service⁶ to retrieve abstracts and PDF files.

7.1.1.2 New Datasets

HAL We use the HAL API⁷ to download research papers written in French. To avoid excessively long (*e.g.*, theses) or short (*e.g.*, posters) documents, extraction is restricted to journal and conference papers.

SciELO Using Scrapy,⁸ we crawl the following SciELO collections: Ecuador, Colombia, Paraguay, Uruguay, Bolivia, Peru, Portugal, Spain and Brazil. We download documents written either in Spanish or Portuguese, according to the metadata, obtaining two distinct datasets: SciELO-ES (Spanish) and SciELO-PT (Portuguese).

KoreaScience Similarly, we scrape the KoreaScience website to extract research papers. We limit search results to documents whose publishers' names contain the word *Korean*. This rule was designed after sampling documents in the repository, and is the simplest way to get a good proportion of papers written in Korean. We show that this rule does not bias the sample towards a specific research area. We compute the distribution of topics covered by all publishers, and compare it to the distribution of topics covered by publishers whose name contains the word *Korean*. Figure 7.3 shows that the distribution obtained using our rule remains roughly the same as the original. Further, search is restricted to papers published between 2012 and 2021, as recent publications are more likely to have digital-born, searchable PDFs. Finally, we download the PDF files of documents that contain an abstract written in Korean.

7.1.2 Data Pre-processing

For each corpus, we use the 95th percentile of the page distribution as an upper bound to filter out documents with too many pages, while the 5th (1st for HAL and SciELO) percentile of the summary length distribution is used as a minimum threshold to remove documents whose abstracts are too short. As our baselines do not consider visual information, we only extract text and layout from the PDF files. Layout is incorporated by providing the spatial position of each word in a document page image, represented by its bounding box (x_0, y_0, x_1, y_1) ,

6. <https://www.ncbi.nlm.nih.gov/pmc/tools/oai/>

7. <https://api.archives-ouvertes.fr/docs/search>

8. <https://scrapy.org/>

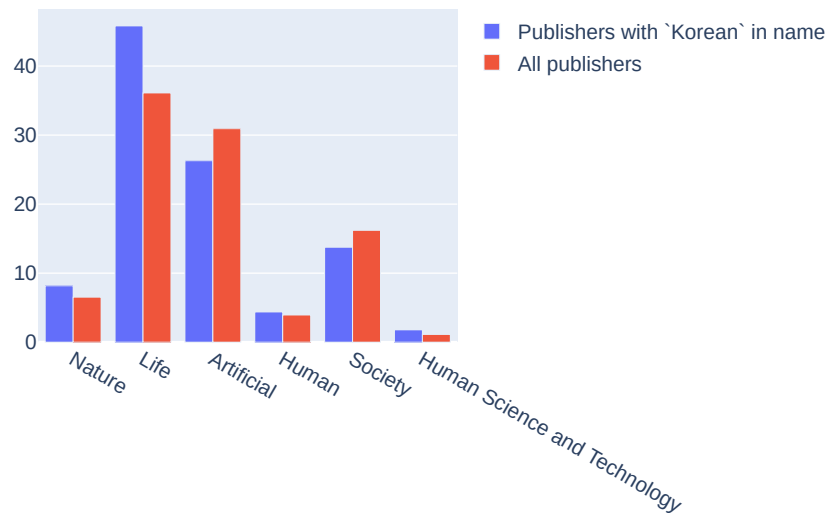


Figure 7.3 – Distribution of topics covered by all publishers (red) vs distribution of topics covered by publishers whose name contains the word *Korean* (blue).

where (x_0, y_0) and (x_1, y_1) respectively denote the coordinates of the top-left and bottom-right corners. Using the PDF rendering library Poppler⁹, text and word bounding boxes are extracted from each PDF, and the sequence order is recovered by Poppler based on heuristics around the document layout (*e.g.*, tables, columns).

Abstracts are then removed by searching for exact matches; when no exact match is found, we use fuzzysearch¹⁰ and regex¹¹ to find near matches. We use a maximum Levenshtein distance of 20 with fuzzysearch, and a maximum number of errors of 3 with regex. For non-English datasets, documents might contain several abstracts, written in different languages. To avoid information leakage, we retrieve the abstract of each document in every language available—according to the API for HAL or the websites for SciELO and KoreaScience—and remove them using the same strategy as for the main language. In the case an abstract cannot be found, we discard the document to prevent any unforeseen leakage.

9. <https://poppler.freedesktop.org/>

10. <https://pypi.org/project/fuzzysearch/>

11. <https://pypi.org/project/regex/>

| Dataset | # Docs | Mean Article Length | Mean Summary Length |
|--------------------------------|-----------|---------------------|---------------------|
| arXiv (Cohan et al. 2018) | 215,913 | 3,016 | 203 |
| PubMed (Cohan et al. 2018) | 133,215 | 4,938 | 220 |
| BigPatent (Sharma et al. 2019) | 1,341,362 | 3,572 | 117 |
| arXiv-Lay | 130,919 | 7,084 | 125 |
| PubMed-Lay | 86,668 | 4,038 | 144 |
| HAL | 46,148 | 4,543 | 134 |
| SciELO-ES | 23,170 | 4,977 | 172 |
| SciELO-PT | 21,563 | 6,853 | 162 |
| KoreaScience | 37,498 | 3,192 | 95 |

Table 7.1 – Datasets statistics. Article and summary lengths are computed in words. For KoreaScience, words are obtained via white-space tokenization. Difference between arXiv and arXiv-Lay is due to the fact that we retain the whole document, while Cohan et al. (2018) truncate it after the conclusion.

7.1.3 Datasets Statistics

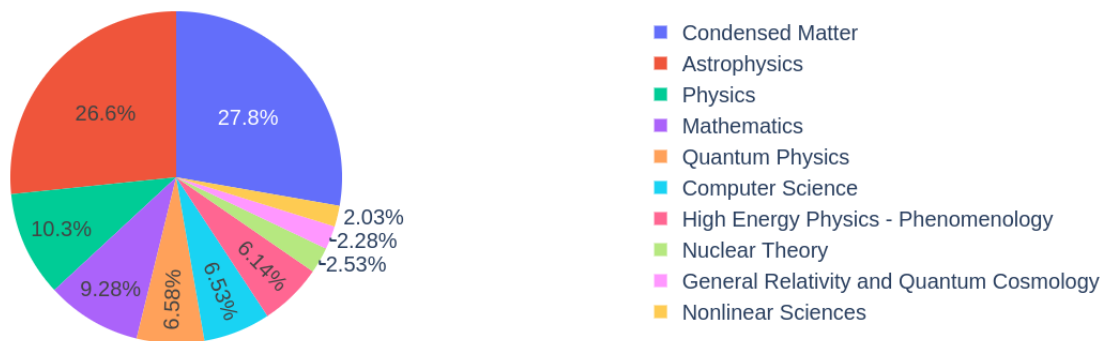
The statistics of our proposed datasets, along with those computed on existing summarization datasets of long documents (Cohan et al. 2018; Sharma et al. 2019) are reported in Table 7.1. We see that document lengths are comparable or greater than for the arXiv, PubMed and BigPatent datasets.

For arXiv-Lay and PubMed-Lay, we retain the original train/validation/splits and try to reconstruct them as faithfully to the originals as possible. For the new datasets, we order documents based on their publication dates and provide splits following a chronological ordering. For HAL and KoreaScience, we retain 3% of the articles as validation data, 3% as test, and the remaining as training data. To match the number of validation/test documents in HAL and KoreaScience, we split the data into 90% for training, 5% for validation and 5% for test, for both SciELO datasets. The statistics of our splits are provided in Table 7.2.

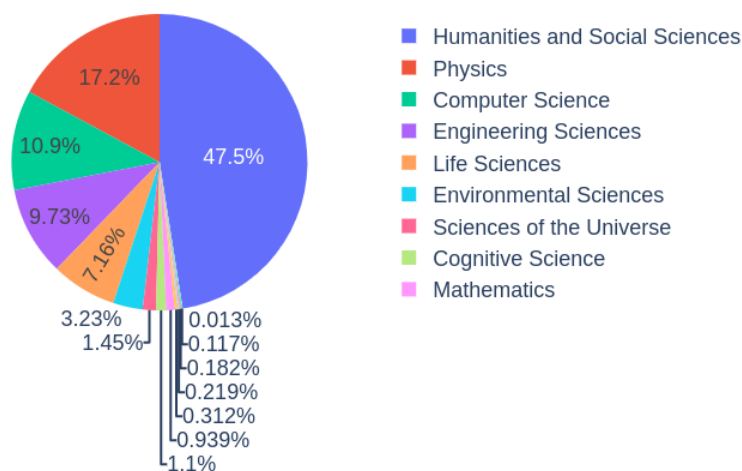
The distribution of research areas in arXiv-Lay and HAL are provided in Figure 7.4. Such distributions are not available for the other datasets, as we did not have access to topic information during extraction.

| Dataset | Instances | | | Input Length | | Output Length | |
|----------------------------|-----------|-------|-------|--------------|---------|---------------|---------|
| | Train | Dev | Test | Median | 90%-ile | Median | 90%-ile |
| arXiv (Cohan et al. 2018) | 203,037 | 6,436 | 6,440 | 6,151 | 14,405 | 171 | 352 |
| PubMed (Cohan et al. 2018) | 119,924 | 6,633 | 6,658 | 2,715 | 6,101 | 212 | 318 |
| arXiv-Lay | 122,189 | 4,374 | 4,356 | 6,225 | 12,541 | 150 | 249 |
| PubMed-Lay | 78,234 | 4,084 | 4,350 | 3,761 | 7,109 | 182 | 296 |
| HAL | 43,379 | 1,384 | 1,385 | 4,074 | 8,761 | 179 | 351 |
| SciELO-ES | 20,853 | 1,158 | 1,159 | 4,859 | 8,519 | 226 | 382 |
| SciELO-PT | 19,407 | 1,078 | 1,078 | 6,090 | 9,655 | 239 | 374 |
| KoreaScience | 35,248 | 1,125 | 1,125 | 2,916 | 5,094 | 219 | 340 |

Table 7.2 – Datasets splits and statistics. Input and output lengths are computed in tokens, obtained using Pegasus and mBART-50’s tokenizers for the English and non-English datasets, respectively.



(a) Distribution of research areas in arXiv-Lay.



(b) Distribution of research areas in HAL.

Figure 7.4 – Distribution of research areas in arXiv-Lay (a) and HAL (b).

7.2 Experiments

7.2.1 Models

We compare baseline models for abstractive summarization, built on the Transformer architecture. These models are categorized into four groups: text-only models with standard input size, layout-aware models with standard input size, long-range text-only models, and *novel* long-range layout-aware models. For reproducibility purposes, we make the models' implementation, along with the fine-tuning and evaluation scripts, publicly available.¹²

We evaluate the hypothesis that incorporating layout information in the form of embeddings enhances the performance of models for long document summarization. We do not explore the use of visual information. While visual features may provide a better understanding of structures such as tables and figures, we do not expect substantial gains with respect to layout-aware models. Indeed, the information provided in figures—information that cannot be captured by layout or text—are commonly described in the caption or related paragraphs.

Text-only Models with Standard Input Size We use Pegasus (Jingqing Zhang et al. 2020) as a text-only baseline for arXiv-Lay and PubMed-Lay. Pegasus is an encoder-decoder model pre-trained using gap-sentences generation, making it a state-of-the-art model for abstractive summarization (Section 3.2.2.2). For the non-English datasets, we rely on a finetuned mBART as our baseline. mBART (Yinhan Liu et al. 2020) is a multilingual sequence-to-sequence model pretrained on large-scale monolingual corpora in many languages using the BART objective (M. Lewis et al. 2019). We use its extension, mBART-50 (Y. Tang et al. 2020),¹³ which is created from the original mBART by extending its embeddings layers and pre-training it on a total of 50 languages. Both Pegasus and mBART are limited to a maximum sequence length of 1,024 tokens, which is well below the median length (6,225/3,761/4,074/4,859/6,090/5,094) of each dataset.

Layout-aware Models with Standard Input Size We introduce layout-aware extensions of Pegasus and mBART, respectively denoted as Pegasus+Layout and mBART+Layout. Following the popular LayoutLM (Yiheng Xu et al. 2020) (Section 4.3.1.1), each token bounding box coordinates (x_0, y_0, x_1, y_1) is normalized into an integer in the range $\{0, \dots, 1000\}$. Spatial positions are encoded using four embedding tables, namely two for the coordinate axes (x and y), and the other

12. <https://github.com/recitalAI/loralay-modeling>

13. For the sake of clarity, we refer to mBART-50 as mBART.

two for the bounding box size (width and height). The layout representation of a token is formed by summing the resulting embedding representations. The final representation of a token is then obtained through point-wise summation of its textual, positional, and layout embeddings.

Long-range, Text-only Models To process longer sequences and facilitate comparison with Zaheer et al. (2020), we leverage BigBird (Zaheer et al. 2020), a sparse-attention based Transformer which reduces the quadratic dependency to a linear one (Section 3.3.1.1). For arXiv-Lay and PubMed-Lay, we initialize BigBird from Pegasus (Zaheer et al. 2020) and for the non-English datasets, we use the weights of mBART. The resulting models are referred to as BigBird-Pegasus and BigBird-mBART. For both models, BigBird sparse attention is used only in the encoder. Both models can handle up to 4,096 inputs tokens, which is greater than the median length in PubMed-Lay, HAL and KoreaScience.

Long-range, Layout-aware Models We also include layout information in long-range text-only models. Similarly to layout-aware models with standard input size, we integrate layout information into our long-range models by encoding each token’s spatial position in the page. The resulting models are denoted as BigBird-Pegasus+Layout and BigBird-mBART+Layout.

7.2.2 Implementation Details

We initialize our Pegasus-based and mBART-based models with, respectively, the google/pegasus-large (568M parameters) and facebook/mbart-large-50 (611M) checkpoints shared through the Hugging Face Model Hub.

Following Jingqing Zhang et al. (2020) and Zaheer et al. (2020), we fine-tune our models up to 74k (100k) steps on arXiv-Lay (PubMed-Lay). On HAL, the total number of steps is set to 100k, while it is decreased to 50k for the other non-English datasets. We tested different values for the number of steps (10k, 25k, 50k, 100k) and chose the one that gave the best validation scores for mBART. For each model, we select the checkpoint with the best validation loss. For Pegasus and mBART models, inputs are truncated at 1,024 tokens. For BigBird-Pegasus models, we follow Zaheer et al. (2020) and set the maximum input length to 3,072 tokens. As the median input length is much greater in almost every non-English dataset, we increase the maximum input length to 4,096 tokens for BigBird-mBART models. Output length is restricted to 256 tokens for all models, which is enough to fully capture at least 50% of the summaries in each dataset.

| Model | # Params | arXiv/ arXiv-Lay | | | PubMed/ PubMed-Lay | | |
|--------------------------------------|----------|---------------------|--------------|--------------|-----------------------|--------------|--------------|
| | | R-1 | R-2 | R-L | R-1 | R-2 | R-L |
| Pegasus (Jingqing Zhang et al. 2020) | 568M | 44.21 | 16.95 | 38.83 | 45.97 | 20.15 | 41.34 |
| BigBird-Pegasus (Zaheer et al. 2020) | 576M | 46.63 | 19.02 | 41.77 | 46.32 | 20.65 | 42.33 |
| Pegasus | 568M | 43.81 | 17.27 | 39.07 | 43.52 | 17.96 | 39.75 |
| Pegasus+Layout | 572M | 44.10 | 17.01 | 39.25 | 43.59 | 18.24 | 39.85 |
| BigBird-Pegasus | 576M | 44.43 | 17.74 | 39.59 | 44.80 | 19.32 | 41.09 |
| BigBird-Pegasus+Layout | 581M | 46.02 | 18.95 | 41.15 | 45.69 | 20.38 | 42.05 |

Table 7.3 – ROUGE scores on arXiv-Lay and PubMed-Lay. Reported results obtained by Pegasus and BigBird-Pegasus on the original arXiv and PubMed are highlighted with a gray background. The best results obtained on arXiv-Lay and PubMed-Lay are denoted in bold.

For evaluation, we use beam search and report a single run for each model and dataset. Following Jingqing Zhang et al. (2020) and Zaheer et al. (2020), we set the number of beams to 8 for Pegasus-based models, and 5 for BigBird-Pegasus-based models. For the non-English datasets, we set it to 5 for all models, for fair comparison. For all experiments, we use a length penalty of 0.8.

Models were implemented in Python using PyTorch (Paszke et al. 2017) and Hugging Face (Wolf et al. 2019) libraries. In all experiments, we use Adafactor (Shazeer and Stern 2018), a stochastic optimization method based on Adam (Kingma and Ba 2014) that reduces memory usage while retaining the empirical benefits of adaptivity. We set a learning rate warmup over the first 10% steps—except on arXiv-Lay where it is set to 10k, consistently with Zaheer et al. (2020)—and use a square root decay of the learning rate. All our experiments have been run on four Nvidia V100 with 32GB each.

7.3 Results and Discussion

We discuss the results obtained by our models on the LoRaLay corpus, and offer a human analysis of the summaries generated by BigBird and its layout-aware counterpart. Additionally, we present case studies to shed light on scenarios in which layout proves to be most beneficial.

7.3.1 General Results

In Table 7.3, we report the ROUGE scores obtained on arXiv and PubMed datasets (reported by Zaheer et al. (2020)), as well as on the corresponding

| | Pegasus | Pegasus +Layout | BigBird-Pegasus | BigBird-Pegasus +Layout |
|------------------------|---------|--------------------|-----------------|----------------------------|
| Pegasus | – | 0.34 / 0.10 | 0.52 / 1.34 | 2.08 / 2.30 |
| Pegasus+Layout | – | – | 0.34 / 1.24 | 1.90 / 2.20 |
| BigBird-Pegasus | – | – | – | 1.56 / 0.96 |

Table 7.4 – Absolute ROUGE-L score differences between each pair of models, on arXiv-Lay/PubMed-Lay (column – row).

| Distribution | Q1 | | Q2 | | Q3 | |
|---------------------------------|-----------|------------|-----------|------------|-----------|------------|
| | arXiv-Lay | PubMed-Lay | arXiv-Lay | PubMed-Lay | arXiv-Lay | PubMed-Lay |
| Article Length | 6,226 | 3,513 | 9,142 | 5,557 | 13,190 | 8,036 |
| Summary Length | 119 | 130 | 159 | 182 | 202 | 247 |
| σ of bounding box height | 3.37 | 1.34 | 3.98 | 1.73 | 4.70 | 2.28 |

Table 7.5 – Quartiles calculated from the distributions of article lengths, summary lengths, and variation in the height of bounding boxes, for arXiv-Lay and PubMed-Lay.

layout-augmented counterparts we release (arXiv-Lay and PubMed-Lay). Table 7.6 presents the ROUGE scores reported on the non-English datasets.

Comparison with the original datasets We observe, for both Pegasus and BigBird-Pegasus, a drop in performance w.r.t. the scores obtained on the original datasets, as reported by Zaheer et al. (2020). This can be explained by two factors. First, our extended datasets contain less training data due to the inability to process all original documents. Secondly, the settings are different: while the original arXiv and PubMed datasets contain clear discourse information (e.g., each section is delimited by markers) obtained from \LaTeX files, documents in our extended versions are built by parsing raw PDF files. Therefore, the task is more challenging for text-only baselines, as they have no access to the discourse structure of documents, which further underlines the importance of taking structural information, brought by visual cues, into account.

Impact of layout on long document summarization We now investigate the impact of integrating layout information for summarizing long documents. On arXiv-Lay and PubMed-Lay, we observe that, while the addition of layout to Pegasus does not improve the ROUGE-L scores, there are gains in integrating layout information into BigBird-Pegasus. To assess whether these gains are significant, we perform significance analysis at the 0.05 level using bootstrap, and estimate a ROUGE-L threshold that predicts when improvements are significant. ROUGE-L improvements between each pair of models are reported in Ta-

| Model | HAL (FR) | | | SciELO-ES (ES) | | | SciELO-PT (PT) | | | KoreaScience (KR) | | |
|----------------------|--------------|--------------|--------------|----------------|--------------|--------------|----------------|--------------|--------------|-------------------|-------------|--------------|
| | R-1 | R-2 | R-L | R-1 | R-2 | R-L | R-1 | R-2 | R-L | R-1 | R-2 | R-L |
| MBART | 47.05 | 22.23 | 42.00 | 41.04 | 15.65 | 36.55 | 41.18 | 15.53 | 36.42 | 17.33 | 7.70 | 16.94 |
| MBART+Layout | 46.65 | 21.96 | 41.67 | 42.27 | 15.73 | 37.47 | 39.45 | 14.17 | 34.37 | 15.43 | 6.69 | 14.98 |
| BigBird-MBART | 49.85 | 25.71 | 45.04 | 42.64 | 16.60 | 37.76 | 44.85 | 18.70 | 39.63 | 18.96 | 8.01 | 18.55 |
| BigBird-MBART+Layout | 49.99 | 25.20 | 45.20 | 45.64 | 19.33 | 40.71 | 45.47 | 20.40 | 40.51 | 20.36 | 9.49 | 19.95 |

Table 7.6 – ROUGE scores on the non-English datasets. The best results for each dataset are reported in bold.

ble 7.4. On arXiv-Lay, we compute a threshold of 1.48 ROUGE-L, showing that BigBird-Pegasus+Layout significantly outperforms every other model. In particular, we find a 1.56 ROUGE-L improvement between BigBird-Pegasus and its layout-augmented counterpart, demonstrating that the addition of layout to long-range modeling significantly improves summarization. On PubMed-Lay, we compute a threshold of 1.77. Hence, the 0.96 ROUGE-L improvement from BigBird-Pegasus to its layout-augmented counterpart is not significant. However, the variance in font sizes in PubMed-Lay is much smaller compared to arXiv-Lay (see Table 7.9, which lists the quartiles computed from the distributions of article lengths, summary lengths, and variation in the height of bounding boxes, for arXiv-Lay and PubMed-Lay). This reflects an overall more simplistic layout. Therefore, we argue that layout integration has a lesser impact in PubMed-Lay, which can explain the non-significance of results. In addition, we find that BigBird-Pegasus significantly outperforms Pegasus and Pegasus+Layout only when augmented with layout, with an improvement of, respectively, 2.3 and 2.2 points. This demonstrates the importance of combining layout-aware and long-range modeling.

On HAL, we note that BigBird-MBART does not benefit from layout. After investigation, we hypothesize that this is due to the larger presence of single-column and simple layouts, which makes layout integration less needed. However, we notice that combining layout with long-range modeling brings substantial improvements over MBART on KoreaScience and both SciELO datasets. Further, we find that the plain-text BigBird models do not improve over the layout-aware Pegasus and MBART on arXiv-Lay and SciELO-ES, respectively. This finding demonstrates that simply capturing more context does not always suffice, emphasizing the need to combine layout-aware and long-range approaches.

Overall, results show a clear benefit of integrating layout information for long document summarization.

Adaptation to multiple languages On HAL, SciELO-ES and SciELO-PT, table 7.6 demonstrates that BigBird+MBART achieves scores comparable to those obtained by BigBird+Pegasus on the English datasets. However, on KoreaScience, we can see a significant drop in performance for every model w.r.t the other non-

| Dataset | Train | Validation | Test |
|-------------------|-------|------------|-------|
| HAL (fr) | 90.72 | 90.54 | 85.84 |
| SciELO-ES (es) | 84.86 | 84.28 | 84.90 |
| SciELO-PT (pt) | 90.95 | 90.58 | 91.96 |
| KoreaScience (ko) | 73.53 | 70.26 | 68.78 |

Table 7.7 – Percent confidence obtained for the main language, for each dataset split.

English datasets. At first glance, we notice a high amount of English segments (*e.g.*, tables, figure captions, scientific concepts) in documents in KoreaScience. To investigate this, we use the `cld2` library¹⁴ to detect the language in each non-English document. We consider the percent confidence of the top-1 matching language as an indicator of the presence of the main language (*i.e.*, French, Spanish, Portuguese or Korean) in a document, and average the results to obtain a score for the whole dataset. Table 7.7 reports the average percent confidence obtained on each split, for each dataset. We find that the percentage of text written in the main language in KoreaScience (*i.e.*, Korean) is smaller than in other datasets. As the MBART-based models expect only one language in a document (the information is encoded using a special token), we claim that the strong presence of non-Korean segments in KoreaScience causes them to suffer from interference problems. Therefore, we highlight that KoreaScience is a more challenging dataset, and we hope our work will boost research on better long-range, multimodal *and* multilingual models.

We employ ROUGE as it is the most commonly employed metric for abstractive summarization, facilitating direct comparison with prior models. However, recognizing the constraints of ROUGE regarding diversity, redundancy, readability, consistency, and relevance (M. Zhang et al. 2024), we acknowledge that exploring alternative metrics such as BLEU (Papineni et al. 2002) and METEOR (Banerjee and Lavie 2005) would yield further insights.

7.3.2 Human Evaluation

Acknowledging the limitations of ROUGE and aiming to strengthen our claim that document layout plays a crucial role in summarizing long textual content, we conduct a human evaluation of summaries generated by BigBird-Pegasus/BigBird-MBART and their layout-aware counterparts. We evenly sample 50 documents from arXiv-Lay and HAL test sets, filtering documents by their topics (computer science) to match the judgment capabilities of the three human annotators. Using

14. <https://github.com/GregBowyer/cld2-cffi>

LoRaLay Evaluation Interface

Evaluation guidelines

Document 0806.3537 (1/50)

Statistical Learning of Arbitrary Computable Classifiers

[Link to full document](#)

Model A

Model B

Ground-truth abstract

Statistical learning theory chiefly studies restricted hypothesis classes, particularly those with finite Vapnik-Chervonenkis (VC) dimension. The fundamental quantity of interest is the **sample complexity**: the **number of samples** required to **learn to a specified level of accuracy**. Here we consider **learning** over the set of all computable labeling functions. Since the VC-dimension is infinite and a priori (uniform) **bounds** on the **number of samples** are impossible, we let the **learning** algorithm decide when it has seen sufficient **samples** to have learned. We first show that **learning** in this setting is indeed possible, and develop a **learning** algorithm. We then show, however, that bounding **sample** complexity independently of the distribution is impossible. Notably, this impossibility is entirely due to the requirement that the **learning** algorithm be computable, and not due to the **statistical** nature of the problem.

You selected the following sentence generated by Model B. Highlight the parts in the sentence that can be found in the ground-truth abstract.

Conventional statistical learning theory attempts to bound the **number of samples needed to learn to a specified level of accuracy** for each of the above models (e.g. neural networks, support vector machines).

1

Next sentence

Summary generated by Model B

| | Sentence | Precision (%) |
|-------------------------------------|---|---------------|
| <input checked="" type="checkbox"/> | Conventional statistical learning theory attempts to bound the number of samples needed to learn to a specified level of accuracy for each of the above models (e.g. neural networks, support vector machines). | 40.62 |
| <input type="checkbox"/> | However, if we allow ourselves to change the model, then the VC-dimension of the overall learning algorithm is not finite, and much of statistical learning theory does not directly apply. | 16.67 |
| <input type="checkbox"/> | In contrast, we prove that distribution-independent bounds do not exist altogether for computable learning algorithms in our setting. | 72.22 |
| <input type="checkbox"/> | Our results imply that computable learning algorithms in the universal setting must "waste samples" in the sense of requiring more samples than is necessary for statistical reasons alone. | 0.0 |

Recall (%)

30.15

Coherence

0 1 2 3 4 5

Fluency

0 1 2 3 4 5

I am unable to evaluate this document.

Next

Figure 7.5 – LoRaLay evaluation interface.

| Metric | BigBird | BigBird+Layout |
|---------------|--------------------|---------------------|
| ↑ Precision % | 35.15 (0.81) | 37.51 (0.70) |
| ↑ Recall % | 28.07 (0.73) | 33.59 (0.86) |
| ↑ Coherence | 3.80 (0.38) | 3.75 (0.62) |
| ↑ Fluency | 4.48 (0.03) | 4.34 (0.16) |
| ↓ Overlap % | 8.77 (0.24) | 7.49 (0.36) |
| ↑ Flow % | 30.75 (0.68) | 33.02 (0.71) |

Table 7.8 – Average human judgement scores obtained by comparing gold-truth abstracts and summaries generated by BigBird and BigBird+Layout from 50 documents sampled from arXiv-Lay and HAL. Inter-rater agreement is computed using Krippendorff’s alpha coefficient, and enclosed between parentheses. Best scores are reported in bold. ↑ means that higher is considered “better”, whereas ↓ signifies the opposite.

the Streamlit¹⁵ framework, we design and develop an interface to aid human evaluation of summarization models.¹⁶ A snapshot of the interface is presented in Figure 7.5. For each document, annotators are provided with the ground-truth abstract, along with the summaries generated by BigBird and BigBird+Layout. To ensure an unbiased evaluation process, annotators are unaware of whether a generated summary is produced by BigBird or BigBird+Layout. The evaluation process is as follows: for each sentence s_i in the generated summary, we ask the annotators to highlight 1) tokens relevant to the ground-truth abstract in s_i , along with 2) the equivalent parts h_i in the ground-truth abstract. Further, we ask them to rate the summary in terms of *coherence* and *fluency*, on a scale of 0 to 5, following the DUC quality guidelines (Dang 2005).

This highlighting process enables the computation of *precision* and *recall*, where precision is the percentage of highlighted information (*i.e.*, relevant tokens) in the generated summary, and recall is the percentage of highlighted information in the ground-truth abstract. Additionally, an *overlap ratio* can be calculated as the percentage of highlighted information that appears multiple times in the generated summary. This is determined by identifying instances where the same highlighted information from the ground-truth abstract corresponds to more than one highlighted part in the generated summary. Furthermore, a *flow percentage* is computed to evaluate how well the order of the ground-truth information is preserved. For each pair of consecutive sentences in the generated summary, represented by s_{i-1} and s_i , we examine the highlighted text h_{i-1} and h_i in the corresponding ground-truth abstract. The objective is to determine how often any

15. <https://streamlit.io/>

16. The code is publicly available at <https://github.com/ngdlaura/loralay-eval-interface>.

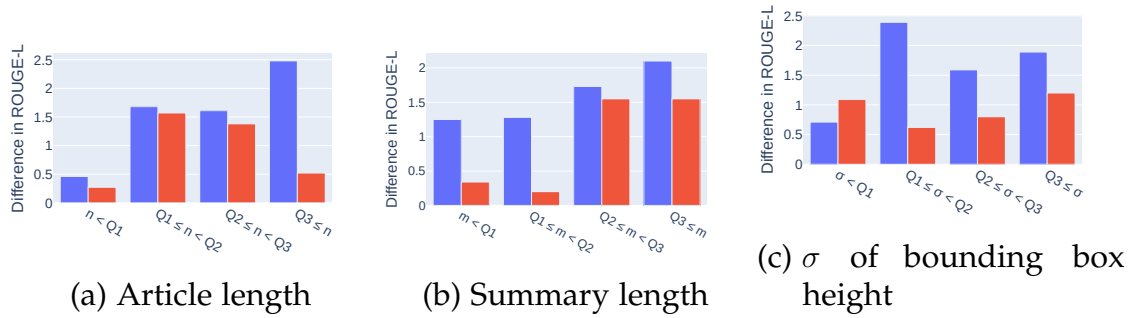


Figure 7.6 – Benefit of using layout on arXiv-Lay (blue) and PubMed-Lay (red), defined as the difference in ROUGE-L scores between BigBird-Pegasus+Layout and BigBird-Pegasus. For each dataset, quartiles are calculated from the distributions of article lengths (a), summary lengths (b) and variance in the height of the bounding boxes (c). ROUGE-L scores are then computed per quartile range, and averaged over each range.

| Distribution | Q ₁ | | Q ₂ | | Q ₃ | |
|---------------------------------|----------------|------------|----------------|------------|----------------|------------|
| | arXiv-Lay | PubMed-Lay | arXiv-Lay | PubMed-Lay | arXiv-Lay | PubMed-Lay |
| Article Length | 6,226 | 3,513 | 9,142 | 5,557 | 13,190 | 8,036 |
| Summary Length | 119 | 130 | 159 | 182 | 202 | 247 |
| σ of bounding box height | 3.37 | 1.34 | 3.98 | 1.73 | 4.70 | 2.28 |

Table 7.9 – Quartiles calculated from the distributions of article lengths, summary lengths, and variation in the height of bounding boxes, for arXiv-Lay and PubMed-Lay.

token from h_i occurs after a token in h_{i-1} . This calculation quantifies how well the sequence of highlighted information in the generated summary aligns with the sequential order of highlighted information in the ground-truth abstract.

Table 7.8 reports the scores for each metric and model, averaged over all 50 documents, along with inter-rater agreements, computed using Krippendorff’s alpha coefficient. We find that adding layout to the models significantly improves precision and recall, results in less overlap (repetition), and is more in line with the ground truth order. In terms of coherence and fluency, both models achieve comparable scores. To conclude, reported results show that human annotators strongly agree that adding layout generates better summaries, further validating our claim that layout provides vital information for summarization tasks.

7.3.3 Case Studies

To have a better understanding of the previous results, we focus on uncovering the cases in which layout is most helpful. To this end, we identify features that relate to the necessity of having layout: 1) article length, as longer texts are intuitively easier to understand with layout, 2) summary length, as longer summaries are likely to cover more salient information, and 3) variance in font sizes (using the height of the bounding boxes), and, as such, the complexity of the layout. The benefit of using layout is measured as the difference in ROUGE-L scores between BigBird-Pegasus+Layout and its purely textual counterpart, on arXiv-Lay and PubMed-Lay. We compute quartiles from the distributions of article lengths, ground-truth summary lengths, and variance in the height of bounding boxes.

The quartiles are provided in Table 7.9. Based on the aforementioned factors, the scores obtained by each model are then grouped by quartile range, and averaged over each range (see Figure 7.6). On arXiv-Lay, we find that layout brings most improvement when dealing with the 25% longest documents and summaries, while, for both datasets, layout is least beneficial for the shortest documents and summaries. These results corroborate our claim that layout can bring important information about long-range context. Concerning the third factor, we see, on PubMed-Lay, that layout is most helpful for documents that have the widest ranges of font sizes, showcasing the advantage of using layout to capture salient information.

7.4 Conclusion

In this chapter, we have presented LoRaLay, a set of large-scale datasets for long-range and layout-aware text summarization. LoRaLay provides the research community with 4 novel multimodal corpora covering French, Spanish, Portuguese, and Korean languages, built from scientific articles. Furthermore, it includes additional layout and visual information for existing long-range summarization datasets (arXiv and PubMed). We provide adapted architectures merging layout-aware and long-range models, and show the importance of layout information in capturing long-range dependencies. Finally, we design an annotation interface for human evaluation of summaries, and introduce the flow metric to offer insights into the preservation of information flow in generated summaries.

CONCLUSION

This thesis explores the field of Document Understanding in the context of Deep Learning. Although Deep Learning has made notable strides in enhancing document understanding systems, the field still encounters various challenges in real-world applications, including limitations in handling long documents, disparities between high-quality training data and real-world documents, resource constraints, and dependence on Optical Character Recognition (OCR) systems.

Our exploration is guided by two main research axes. First, we delve into the challenge of efficiently and effectively encoding the multimodal nature of documents, extending beyond textual content to consider the complex interplay of layout and visual elements. Secondly, we address the challenge of efficiently and effectively processing long and complex documents while leveraging their layout.

8.1 Summary of Contributions

In the past decade, Deep Learning techniques, in particular Foundation Models (Devlin et al. 2018; Radford et al. 2019; Touvron et al. 2023) built upon the Transformer architecture (Vaswani et al. 2017), have dominated the field of Natural Language Processing (NLP). This trend has prompted a paradigm shift in the field of Document Understanding, witnessing a significant rise in the adoption of large-scale, general-purpose multimodal Pre-trained Language Models. LayoutLM (Yiheng Xu et al. 2020) initiated the use of Transformers' modeling capabilities to jointly pre-train text and layout information for a range of tasks involving visually-rich documents.

Our first research axis involved contributing to this shift towards integrating document layout into pre-training by developing novel strategies for encoding layout information efficiently (Chapter 5) and enhancing robustness on documents with unreliable reading order information (Chapter 6).

Various approaches have investigated the extension of multimodal Pre-trained Language Models to longer, more complex documents. Our second research axis

centered on leveraging layout information to capture long-range dependencies (Chapter 7), capitalizing on its potential to reduce the complexity of self-attention mechanisms (Chapter 5). We will now outline the main contributions derived from our work.

Literature review In Chapters 2 to 4, we conducted a thorough literature review of works relevant to our research. Chapter 2 focused on foundational aspects of language modeling, particularly Neural Language Models. In Chapter 3, we detailed the Transformer architecture and its core component—the attention mechanism. We explored the use of Transformers to construct key Pre-trained Language Models, before focusing on efficient self-attention model variants for handling long texts. In Chapter 4, we examined tasks, datasets, and Deep Learning approaches in Document Understanding, emphasizing multimodal pre-training techniques pivotal to our contributions.

Layout-based attention strategies While prior works typically consider layout as a mere positional feature, failing to fully capitalize on the strong correlation between text and layout, we took a different approach in Chapter 5. Moving beyond this conventional view of layout, we drew inspiration from human reading strategies to effectively leverage layout and tackle the efficiency challenges of multimodal pre-training techniques. We introduced *Skim-Attention*, a novel attention mechanism based on layout information exclusively. Addressing key research questions, we illustrated Skim-Attention’s ability to determine attention solely from layout (Skimformer), which can potentially be used to reduce the complexity of self-attention. Furthermore, we demonstrated that Skim-Attention can also be used as a mask for any Pre-trained Language Model (Skimming Mask), enhancing performance while sparsifying attention. Lastly, we showed the emergence of a document structure representation within Skim-Attention. In summary, by strategically integrating layout into language modeling, Skim-Attention marks a pioneering effort in leveraging layout to alleviate the computational burden associated with self-attention.

While improving efficiency, determining attention solely from layout might overlook crucial semantic clues essential for the model’s comprehension. While Skimming Mask addresses this issue, the masking pattern should be learned in an end-to-end fashion alongside the Transformer model for full adaptability. Therefore, potential extensions of this work include the implementation of an end-to-end version of Skimming Mask. Given the discrete nature of attention masks, reinforcement learning algorithms could be explored. Another research direction involves the application of Skim-Attention to tasks that require capturing longer-range dependencies.

Leveraging layout to avoid reading order issues The majority of pre-training methods for Document Understanding rely on serialized text. However, OCR engines often struggle to accurately determine reading orders for documents with complex layouts, thereby impacting the entire text processing pipeline. In contrast, layout inherently organizes content visually, providing the correct reading order for documents. In Chapter 6, we diverged from conventional approaches by relying solely on layout information to offer an alternative for the provided reading order of documents. We introduced *Layout2Pos*—a shallow Transformer that generates position embeddings from the spatial arrangement of text, avoiding reading order issues. Integrated into a Bidirectional and Auto-Regressive Transformers (BART) model (M. Lewis et al. 2019) architecture, *Layout2Pos* competes with models relying on reading order across three benchmark datasets for visual information extraction. Importantly, we showcased substantial performance drops in models dependent on reading order when evaluated with a different reading order than seen during training. This emphasizes the significance of not relying on OCR-induced reading orders. Overall, *Layout2Pos* marks the pioneering use of layout as a method to provide reading order information, completely eliminating reliance on OCR-induced reading orders for downstream tasks.

To further enhance the model, several avenues for improvement can be explored. Layout information could be better captured, for instance, by incorporating details such as page numbers. Integrating visual information could also help generate position embeddings that better align with the reading order of documents. Moreover, to improve generalizability, another potential extension of this work involves focusing on more complex datasets containing longer documents and encompassing languages beyond English. Furthermore, in our sequence-to-sequence evaluation framework, any organization of key-value pairs is deemed valid. However, the use of teacher forcing during training tends to prioritize a single correct output, potentially penalizing valid responses with different entity orders. Therefore, future work involves exploring permutation invariant losses to enhance robustness.

Leveraging layout to deal with long and layout-rich documents Prior research in cognitive sciences, particularly in the context of long documents, has showcased the need for efficient modeling of layout information in information processing. However, methods that address efficient understanding of long documents with awareness of their layouts are still largely underexplored. In Chapter 7, we extended the exploration of layout-based techniques introduced in Chapters 5 and 6 to long documents. We focused on the Text Summarization task, given its reliance on document structures to guide the generation of summaries and its natural suitability to benefit from a global context. We introduced *LoRaLay*, the first collection of datasets for long-range summarization with visual/layout information, ad-

addressing the lack of multimodal information in existing summarization datasets. We augmented popular English datasets with layout/visual information and built new datasets for French, Spanish, Portuguese, and Korean. Baselines combining layout-aware and long-range models were introduced, achieving state-of-the-art results and highlighting the importance of considering both research directions for summarization of long documents. Additionally, we proposed an annotation interface for human evaluation and a novel metric to assess information flow preservation in generated summaries. By bridging a significant gap in existing resources, LoRaLay provides a foundation for advancing our understanding of the role of layout in handling long textual content, propelling progress in developing efficient and effective layout-aware long-range approaches.

Future extensions of this work could involve developing vision-aware models, further expanding the scope of multimodal considerations in the context of long document summarization. Additionally, future work could include the development of more advanced models that strategically leverage layout to efficiently and effectively process long documents.

8.2 Perspectives

Towards universal end-to-end document understanding systems A possible direction involves the development of unified foundation document understanding models. Propelled by Text-to-Text Transfer Transformer (T₅) (Raffel et al. 2020), research into unifying training processes across diverse tasks and domains using the sequence-to-sequence framework has made remarkable advancements. In the field of Document Understanding, unified training has also gained traction with the development of generative language models. As detailed in Section 4.3.3.3, TILT (Powalski et al. 2021) is partly pre-trained in a supervised manner using various tasks related to visually-rich documents and unified under the sequence-to-sequence framework, whereas UDOP (Z. Tang et al. 2023) unifies representations for image, text, and layout while employing self-supervised tasks across all modalities. Leveraging the same sequence-to-sequence framework, our generative language models introduced in Chapter 6 and 7 align with this research trajectory and present opportunities for further improvement through unified training approaches.

While our generative Pre-trained Language Model proposed in Chapter 6 does not rely on sequential position information, it still employs text extracted through OCR, introducing the possibility of inaccuracies and error propagation to the document understanding model. Another noteworthy direction in advancing end-to-end document understanding systems involves OCR-free models, exemplified by

Donut (Kim et al. 2022). Donut employs a high-resolution image encoder during pre-training to detect text, thereby establishing a direct mapping from raw input images to the desired outputs. As such, a perspective for future research involves the development of OCR-free end-to-end frameworks that unify all modalities and task paradigms, laying the groundwork for foundation document understanding models. Nevertheless, it is worth noting that the increasing prevalence of digital-born documents may reduce the necessity and importance of OCR in the long term. Consequently, our work in this PhD, which still depends on text extracted through another system (*e.g.*, PDF structured information), represents an important research avenue.

Leveraging Large Language Models The contributions in this PhD focus on Pre-trained Language Models, which necessitate specific fine-tuning on diverse downstream datasets to achieve optimal performance and fall short under the zero-shot setting. In contrast, the next iteration in Language Models, Large Language Models, has transformed the landscape of NLP by showcasing remarkable abilities to address tasks based on minimal instructions (J. Wei et al. 2021) or a small number of examples integrated into the prompt (T. Brown et al. 2020) across diverse linguistic applications. Exploration of Large Language Models for multimodal generation has gained traction recently. LLaVA (H. Liu et al. 2023), mPLUG-Owl (Q. Ye et al. 2023), and IDEFICS (Laurençon et al. 2023) leverage Large Language Models to construct unified end-to-end multimodal models tailored for processing images that contain text (*e.g.*, book covers and movie posters). Despite these achievements, successfully applying Large Language Models to the domain of Document Understanding, where text is dense and visually-situated, remains a challenge primarily due to the lack of in-domain training. Specifically, these models struggle with tasks such as handling diverse image types, recognizing complex textual content, and comprehending the relationships between visual semantics and textual information. With the rise of prominent efforts to improve Large Language Models for document understanding tasks, such as mPLUG-DocOwl (Q. Ye et al. 2023), LMDX (Perot et al. 2023), and DocLLM (D. Wang et al. 2023), a natural perspective for future research involves developing stronger document understanding abilities within Large Language Models.

BIBLIOGRAPHY

- Adel, Heike, Benjamin Roth, and Hinrich Schütze (2016). “Comparing convolutional neural networks to traditional models for slot filling”. In: *arXiv preprint arXiv:1603.05157* (cit. on p. 23).
- Afzal, Muhammad Zeshan, Samuele Capobianco, Muhammad Imran Malik, Simone Marinai, Thomas M Breuel, Andreas Dengel, and Marcus Liwicki (2015). “Deepdocclassifier: Document classification with deep convolutional neural network”. In: *2015 13th international conference on document analysis and recognition (ICDAR)*. IEEE, pp. 1111–1115 (cit. on p. 66).
- Ainslie, Joshua, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang (2020). “ETC: Encoding long and structured inputs in transformers”. In: *arXiv preprint arXiv:2004.08483* (cit. on p. 52).
- Antonacopoulos, Apostolos, Christian Clausner, Christos Papadopoulos, and Stefan Pletschacher (2013). “Icdar 2013 competition on historical newspaper layout analysis (hnlA 2013)”. In: *2013 12th International Conference on Document Analysis and Recognition*. IEEE, pp. 1454–1458 (cit. on p. 62).
- Appalaraju, Srikar, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R Manmatha (2021). “DocFormer: End-to-End Transformer for Document Understanding”. In: *arXiv preprint arXiv:2106.11539* (cit. on pp. ix, 70, 72, 74, 76, 77, 79, 80).
- Arisoy, Ebru, Tara N Sainath, Brian Kingsbury, and Bhuvana Ramabhadran (2012). “Deep neural network language models”. In: *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pp. 20–28 (cit. on p. 16).
- Baechler, Micheal and Rolf Ingold (2011). “Multi resolution layout analysis of medieval manuscripts using dynamic mlp”. In: *2011 International Conference on Document Analysis and Recognition*. IEEE, pp. 1185–1189 (cit. on p. 60).
- Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). “Neural machine translation by jointly learning to align and translate”. In: *arXiv preprint arXiv:1409.0473* (cit. on pp. 29–32).
- Bahl, Lalit R, Peter F Brown, Peter V de Souza, and Robert L Mercer (1989). “A tree-based statistical language model for natural language speech recognition”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37:7, pp. 1001–1008 (cit. on p. 16).
- Banerjee, Satanjeev and Alon Lavie (2005). “METEOR: An automatic metric for MT evaluation with improved correlation with human judgments”. In: *Proceed-*

- ings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pp. 65–72 (cit. on p. 141).
- Bang, Yejin, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. (2023). “A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity”. In: *arXiv preprint arXiv:2302.04023* (cit. on p. 49).
- Bao, Hangbo, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, et al. (2020). “Unilmv2: Pseudo-masked language models for unified language model pre-training”. In: *International conference on machine learning*. PMLR, pp. 642–652 (cit. on p. 78).
- Beltagy, Iz, Arman Cohan, and Kyle Lo (2019). “Scibert: Pretrained contextualized embeddings for scientific text”. In: *arXiv preprint arXiv:1903.10676* 1.1.3, p. 8 (cit. on p. 42).
- Beltagy, Iz, Matthew E Peters, and Arman Cohan (2020). “Longformer: The long-document transformer”. In: *arXiv preprint arXiv:2004.05150* (cit. on pp. 52, 82, 96, 97, 100, 127).
- Bengio, Yoshua, Réjean Ducharme, and Pascal Vincent (2000). “A neural probabilistic language model”. In: *Advances in neural information processing systems* 13 (cit. on pp. 20, 21, 24).
- Binmakhashen, Galal M and Sabri A Mahmoud (2019). “Document layout analysis: a comprehensive survey”. In: *ACM Computing Surveys (CSUR)* 52.6, pp. 1–36 (cit. on p. 62).
- Blei, David M, Andrew Y Ng, and Michael I Jordan (2003). “Latent dirichlet allocation”. In: *Journal of machine Learning research* 3, Jan, pp. 993–1022 (cit. on p. 21).
- Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov (2017). “Enriching word vectors with subword information”. In: *Transactions of the association for computational linguistics* 5, pp. 135–146 (cit. on p. 25).
- Borchmann, Łukasz, Michał Pietruszka, Tomasz Stanislawek, Dawid Jurkiewicz, Michał Turski, Karolina Szyndler, and Filip Graliński (2021). “DUE: End-to-End Document Understanding Benchmark”. In: *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)* (cit. on pp. 3, 64, 65).
- Britton, Bruce K, Shawn M Glynn, Bonnie J Meyer, and MJ Penland (1982). “Effects of text structure on use of cognitive capacity during reading.” In: *Journal of Educational Psychology* 74.1, p. 51 (cit. on p. 86).
- Brown, Peter F, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer (1992). “Class-based n-gram models of natural language”. In: *Computational linguistics* 18.4, pp. 467–480 (cit. on p. 15).
- Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda

- Askell, et al. (2020). "Language models are few-shot learners". In: *Advances in neural information processing systems* 33, pp. 1877–1901 (cit. on pp. 46, 48, 58, 151).
- Burt, Peter J and Edward H Adelson (1987). "The Laplacian pyramid as a compact image code". In: *Readings in computer vision*. Elsevier, pp. 671–679 (cit. on p. 66).
- Cai, Zhaowei and Nuno Vasconcelos (2018). "Cascade r-cnn: Delving into high quality object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6154–6162 (cit. on pp. 66, 67).
- Cao, Shuyang and Lu Wang (2022). "Hibrids: Attention with hierarchical biases for structure-aware long document summarization". In: *arXiv preprint arXiv:2203.10741* (cit. on p. 128).
- Chaudhari, Sneha, Varun Mithal, Gungor Polatkan, and Rohan Ramanath (2021). "An attentive survey of attention models". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 12.5, pp. 1–32 (cit. on p. 31).
- Chelba, Ciprian, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson (2013). "One billion word benchmark for measuring progress in statistical language modeling". In: *arXiv preprint arXiv:1312.3005* (cit. on p. 47).
- Child, Rewon, Scott Gray, Alec Radford, and Ilya Sutskever (2019). "Generating long sequences with sparse transformers". In: *arXiv preprint arXiv:1904.10509* (cit. on p. 52).
- Cho, Jaemin, Jiasen Lu, Dustin Schwenk, Hannaneh Hajishirzi, and Aniruddha Kembhavi (2020). "X-lxmert: Paint, caption and answer questions with multi-modal transformers". In: *arXiv preprint arXiv:2009.11278* (cit. on p. 80).
- Cho, Kyunghyun, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio (2014). "On the properties of neural machine translation: Encoder-decoder approaches". In: *arXiv preprint arXiv:1409.1259* (cit. on p. 30).
- Choromanski, Krzysztof, Valerii Likhoshesterov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. (2020). "Rethinking attention with performers". In: *arXiv preprint arXiv:2009.14794* (cit. on p. 54).
- Chowdhery, Aakanksha, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. (2022). "Palm: Scaling language modeling with pathways". In: *arXiv preprint arXiv:2204.02311* (cit. on p. 48).
- Clausner, Christian, Stefan Pletschacher, and Apostolos Antonopoulos (2013). "The significance of reading order in document recognition and its evaluation". In: *2013 12th International Conference on Document Analysis and Recognition*. IEEE, pp. 688–692 (cit. on p. 104).

- Clissa, Luca (2022). "Survey of Big Data sizes in 2021". In: *arXiv preprint arXiv:2202.07659* (cit. on p. 1).
- Cohan, Arman, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian (2018). "A discourse-aware attention model for abstractive summarization of long documents". In: *arXiv preprint arXiv:1804.05685* (cit. on pp. 50, 128–131, 134, 135).
- Collobert, Ronan (2011). "Deep learning for efficient discriminative parsing". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, pp. 224–232 (cit. on p. 23).
- Collobert, Ronan, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa (2011). "Natural language processing (almost) from scratch". In: *Journal of machine learning research* 12.ARTICLE, pp. 2493–2537 (cit. on p. 24).
- Conneau, Alexis, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov (2019). "Unsupervised cross-lingual representation learning at scale". In: *arXiv preprint arXiv:1911.02116* (cit. on p. 43).
- Dai, Zihang, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov (2019). "Transformer-xl: Attentive language models beyond a fixed-length context". In: *arXiv preprint arXiv:1901.02860* (cit. on pp. 38, 53).
- Dang, Hoa Trang (2005). "Overview of DUC 2005". In: *Proceedings of the document understanding conference*. Vol. 2005, pp. 1–12 (cit. on p. 143).
- Das, Arindam, Saikat Roy, Ujjwal Bhattacharya, and Swapan K Parui (2018). "Document image classification with intra-domain transfer learning and stacked generalization of deep convolutional neural networks". In: *2018 24th international conference on pattern recognition (ICPR)*. IEEE, pp. 3180–3185 (cit. on p. 66).
- Dasigi, Pradeep, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A Smith, and Matt Gardner (2021). "A dataset of information-seeking questions and answers anchored in research papers". In: *arXiv preprint arXiv:2105.03011* (cit. on p. 50).
- Dauphinee, Tyler, Nikunj Patel, and Mohammad Rashidi (2019). "Modular multi-modal architecture for document classification". In: *arXiv preprint arXiv:1912.04376* (cit. on p. 66).
- Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255 (cit. on p. 66).
- Denk, Timo I and Christian Reisswig (2019). "Bertgrid: Contextualized embedding for 2d document representation and understanding". In: *arXiv preprint arXiv:1909.04948* (cit. on pp. 67, 68).
- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "Bert: Pre-training of deep bidirectional transformers for language understanding".

- In: *arXiv preprint arXiv:1810.04805* (cit. on pp. 27, 28, 37, 39, 42, 58, 78, 96, 97, 100, 118, 147).
- Ding, Ming, Chang Zhou, Hongxia Yang, and Jie Tang (2020). “Cogltx: Applying bert to long texts”. In: *Advances in Neural Information Processing Systems* 33, pp. 12792–12804 (cit. on p. 50).
- Dong, Haoyu, Shijie Liu, Shi Han, Zhouyu Fu, and Dongmei Zhang (2019). “Tablesense: Spreadsheet table detection with convolutional neural networks”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01, pp. 69–76 (cit. on p. 67).
- Dong, Li, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon (2019). “Unified language model pre-training for natural language understanding and generation”. In: *Advances in neural information processing systems* 32 (cit. on p. 42).
- Dong, Zican, Tianyi Tang, Lunyi Li, and Wayne Xin Zhao (2023). “A survey on long text modeling with transformers”. In: *arXiv preprint arXiv:2302.14502* (cit. on p. 51).
- Fan, Angela, Mike Lewis, and Yann Dauphin (2018). “Hierarchical neural story generation”. In: *arXiv preprint arXiv:1805.04833* (cit. on p. 44).
- Fisher, James L, Stuart C Hinds, and Donald P D’Amato (1990). “A rule-based system for document image segmentation”. In: *[1990] Proceedings. 10th International Conference on Pattern Recognition*. Vol. 1. IEEE, pp. 567–572 (cit. on p. 60).
- Gage, Philip (1994). “A new algorithm for data compression”. In: *C Users Journal* 12.2, pp. 23–38 (cit. on pp. 13, 29).
- Gale, William A and Geoffrey Sampson (1995). “Good-turing frequency estimation without tears”. In: *Journal of quantitative linguistics* 2.3, pp. 217–237 (cit. on p. 16).
- Gao, Liangcai, Yilun Huang, Hervé Déjean, Jean-Luc Meunier, Qinqin Yan, Yu Fang, Florian Kleber, and Eva Lang (2019). “ICDAR 2019 competition on table detection and recognition (cTDaR)”. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, pp. 1510–1515 (cit. on p. 63).
- Gao, Liangcai, Xiaohan Yi, Zhuoren Jiang, Leipeng Hao, and Zhi Tang (2017). “ICDAR2017 competition on page object detection”. In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 1. IEEE, pp. 1417–1422 (cit. on p. 62).
- Gehman, Samuel, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith (2020). “Realtotoxicityprompts: Evaluating neural toxic degeneration in language models”. In: *arXiv preprint arXiv:2009.11462* (cit. on p. 49).
- Göbel, Max, Tamir Hassan, Ermelinda Oro, and Giorgio Orsi (2013). “ICDAR 2013 table competition”. In: *2013 12th International Conference on Document Analysis and Recognition*. IEEE, pp. 1449–1453 (cit. on p. 63).

- Graliński, Filip, Tomasz Stanisławek, Anna Wróblewska, Dawid Lipiński, Agnieszka Kaliska, Paulina Rosalska, Bartosz Topolski, and Przemysław Biecek (2020). “Kleister: A novel task for information extraction involving long documents with complex layout”. In: *arXiv preprint arXiv:2003.02356* (cit. on pp. 64, 105, 124).
- Gu, Zhangxuan, Changhua Meng, Ke Wang, Jun Lan, Weiqiang Wang, Ming Gu, and Liqing Zhang (2022). “Xylayoutlm: Towards layout-aware multimodal networks for visually-rich document understanding”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4583–4592 (cit. on p. 75).
- Guthrie, John T, Tracy Britten, and K Georgene Barker (1991). “Roles of document structure, cognitive strategy, and awareness in searching for information”. In: *Reading Research Quarterly*, pp. 300–324 (cit. on p. 69).
- Ha, Jaekyu, Robert M Haralick, and Ihsin T Phillips (1995). “Recursive XY cut using bounding boxes of connected components”. In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 2. IEEE, pp. 952–955 (cit. on p. 75).
- Hadi, Muhammad Usman, Rizwan Qureshi, Abbas Shah, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, Seyedali Mirjalili, et al. (2023). “Large Language Models: A Comprehensive Survey of its Applications, Challenges, Limitations, and Future Prospects”. In: (cit. on p. 48).
- Harley, Adam W, Alex Ufkes, and Konstantinos G Derpanis (2015). “Evaluation of deep convolutional nets for document image classification and retrieval”. In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, pp. 991–995 (cit. on pp. 61, 63, 79, 94, 95).
- Hauser, Marc D, Noam Chomsky, and W Tecumseh Fitch (2002). “The faculty of language: what is it, who has it, and how did it evolve?” In: *science* 298.5598, pp. 1569–1579 (cit. on p. 7).
- Haviv, Adi, Ori Ram, Ofir Press, Peter Izsak, and Omer Levy (2022). “Transformer language models without positional encodings still learn positional information”. In: *arXiv preprint arXiv:2203.16634* (cit. on p. 106).
- He, Kaiming, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick (2022). “Masked autoencoders are scalable vision learners”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009 (cit. on p. 78).
- He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778 (cit. on p. 33).
- He, Pengcheng, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen (2020). “Deberta: Decoding-enhanced bert with disentangled attention”. In: *arXiv preprint arXiv:2006.03654* (cit. on p. 74).

- Hermann, Karl Moritz, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom (2015). "Teaching machines to read and comprehend". In: *Advances in neural information processing systems* 28 (cit. on p. 128).
- Hilbert, Martin and Priscila López (2011). "The world's technological capacity to store, communicate, and compute information". In: *science* 332.6025, pp. 60–65 (cit. on p. 1).
- Hochreiter, Sepp, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. (2001). *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies* (cit. on p. 23).
- Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780 (cit. on p. 23).
- Hofmann, Thomas (2001). "Unsupervised learning by probabilistic latent semantic analysis". In: *Machine learning* 42, pp. 177–196 (cit. on p. 21).
- Hong, Teakgyu, DongHyun Kim, Mingi Ji, Wonseok Hwang, Daehyun Nam, and Sungrae Park (2020). "Bros: A pre-trained language model for understanding texts in document". In: (cit. on pp. 70, 72, 75, 79).
- Huang, Cheng-Zhi Anna, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck (2018). "Music transformer". In: *arXiv preprint arXiv:1809.04281* (cit. on p. 38).
- Huang, Yupan, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei (2022). "Layoutlmv3: Pre-training for document ai with unified text and image masking". In: *Proceedings of the 30th ACM International Conference on Multimedia*, pp. 4083–4091 (cit. on pp. ix, 70, 72, 76, 77, 80).
- Huang, Zheng, Kai Chen, Jianhua He, Xiang Bai, Dimosthenis Karatzas, Shijian Lu, and CV Jawahar (2019). "Icdar2019 competition on scanned receipt ocr and information extraction". In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, pp. 1516–1520 (cit. on pp. 64, 117, 118).
- Hwang, Wonseok, Jinyeong Yim, Seunghyun Park, Sohee Yang, and Minjoon Seo (2020). "Spatial dependency parsing for semi-structured document information extraction". In: *arXiv preprint arXiv:2005.00642* (cit. on p. 68).
- Jaume, Guillaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran (2019). "Funsd: A dataset for form understanding in noisy scanned documents". In: *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*. Vol. 2. IEEE, pp. 1–6 (cit. on pp. 61, 63, 117, 118).
- Jiang, Jinhao, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen (2023). "Structgpt: A general framework for large language model to reason over structured data". In: *arXiv preprint arXiv:2305.09645* (cit. on p. 49).

- Jiao, Xiaoqi, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu (2019). "Tinybert: Distilling bert for natural language understanding". In: *arXiv preprint arXiv:1909.10351* (cit. on p. 43).
- Jones, Karen Spärck (2005). "Some points in a time". In: *Computational Linguistics* 31.1, pp. 1–14 (cit. on p. 11).
- Joshi, Mandar, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy (2020). "Spanbert: Improving pre-training by representing and predicting spans". In: *Transactions of the association for computational linguistics* 8, pp. 64–77 (cit. on pp. 42, 79).
- Jozefowicz, Rafal, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu (2016). "Exploring the limits of language modeling". In: *arXiv preprint arXiv:1602.02410* (cit. on p. 15).
- Kalchbrenner, Nal, Edward Grefenstette, and Phil Blunsom (2014). "A convolutional neural network for modelling sentences". In: *arXiv preprint arXiv:1404.2188* (cit. on p. 23).
- Kaplan, Jared, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei (2020). "Scaling laws for neural language models". In: *arXiv preprint arXiv:2001.08361* (cit. on pp. 28, 48).
- Kardas, Marcin, Piotr Czapla, Pontus Stenetorp, Sebastian Ruder, Sebastian Riedel, Ross Taylor, and Robert Stojnic (2020). "Axccl: Automatic extraction of results from machine learning papers". In: *arXiv preprint arXiv:2004.14356* (cit. on p. 64).
- Katti, Anoop Raveendra, Christian Reisswig, Cordula Guder, Sebastian Brarda, Steffen Bickel, Johannes Höhne, and Jean Baptiste Faddoul (2018). "Chargrid: Towards understanding 2d documents". In: *arXiv preprint arXiv:1809.08799* (cit. on p. 67).
- Katz, Slava (1987). "Estimation of probabilities from sparse data for the language model component of a speech recognizer". In: *IEEE transactions on acoustics, speech, and signal processing* 35.3, pp. 400–401 (cit. on p. 16).
- Kay, Anthony (July 2007). "Tesseract: An Open-Source Optical Character Recognition Engine". In: *Linux J.* 2007.159, p. 2 (cit. on pp. 71, 95, 108).
- Ke, Guolin, Di He, and Tie-Yan Liu (2020). "Rethinking positional encoding in language pre-training". In: *arXiv preprint arXiv:2006.15595* (cit. on p. 38).
- Kerroumi, Mohamed, Othmane Sayem, and Aymen Shabou (2021). "VisualWord-Grid: Information extraction from scanned documents using a multimodal approach". In: *International Conference on Document Analysis and Recognition*. Springer, pp. 389–402 (cit. on pp. 67, 68).
- Khandelwal, Urvashi, He He, Peng Qi, and Dan Jurafsky (2018). "Sharp nearby, fuzzy far away: How neural language models use context". In: *arXiv preprint arXiv:1805.04623* (cit. on p. 55).

- Kieras, David (1978). "Beyond pictures and words: Alternative information-processing models for imagery effect in verbal memory." In: *Psychological Bulletin* 85.3, p. 532 (cit. on p. 86).
- Kiesel, Johannes, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast (2019). "Semeval-2019 task 4: Hyperpartisan news detection". In: *Proceedings of the 13th International Workshop on Semantic Evaluation*, pp. 829–839 (cit. on p. 55).
- Kim, Geewook, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park (2022). "Ocr-free document understanding transformer". In: *European Conference on Computer Vision*. Springer, pp. 498–517 (cit. on pp. 70, 78, 106, 151).
- Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (cit. on p. 138).
- Kitaev, Nikita, Łukasz Kaiser, and Anselm Levskaya (2020). "Reformer: The Efficient Transformer". In: *arXiv preprint arXiv:2001.04451* (cit. on pp. 54, 93).
- Koh, Huan Yee, Jiaxin Ju, Ming Liu, and Shirui Pan (2022). "An empirical survey on long document summarization: Datasets, models, and metrics". In: *ACM computing surveys* 55.8, pp. 1–35 (cit. on p. 50).
- Kovačević, Aldin and Dino Kečo (2022). "Bidirectional LSTM networks for abstractive text summarization". In: *Advanced Technologies, Systems, and Applications VI: Proceedings of the International Symposium on Innovative and Interdisciplinary Applications of Advanced Technologies (IAT) 2021*. Springer, pp. 281–293 (cit. on p. 23).
- Kress, Gunther R, Theo Van Leeuwen, et al. (1996). *Reading images: The grammar of visual design*. Psychology Press (cit. on p. 86).
- Kumar, Jayant and David Doermann (2013). "Unsupervised classification of structurally similar document images". In: *2013 12th International Conference on Document Analysis and Recognition*. IEEE, pp. 1225–1229 (cit. on p. 63).
- Lai, Guokun, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy (2017). "Race: Large-scale reading comprehension dataset from examinations". In: *arXiv preprint arXiv:1704.04683* (cit. on p. 55).
- Lample, Guillaume and Alexis Conneau (2019). "Cross-lingual language model pretraining". In: *arXiv preprint arXiv:1901.07291* (cit. on p. 43).
- Lan, Zhenzhong, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut (2019). "Albert: A lite bert for self-supervised learning of language representations". In: *arXiv preprint arXiv:1909.11942* (cit. on p. 42).
- Laurençon, Hugo, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander M Rush, Douwe Kiela, et al. (2023). "Obelisc: An open web-scale filtered dataset of

- interleaved image-text documents". In: *arXiv preprint arXiv:2306.16527* (cit. on p. 151).
- Le, Hang, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoit Crabbé, Laurent Besacier, and Didier Schwab (2019). "Flaubert: Unsupervised language model pre-training for french". In: *arXiv preprint arXiv:1912.05372* (cit. on p. 42).
- Lebourgeois, Frank, Zbigniew Bublinski, and Hubert Emptoz (1992). "A fast and efficient method for extracting text paragraphs and graphics from unconstrained documents". In: *11th IAPR International Conference on Pattern Recognition. Vol. II. Conference B: Pattern Recognition Methodology and Systems. Vol. 1*. IEEE Computer Society, pp. 272–273 (cit. on p. 60).
- Leckner, Sara (2012). "Presentation factors affecting reading behaviour in readers of newspaper media: an eye-tracking perspective". In: *Visual Communication* 11.2, pp. 163–184 (cit. on p. 86).
- LeCun, Yann, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel (1989). "Backpropagation applied to handwritten zip code recognition". In: *Neural computation* 1.4, pp. 541–551 (cit. on p. 23).
- Lee, Chen-Yu, Chun-Liang Li, Timothy Dozat, Vincent Perot, Guolong Su, Nan Hua, Joshua Ainslie, Renshen Wang, Yasuhisa Fujii, and Tomas Pfister (2022). "Formnet: Structural encoding beyond sequential modeling in form document information extraction". In: *arXiv preprint arXiv:2203.08411* (cit. on pp. 70, 74, 75).
- Lee, Jinhyuk, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang (2020). "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: *Bioinformatics* 36.4, pp. 1234–1240 (cit. on p. 42).
- Lemarié, Julie, Hélène Eyrolle, and Jean-Marie Cellier (2008). "The segmented presentation of visually structured texts: Effects on text comprehension". In: *Computers in human behavior* 24.3, pp. 888–902 (cit. on p. 86).
- Lewis, David, Gady Agam, Shlomo Argamon, Ophir Frieder, D Grossman, and Jefferson Heard (2006). "Building a test collection for complex document information processing". In: *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 665–666 (cit. on pp. 78, 115).
- Lewis, Mike, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer (2019). "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension". In: *arXiv preprint arXiv:1910.13461* (cit. on pp. 45, 50, 58, 78, 114, 136, 149).

- Li, Chenliang, Bin Bi, Ming Yan, Wei Wang, Songfang Huang, Fei Huang, and Luo Si (2021). “Structurallm: Structural pre-training for form understanding”. In: *arXiv preprint arXiv:2105.11210* (cit. on pp. 70, 72, 78, 81).
- Li, Minghao, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li (2019). “Tablebank: A benchmark dataset for table detection and recognition”. In: *arXiv preprint arXiv:1903.01949* (cit. on p. 63).
- Li, Minghao, Yiheng Xu, Lei Cui, Shaohan Huang, Furu Wei, Zhoujun Li, and Ming Zhou (2020). “DocBank: A benchmark dataset for document layout analysis”. In: *arXiv preprint arXiv:2006.01038* (cit. on pp. 61, 62, 70, 87, 94–96).
- Li, Peizhao, Jiuxiang Gu, Jason Kuen, Vlad I Morariu, Handong Zhao, Rajiv Jain, Varun Manjunatha, and Hongfu Liu (2021). “Selfdoc: Self-supervised document representation learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5652–5660 (cit. on pp. ix, 70, 73, 76, 77, 79–81).
- Lin, Chin-Yew (2004). “Rouge: A package for automatic evaluation of summaries”. In: *Text summarization branches out*, pp. 74–81 (cit. on p. 18).
- Lin, Tsung-Yi, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie (2017). “Feature pyramid networks for object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125 (cit. on p. 77).
- Lin, Weihong, Qifang Gao, Lei Sun, Zhuoyao Zhong, Kai Hu, Qin Ren, and Qiang Huo (2021). “ViBERTgrid: a jointly trained multi-modal 2D document representation for key information extraction from documents”. In: *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part I* 16. Springer, pp. 548–563 (cit. on p. 68).
- Liu, Haotian, Chunyuan Li, Qingyang Wu, and Yong Jae Lee (2023). “Visual instruction tuning”. In: *arXiv preprint arXiv:2304.08485* (cit. on p. 151).
- Liu, Xiaojing, Feiyu Gao, Qiong Zhang, and Huasha Zhao (2019). “Graph convolution for multimodal information extraction from visually rich documents”. In: *arXiv preprint arXiv:1903.11279* (cit. on p. 68).
- Liu, Yinhan, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer (2020). “Multilingual denoising pre-training for neural machine translation”. In: *Transactions of the Association for Computational Linguistics* 8, pp. 726–742 (cit. on pp. 45, 78, 136).
- Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov (2019). “Roberta: A robustly optimized bert pretraining approach”. In: *arXiv preprint arXiv:1907.11692* (cit. on pp. 42, 78).

- Liu, Yixin, Ansong Ni, Linyong Nan, Budhaditya Deb, Chenguang Zhu, Ahmed H Awadallah, and Dragomir Radev (2022). “Leveraging locality in abstractive text summarization”. In: *arXiv preprint arXiv:2205.12476* (cit. on p. 52).
- Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo (2021). “Swin transformer: Hierarchical vision transformer using shifted windows”. In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022 (cit. on p. 78).
- Luo, Ling, Xiang Ao, Yan Song, Feiyang Pan, Min Yang, and Qing He (2019). “Reading like HER: Human reading inspired extractive summarization”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3033–3043 (cit. on p. 128).
- Luo, Siwen, Yihao Ding, Siqu Long, Josiah Poon, and Soyeon Caren Han (2022). “Doc-gcn: Heterogeneous graph convolutional networks for document layout analysis”. In: *arXiv preprint arXiv:2208.10970* (cit. on p. 67).
- Maas, Andrew, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts (2011). “Learning word vectors for sentiment analysis”. In: *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150 (cit. on p. 55).
- Manning, Christopher and Hinrich Schütze (1999). *Foundations of statistical natural language processing*. MIT press (cit. on p. 12).
- Marcus, Mitchell, Beatrice Santorini, and Mary Ann Marcinkiewicz (1993). “Building a large annotated corpus of English: The Penn Treebank”. In: (cit. on p. 47).
- Martin, Louis, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de La Clergerie, Djamé Seddah, and Benoît Sagot (2019). “CamemBERT: a tasty French language model”. In: *arXiv preprint arXiv:1911.03894* (cit. on p. 42).
- Mathew, Minesh, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar (2022). “InfographicVQA”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1697–1706 (cit. on p. 65).
- Mathew, Minesh, Dimosthenis Karatzas, and CV Jawahar (2021). “Docvqa: A dataset for vqa on document images”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2200–2209 (cit. on pp. 61, 65).
- Merity, Stephen, Caiming Xiong, James Bradbury, and Richard Socher (2016). “Pointer sentinel mixture models”. In: *arXiv preprint arXiv:1609.07843* (cit. on p. 47).
- Meyer, Bonnie JF, David M Brandt, and George J Bluth (1980). “Use of top-level structure in text: Key for reading comprehension of ninth-grade students”. In: *Reading research quarterly*, pp. 72–103 (cit. on pp. 69, 86).

- Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean (2013). "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (cit. on pp. 21, 24, 25, 27).
- Mikolov, Tomas, Martin Karafiat, Lukas Burget, Jan Cernocky, and Sanjeev Khudanpur (2010). "Recurrent neural network based language model." In: *Inter-speech*. Vol. 2. 3. Makuhari, pp. 1045–1048 (cit. on p. 23).
- Mindee (2021). *docTR: Document Text Recognition*. <https://github.com/mindee/doctr> (cit. on pp. 108, 116).
- Mohamadi, Salman, Ghulam Mujtaba, Ngan Le, Gianfranco Doretto, and Donald A Adjeroh (2023). "Chatgpt in the age of generative ai and large language models: A concise survey". In: *arXiv preprint arXiv:2307.04251* (cit. on p. 48).
- Nakayama, Hiroki (2018). *seqeval: A Python framework for sequence labeling evaluation*. Software available from <https://github.com/chakki-works/seqeval>. URL: <https://github.com/chakki-works/seqeval> (cit. on p. 121).
- Nguyen, Laura, Thomas Scialom, Benjamin Piwowarski, and Jacopo Staiano (May 2023). "LoRaLay: A Multilingual and Multimodal Dataset for Long Range and Layout-Aware Summarization". In: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Dubrovnik, Croatia: Association for Computational Linguistics, pp. 636–651. URL: <https://aclanthology.org/2023.eacl-main.46> (cit. on p. 125).
- Nguyen, Laura, Thomas Scialom, Jacopo Staiano, and Benjamin Piwowarski (Nov. 2021). "Skim-Attention: Learning to Focus via Document Layout". In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 2413–2427. URL: <https://aclanthology.org/2021.findings-emnlp.207> (cit. on p. 85).
- Nguyen, Thien Huu and Ralph Grishman (2015). "Relation extraction: Perspective from convolutional neural networks". In: *Proceedings of the 1st workshop on vector space modeling for natural language processing*, pp. 39–48 (cit. on p. 23).
- Olive, Thierry and Marie-Laure Barbier (2017). "Processing time and cognitive effort of longhand note taking when reading and summarizing a structured or linear text". In: *Written Communication* 34.2, pp. 224–246 (cit. on p. 86).
- Oliveira, Sofia Ares, Benoit Seguin, and Frederic Kaplan (2018). "dhSegment: A generic deep-learning approach for document segmentation". In: *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, pp. 7–12 (cit. on p. 67).
- Omar, Nazlia and Qasem Al-Tashi (2018). "Arabic nested noun compound extraction based on linguistic features and statistical measures". In: *GEMA Online® Journal of Language Studies* 18.2 (cit. on p. 15).
- Ouyang, Long, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. (2022). "Training language models to follow instructions with human feed-

- back". In: *Advances in Neural Information Processing Systems* 35, pp. 27730–27744 (cit. on p. 46).
- Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu (2002). "Bleu: a method for automatic evaluation of machine translation". In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318 (cit. on pp. 18, 141).
- Park, Seunghyun, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee (2019). "CORD: a consolidated receipt dataset for post-OCR parsing". In: *Workshop on Document Intelligence at NeurIPS 2019* (cit. on pp. 64, 117, 118).
- Parmar, Niki, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran (2018). "Image transformer". In: *International conference on machine learning*. PMLR, pp. 4055–4064 (cit. on p. 55).
- Pasupat, Panupong and Percy Liang (2015). "Compositional semantic parsing on semi-structured tables". In: *arXiv preprint arXiv:1508.00305* (cit. on p. 65).
- Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer (2017). "Automatic differentiation in PyTorch". In: (cit. on pp. 118, 138).
- Paulus, Romain, Caiming Xiong, and Richard Socher (2017). "A deep reinforced model for abstractive summarization". In: *arXiv preprint arXiv:1705.04304* (cit. on p. 44).
- Peng, Qiming, Yinxu Pan, Wenjin Wang, Bin Luo, Zhenyu Zhang, Zhengjie Huang, Teng Hu, Weichong Yin, Yongfeng Chen, Yin Zhang, et al. (2022). "Ernie-layout: Layout knowledge enhanced pre-training for visually-rich document understanding". In: *arXiv preprint arXiv:2210.06155* (cit. on pp. 60, 70, 72, 74, 75, 78, 79, 106, 113).
- Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543 (cit. on p. 25).
- Perot, Vincent, Kai Kang, Florian Luisier, Guolong Su, Xiaoyu Sun, Ramya Sree Boppana, Zilong Wang, Jiaqi Mu, Hao Zhang, and Nan Hua (2023). "LMDX: Language Model-based Document Information Extraction and Localization". In: *arXiv preprint arXiv:2309.10952* (cit. on p. 151).
- Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer (June 2018). "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. URL: <https://aclanthology.org/N18-1202> (cit. on p. 27).

- Petroni, Fabio, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Mail-lard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel (June 2021). "KILT: a Benchmark for Knowledge Intensive Language Tasks". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, pp. 2523–2544. URL: <https://aclanthology.org/2021.naacl-main.200> (cit. on p. 11).
- Pham, Hai, Guoxin Wang, Yijuan Lu, Dinei Florencio, and Cha Zhang (2022). "Understanding long documents with different position-aware attentions". In: *arXiv preprint arXiv:2208.08201* (cit. on pp. 82, 127).
- Pham, Ngoc-Quan, German Kruszewski, and Gemma Boleda (2016). "Convolutional neural network language models". In: *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 1153–1162 (cit. on p. 24).
- Powalski, Rafał, Łukasz Borchmann, Dawid Jurkiewicz, Tomasz Dwojak, Michał Pietruszka, and Gabriela Pałka (2021). "Going Full-TILT Boogie on Document Understanding with Text-Image-Layout Transformer". In: *arXiv preprint arXiv:2102.09550* (cit. on pp. 70, 74, 78, 79, 106, 150).
- Pramanik, Subhojeet, Shashank Mujumdar, and Hima Patel (2020). "Towards a Multi-modal, Multi-task Learning based Pre-training Framework for Document Representation Learning". In: *arXiv preprint arXiv:2009.14457* (cit. on pp. 70, 82, 127).
- Prasad, Devashish, Ayan Gadpal, Kshitij Kapadni, Manish Visave, and Kavita Sultantpure (2020). "CascadeTabNet: An approach for end to end table detection and structure recognition from image-based documents". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 572–573 (cit. on p. 67).
- Qin, Guanghui, Yukun Feng, and Benjamin Van Durme (2022). "The nlp task effectiveness of long-range transformers". In: *arXiv preprint arXiv:2202.07856* (cit. on pp. 51, 56, 57).
- Radford, Alec, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. (2018). "Improving language understanding by generative pre-training". In: (cit. on pp. 27, 28, 46, 47).
- Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. (2019). "Language models are unsupervised multitask learners". In: *OpenAI blog* 1.8, p. 9 (cit. on pp. 47, 147).
- Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu (2020). "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *The Journal of Machine Learning Research* 21.1, pp. 5485–5551 (cit. on pp. 38, 46, 47, 73, 78, 81, 150).

- Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang (2016). "Squad: 100,000+ questions for machine comprehension of text". In: *arXiv preprint arXiv:1606.05250* (cit. on p. 11).
- Ramesh, Aditya, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever (2021). "Zero-shot text-to-image generation". In: *International Conference on Machine Learning*. PMLR, pp. 8821–8831 (cit. on p. 80).
- Ramshaw, Lance A and Mitchell P Marcus (1999). "Text chunking using transformation-based learning". In: *Natural language processing using very large corpora*. Springer, pp. 157–176 (cit. on pp. 105, 120).
- Ranzato, Marc'Aurelio, Sumit Chopra, Michael Auli, and Wojciech Zaremba (2015). "Sequence level training with recurrent neural networks". In: *arXiv preprint arXiv:1511.06732* (cit. on p. 44).
- Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun (2015). "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems* 28 (cit. on pp. 66, 67, 73, 76).
- Rosenfeld, Ronald (2000). "Two decades of statistical language modeling: Where do we go from here?" In: *Proceedings of the IEEE* 88.8, pp. 1270–1278 (cit. on p. 16).
- Sage, Clément, Alex Aussem, Véronique Eglin, Haytham Elghazel, and Jérémy Espinas (2020). "End-to-end extraction of structured information from business documents with pointer-generator networks". In: *Proceedings of the fourth workshop on structured prediction for NLP*, pp. 43–52 (cit. on p. 106).
- Sanh, Victor, Lysandre Debut, Julien Chaumond, and Thomas Wolf (2019). "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *arXiv preprint arXiv:1910.01108* (cit. on p. 43).
- Scao, Teven Le, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. (2022). "Bloom: A 176b-parameter open-access multilingual language model". In: *arXiv preprint arXiv:2211.05100* (cit. on p. 48).
- Schwenk, Holger (2007). "Continuous space language models". In: *Computer Speech & Language* 21.3, pp. 492–518 (cit. on p. 23).
- Scialom, Thomas, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano (2020). "MLSUM: The multilingual summarization corpus". In: *arXiv preprint arXiv:2004.14900* (cit. on p. 128).
- Scialom, Thomas, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano (2019). "Answers unite! unsupervised metrics for reinforced summarization models". In: *arXiv preprint arXiv:1909.01610* (cit. on p. 44).
- See, Abigail, Peter J Liu, and Christopher D Manning (2017). "Get to the point: Summarization with pointer-generator networks". In: *arXiv preprint arXiv:1704.04368* (cit. on p. 106).

- Shaham, Uri, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, et al. (2022). “Scrolls: Standardized comparison over long language sequences”. In: *arXiv preprint arXiv:2201.03533* (cit. on p. 56).
- Sharma, Eva, Chen Li, and Lu Wang (2019). “BIGPATENT: A large-scale dataset for abstractive and coherent summarization”. In: *arXiv preprint arXiv:1906.03741* (cit. on pp. 50, 128, 134).
- Shaw, Peter, Jakob Uszkoreit, and Ashish Vaswani (2018). “Self-attention with relative position representations”. In: *arXiv preprint arXiv:1803.02155* (cit. on p. 38).
- Shazeer, Noam, Ryan Doherty, Colin Evans, and Chris Waterson (2016). “Swivel: Improving embeddings by noticing what’s missing”. In: *arXiv preprint arXiv:1602.02215* (cit. on pp. 21, 25).
- Shazeer, Noam and Mitchell Stern (2018). “Adafactor: Adaptive learning rates with sublinear memory cost”. In: *International Conference on Machine Learning*. PMLR, pp. 4596–4604 (cit. on p. 138).
- Shen, Zejiang, Kyle Lo, Lucy Lu Wang, Bailey Kuehl, Daniel S Weld, and Doug Downey (2022). “VILA: Improving structured content extraction from scientific PDFs using visual layout groups”. In: *Transactions of the Association for Computational Linguistics* 10, pp. 376–392 (cit. on pp. 70, 73).
- Sheng, Emily, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng (2021). “Societal biases in language generation: Progress and challenges”. In: *arXiv preprint arXiv:2105.04054* (cit. on p. 49).
- Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts (2013). “Recursive deep models for semantic compositionality over a sentiment treebank”. In: *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642 (cit. on p. 55).
- Soto, Carlos and Shinjae Yoo (2019). “Visual detection with context for document layout analysis”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3464–3470 (cit. on p. 67).
- Sun, Simeng, Kalpesh Krishna, Andrew Mattarella-Micke, and Mohit Iyyer (2021). “Do long-range language models actually use long-range context?” In: *arXiv preprint arXiv:2109.09115* (cit. on pp. 55, 57).
- Sutskever, Ilya, Oriol Vinyals, and Quoc V Le (2014). “Sequence to sequence learning with neural networks”. In: *Advances in neural information processing systems* 27 (cit. on p. 29).
- Tanaka, Ryota, Kyosuke Nishida, and Sen Yoshida (2021). “Visualmrc: Machine reading comprehension on document images”. In: *Proceedings of the AAAI*

- Conference on Artificial Intelligence*. Vol. 35. 15, pp. 13878–13888 (cit. on pp. 65, 70, 78).
- Tang, Yuqing, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan (2020). “Multilingual translation with extensible multilingual pretraining and finetuning”. In: *arXiv preprint arXiv:2008.00401* (cit. on p. 136).
- Tang, Zineng, Ziyi Yang, Guoxin Wang, Yuwei Fang, Yang Liu, Chenguang Zhu, Michael Zeng, Cha Zhang, and Mohit Bansal (2023). “Unifying vision, text, and layout for universal document processing”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19254–19264 (cit. on pp. 70, 73, 77, 78, 80, 150).
- Tay, Yi, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler (2020a). “Long Range Arena: A Benchmark for Efficient Transformers”. In: *arXiv preprint arXiv:2011.04006* (cit. on p. 55).
- Tay, Yi, Mostafa Dehghani, Dara Bahri, and Donald Metzler (2020b). “Efficient Transformers: A Survey”. In: *arXiv preprint arXiv:2009.06732* (cit. on p. 51).
- Thede, Scott M and Mary Harper (1999). “A second-order hidden Markov model for part-of-speech tagging”. In: *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, pp. 175–182 (cit. on p. 16).
- Thirunavukarasu, Arun James, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting (2023). “Large language models in medicine”. In: *Nature medicine*, pp. 1–11 (cit. on pp. 12, 48).
- Toprak, Elif and Gamze Almacioğlu (2009). “Three reading phases and their applications in the teaching of english as a foreign language in reading classes with young learners”. In: *Journal of language and Linguistic Studies* 5.1 (cit. on p. 128).
- Touvron, Hugo, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. (2023). “Llama 2: Open foundation and fine-tuned chat models”. In: *arXiv preprint arXiv:2307.09288* (cit. on pp. 46, 48, 147).
- Townsend, Benjamin, Eamon Ito-Fisher, Lily Zhang, and Madison May (2021). “Doc2dict: Information extraction as text generation”. In: *arXiv preprint arXiv:2105.07510* (cit. on p. 106).
- Turing, A. M. (1950). “Computing Machinery and Intelligence”. English. In: *Mind*. New Series 59.236, pp. 433–460. URL: <http://www.jstor.org/stable/2251299> (cit. on p. 7).
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). “Attention is all you need”. In: *Advances in neural information processing systems* 30 (cit. on pp. 3, 28, 29, 32, 37, 69, 106, 127, 147).

- Villalobos, Pablo, Jaime Sevilla, Lennart Heim, Tamay Besiroglu, Marius Hobbhahn, and Anson Ho (2022). "Will we run out of data? An analysis of the limits of scaling datasets in Machine Learning". In: *arXiv preprint arXiv:2211.04325* (cit. on p. 1).
- Von Platen, Patrick (2020). "How to generate text: using different decoding methods for language generation with Transformers". "Website". <https://huggingface.co/blog/how-to-generate#how-to-generate-text-using-different-decoding-methods-for-language-generation-with-transformers> (cit. on p. 44).
- Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman (2018). "GLUE: A multi-task benchmark and analysis platform for natural language understanding". In: *arXiv preprint arXiv:1804.07461* (cit. on pp. 11, 55).
- Wang, Dongsheng, Natraj Raman, Mathieu Sibue, Zhiqiang Ma, Petr Babkin, Simerjot Kaur, Yulong Pei, Armineh Nourbakhsh, and Xiaomo Liu (2023). "DocLLM: A layout-aware generative language model for multimodal document understanding". In: *arXiv preprint arXiv:2401.00908* (cit. on p. 151).
- Wang, Guoxin, Yijuan Lu, Lei Cui, Tengchao Lv, Dinei Florencio, and Cha Zhang (2022). "A Simple yet Effective Learnable Positional Encoding Method for Improving Document Transformer Model". In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2022*, pp. 453–463 (cit. on p. 75).
- Wang, Hanchen, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. (2023). "Scientific discovery in the age of artificial intelligence". In: *Nature* 620.7972, pp. 47–60 (cit. on pp. 12, 48).
- Wang, Jiapeng, Lianwen Jin, and Kai Ding (2022). "Lilt: A simple yet effective language-independent layout transformer for structured document understanding". In: *arXiv preprint arXiv:2202.13669* (cit. on pp. 70, 74).
- Wang, Jiapeng, Chongyu Liu, Lianwen Jin, Guozhi Tang, Jiaxin Zhang, Shuaitao Zhang, Qianying Wang, Yaqiang Wu, and Mingxiang Cai (2021). "Towards robust visual information extraction in real world: New dataset and novel solution". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 4, pp. 2738–2745 (cit. on p. 64).
- Wang, Junjie, Yuchao Huang, Chunyang Chen, Zhe Liu, Song Wang, and Qing Wang (2023). "Software testing with large language model: Survey, landscape, and vision". In: *arXiv preprint arXiv:2307.07221* (cit. on p. 48).
- Wang, Sinong, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma (2020). "Linformer: Self-Attention with Linear Complexity". In: *arXiv preprint arXiv:2006.04768* (cit. on p. 54).

- Wang, Zhiguo, Wael Hamza, and Radu Florian (2017). “Bilateral multi-perspective matching for natural language sentences”. In: *arXiv preprint arXiv:1702.03814* (cit. on p. 55).
- Wang, Zilong, Yiheng Xu, Lei Cui, Jingbo Shang, and Furu Wei (2021). *LayoutReader: Pre-training of Text and Layout for Reading Order Detection*. arXiv: 2108.11591 [cs.CL] (cit. on pp. 75, 108, 116).
- Wang, Zilong, Yichao Zhou, Wei Wei, Chen-Yu Lee, and Sandeep Tata (2023). “Vrdu: A benchmark for visually-rich document understanding”. In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5184–5193 (cit. on p. 64).
- Wehrmann, Joonatas, Willian Becker, Henry EL Cagnini, and Rodrigo C Barros (2017). “A character-based convolutional neural network for language-agnostic Twitter sentiment analysis”. In: *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 2384–2391 (cit. on p. 13).
- Wei, Hao, Micheal Baechler, Fouad Slimane, and Rolf Ingold (2013). “Evaluation of SVM, MLP and GMM classifiers for layout analysis of historical documents”. In: *2013 12th International Conference on Document Analysis and Recognition*. IEEE, pp. 1220–1224 (cit. on p. 60).
- Wei, Jason, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le (2021). “Finetuned language models are zero-shot learners”. In: *arXiv preprint arXiv:2109.01652* (cit. on p. 151).
- Welbl, Johannes, Pontus Stenetorp, and Sebastian Riedel (2018). “Constructing datasets for multi-hop reading comprehension across documents”. In: *Transactions of the Association for Computational Linguistics* 6, pp. 287–302 (cit. on p. 55).
- Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. (2019). “Huggingface’s transformers: State-of-the-art natural language processing”. In: *arXiv preprint arXiv:1910.03771* (cit. on pp. 99, 118, 138).
- Wong, Kwan Y., Richard G. Casey, and Friedrich M. Wahl (1982). “Document analysis system”. In: *IBM journal of research and development* 26.6, pp. 647–656 (cit. on p. 60).
- Wright, Patricia (1999). “The psychology of layout: Consequences of the visual structure of documents”. In: *American Association for Artificial Intelligence Technical Report FS-99-04*, pp. 1–9 (cit. on pp. 69, 86).
- Wu, Te-Lin, Cheng Li, Mingyang Zhang, Tao Chen, Spurthi Amba Hombaiah, and Michael Bendersky (2021). “Lampret: Layout-aware multimodal pretraining for document understanding”. In: *arXiv preprint arXiv:2104.08405* (cit. on pp. 70, 73, 77, 79).

- Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. (2016). “Google’s neural machine translation system: Bridging the gap between human and machine translation”. In: *arXiv preprint arXiv:1609.08144* (cit. on pp. 14, 29).
- Xie, Saining, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He (2017). “Aggregated residual transformations for deep neural networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500 (cit. on p. 77).
- Xiong, Lee, Chuan Hu, Chenyan Xiong, Daniel Campos, and Arnold Overwijk (2019). “Open domain web keyphrase extraction beyond language modeling”. In: *arXiv preprint arXiv:1911.02671* (cit. on p. 55).
- Xu, Frank F, Uri Alon, Graham Neubig, and Vincent Josua Hellendoorn (2022). “A systematic evaluation of large language models of code”. In: *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, pp. 1–10 (cit. on pp. 12, 48).
- Xu, Yang, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. (2020). “LayoutLMv2: Multimodal Pre-training for Visually-Rich Document Understanding”. In: *arXiv preprint arXiv:2012.14740* (cit. on pp. 70, 73, 76, 78, 79, 106).
- Xu, Yiheng, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou (2020). “Layoutlm: Pre-training of text and layout for document image understanding”. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1192–1200 (cit. on pp. ix, 2, 60, 70, 71, 75, 76, 78, 79, 89, 96, 97, 100, 106, 123, 136, 147).
- Xu, Yiheng, Tengchao Lv, Lei Cui, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, and Furu Wei (2021). “Layoutxlm: Multimodal pre-training for multilingual visually-rich document understanding”. In: *arXiv preprint arXiv:2104.08836* (cit. on p. 81).
- Xu, Yiheng, Tengchao Lv, Lei Cui, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, and Furu Wei (May 2022). “XFUND: A Benchmark Dataset for Multilingual Visually Rich Form Understanding”. In: *Findings of the Association for Computational Linguistics: ACL 2022*. Dublin, Ireland: Association for Computational Linguistics, pp. 3214–3224. URL: <https://aclanthology.org/2022.findings-acl.253> (cit. on pp. 64, 70).
- Yang, Xiao, Ersin Yumer, Paul Asente, Mike Kraley, Daniel Kifer, and C Lee Giles (2017). “Learning to extract semantic structure from documents using multimodal fully convolutional neural networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5315–5324 (cit. on p. 66).
- Yang, Zhilin, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le (2019). “Xlnet: Generalized autoregressive pretraining for

- language understanding". In: *Advances in neural information processing systems* 32 (cit. on p. 53).
- Yang, Zhilin, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning (2018). "HotpotQA: A dataset for diverse, explainable multi-hop question answering". In: *arXiv preprint arXiv:1809.09600* (cit. on p. 55).
- Yao, Yunzhi, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang (2023). "Editing Large Language Models: Problems, Methods, and Opportunities". In: *arXiv preprint arXiv:2305.13172* (cit. on p. 49).
- Ye, Junjie, Xuantang Chen, Nuo Xu, Can Zu, Zekai Shao, Shichun Liu, Yuhan Cui, Zeyang Zhou, Chao Gong, Yang Shen, et al. (2023). "A comprehensive capability analysis of gpt-3 and gpt-3.5 series models". In: *arXiv preprint arXiv:2303.10420* (cit. on p. 49).
- Ye, Qinghao, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. (2023). "mplug-owl: Modularization empowers large language models with multimodality". In: *arXiv preprint arXiv:2304.14178* (cit. on p. 151).
- Yu, Wenwen, Ning Lu, Xianbiao Qi, Ping Gong, and Rong Xiao (2021). "PICK: processing key information extraction from documents using improved graph learning-convolutional networks". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, pp. 4363–4370 (cit. on p. 68).
- Zaheer, Manzil, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. (2020). "Big bird: Transformers for longer sequences". In: *Advances in Neural Information Processing Systems* 33 (cit. on pp. 51, 52, 58, 78, 137–139).
- Zhang, Chong, Ya Guo, Yi Tu, Huan Chen, Jinyang Tang, Huijia Zhu, Qi Zhang, and Tao Gui (2023). "Reading Order Matters: Information Extraction from Visually-rich Documents by Token Path Prediction". In: *arXiv preprint arXiv:2310.11016* (cit. on p. 73).
- Zhang, Jingqing, Yao Zhao, Mohammad Saleh, and Peter Liu (2020). "Pegasus: Pre-training with extracted gap-sentences for abstractive summarization". In: *International Conference on Machine Learning*. PMLR, pp. 11328–11339 (cit. on pp. 45, 136–138).
- Zhang, Ming, Chengzhang Li, Meilin Wan, Xuejun Zhang, and Qingwei Zhao (2024). "ROUGE-SEM: Better evaluation of summarization using ROUGE combined with semantics". In: *Expert Systems with Applications* 237, p. 121364 (cit. on p. 141).
- Zhao, Wayne Xin, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. (2023). "A

- survey of large language models". In: *arXiv preprint arXiv:2303.18223* (cit. on p. 48).
- Zhong, Xu, Elaheh ShafieiBavani, and Antonio Jimeno Yepes (2020). "Image-based table recognition: data, model, and evaluation". In: *European conference on computer vision*. Springer, pp. 564–580 (cit. on p. 63).
- Zhong, Xu, Jianbin Tang, and Antonio Jimeno Yepes (2019). "Publaynet: largest dataset ever for document layout analysis". In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, pp. 1015–1022 (cit. on pp. 62, 94, 95).
- Zhu, Yukun, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler (2015). "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books". In: *Proceedings of the IEEE international conference on computer vision*, pp. 19–27 (cit. on p. 40).

BIBLIOGRAPHY

- Adel, Heike, Benjamin Roth, and Hinrich Schütze (2016). “Comparing convolutional neural networks to traditional models for slot filling”. In: *arXiv preprint arXiv:1603.05157* (cit. on p. 23).
- Afzal, Muhammad Zeshan, Samuele Capobianco, Muhammad Imran Malik, Simone Marinai, Thomas M Breuel, Andreas Dengel, and Marcus Liwicki (2015). “Deepdocclassifier: Document classification with deep convolutional neural network”. In: *2015 13th international conference on document analysis and recognition (ICDAR)*. IEEE, pp. 1111–1115 (cit. on p. 66).
- Ainslie, Joshua, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang (2020). “ETC: Encoding long and structured inputs in transformers”. In: *arXiv preprint arXiv:2004.08483* (cit. on p. 52).
- Antonacopoulos, Apostolos, Christian Clausner, Christos Papadopoulos, and Stefan Pletschacher (2013). “Icdar 2013 competition on historical newspaper layout analysis (hnlA 2013)”. In: *2013 12th International Conference on Document Analysis and Recognition*. IEEE, pp. 1454–1458 (cit. on p. 62).
- Appalaraju, Srikar, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R Manmatha (2021). “DocFormer: End-to-End Transformer for Document Understanding”. In: *arXiv preprint arXiv:2106.11539* (cit. on pp. ix, 70, 72, 74, 76, 77, 79, 80).
- Arisoy, Ebru, Tara N Sainath, Brian Kingsbury, and Bhuvana Ramabhadran (2012). “Deep neural network language models”. In: *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pp. 20–28 (cit. on p. 16).
- Baechler, Micheal and Rolf Ingold (2011). “Multi resolution layout analysis of medieval manuscripts using dynamic mlp”. In: *2011 International Conference on Document Analysis and Recognition*. IEEE, pp. 1185–1189 (cit. on p. 60).
- Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). “Neural machine translation by jointly learning to align and translate”. In: *arXiv preprint arXiv:1409.0473* (cit. on pp. 29–32).
- Bahl, Lalit R, Peter F Brown, Peter V de Souza, and Robert L Mercer (1989). “A tree-based statistical language model for natural language speech recognition”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37:7, pp. 1001–1008 (cit. on p. 16).
- Banerjee, Satanjeev and Alon Lavie (2005). “METEOR: An automatic metric for MT evaluation with improved correlation with human judgments”. In: *Proceed-*

- ings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pp. 65–72 (cit. on p. 141).
- Bang, Yejin, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. (2023). “A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity”. In: *arXiv preprint arXiv:2302.04023* (cit. on p. 49).
- Bao, Hangbo, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, et al. (2020). “Unilmv2: Pseudo-masked language models for unified language model pre-training”. In: *International conference on machine learning*. PMLR, pp. 642–652 (cit. on p. 78).
- Beltagy, Iz, Arman Cohan, and Kyle Lo (2019). “Scibert: Pretrained contextualized embeddings for scientific text”. In: *arXiv preprint arXiv:1903.10676* 1.1.3, p. 8 (cit. on p. 42).
- Beltagy, Iz, Matthew E Peters, and Arman Cohan (2020). “Longformer: The long-document transformer”. In: *arXiv preprint arXiv:2004.05150* (cit. on pp. 52, 82, 96, 97, 100, 127).
- Bengio, Yoshua, Réjean Ducharme, and Pascal Vincent (2000). “A neural probabilistic language model”. In: *Advances in neural information processing systems* 13 (cit. on pp. 20, 21, 24).
- Binmakhashen, Galal M and Sabri A Mahmoud (2019). “Document layout analysis: a comprehensive survey”. In: *ACM Computing Surveys (CSUR)* 52.6, pp. 1–36 (cit. on p. 62).
- Blei, David M, Andrew Y Ng, and Michael I Jordan (2003). “Latent dirichlet allocation”. In: *Journal of machine Learning research* 3, Jan, pp. 993–1022 (cit. on p. 21).
- Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov (2017). “Enriching word vectors with subword information”. In: *Transactions of the association for computational linguistics* 5, pp. 135–146 (cit. on p. 25).
- Borchmann, Łukasz, Michał Pietruszka, Tomasz Stanislawek, Dawid Jurkiewicz, Michał Turski, Karolina Szyndler, and Filip Graliński (2021). “DUE: End-to-End Document Understanding Benchmark”. In: *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)* (cit. on pp. 3, 64, 65).
- Britton, Bruce K, Shawn M Glynn, Bonnie J Meyer, and MJ Penland (1982). “Effects of text structure on use of cognitive capacity during reading.” In: *Journal of Educational Psychology* 74.1, p. 51 (cit. on p. 86).
- Brown, Peter F, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer (1992). “Class-based n-gram models of natural language”. In: *Computational linguistics* 18.4, pp. 467–480 (cit. on p. 15).
- Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda

- Aspell, et al. (2020). “Language models are few-shot learners”. In: *Advances in neural information processing systems* 33, pp. 1877–1901 (cit. on pp. 46, 48, 58, 151).
- Burt, Peter J and Edward H Adelson (1987). “The Laplacian pyramid as a compact image code”. In: *Readings in computer vision*. Elsevier, pp. 671–679 (cit. on p. 66).
- Cai, Zhaowei and Nuno Vasconcelos (2018). “Cascade r-cnn: Delving into high quality object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6154–6162 (cit. on pp. 66, 67).
- Cao, Shuyang and Lu Wang (2022). “Hibrids: Attention with hierarchical biases for structure-aware long document summarization”. In: *arXiv preprint arXiv:2203.10741* (cit. on p. 128).
- Chaudhari, Sneha, Varun Mithal, Gungor Polatkan, and Rohan Ramanath (2021). “An attentive survey of attention models”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 12.5, pp. 1–32 (cit. on p. 31).
- Chelba, Ciprian, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson (2013). “One billion word benchmark for measuring progress in statistical language modeling”. In: *arXiv preprint arXiv:1312.3005* (cit. on p. 47).
- Child, Rewon, Scott Gray, Alec Radford, and Ilya Sutskever (2019). “Generating long sequences with sparse transformers”. In: *arXiv preprint arXiv:1904.10509* (cit. on p. 52).
- Cho, Jaemin, Jiasen Lu, Dustin Schwenk, Hannaneh Hajishirzi, and Aniruddha Kembhavi (2020). “X-lxmert: Paint, caption and answer questions with multi-modal transformers”. In: *arXiv preprint arXiv:2009.11278* (cit. on p. 80).
- Cho, Kyunghyun, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio (2014). “On the properties of neural machine translation: Encoder-decoder approaches”. In: *arXiv preprint arXiv:1409.1259* (cit. on p. 30).
- Choromanski, Krzysztof, Valerii Likhoshesterov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. (2020). “Rethinking attention with performers”. In: *arXiv preprint arXiv:2009.14794* (cit. on p. 54).
- Chowdhery, Aakanksha, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. (2022). “Palm: Scaling language modeling with pathways”. In: *arXiv preprint arXiv:2204.02311* (cit. on p. 48).
- Clausner, Christian, Stefan Pletschacher, and Apostolos Antonopoulos (2013). “The significance of reading order in document recognition and its evaluation”. In: *2013 12th International Conference on Document Analysis and Recognition*. IEEE, pp. 688–692 (cit. on p. 104).

- Clissa, Luca (2022). “Survey of Big Data sizes in 2021”. In: *arXiv preprint arXiv:2202.07659* (cit. on p. 1).
- Cohan, Arman, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian (2018). “A discourse-aware attention model for abstractive summarization of long documents”. In: *arXiv preprint arXiv:1804.05685* (cit. on pp. 50, 128–131, 134, 135).
- Collobert, Ronan (2011). “Deep learning for efficient discriminative parsing”. In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, pp. 224–232 (cit. on p. 23).
- Collobert, Ronan, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa (2011). “Natural language processing (almost) from scratch”. In: *Journal of machine learning research* 12.ARTICLE, pp. 2493–2537 (cit. on p. 24).
- Conneau, Alexis, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov (2019). “Unsupervised cross-lingual representation learning at scale”. In: *arXiv preprint arXiv:1911.02116* (cit. on p. 43).
- Dai, Zihang, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov (2019). “Transformer-xl: Attentive language models beyond a fixed-length context”. In: *arXiv preprint arXiv:1901.02860* (cit. on pp. 38, 53).
- Dang, Hoa Trang (2005). “Overview of DUC 2005”. In: *Proceedings of the document understanding conference*. Vol. 2005, pp. 1–12 (cit. on p. 143).
- Das, Arindam, Saikat Roy, Ujjwal Bhattacharya, and Swapan K Parui (2018). “Document image classification with intra-domain transfer learning and stacked generalization of deep convolutional neural networks”. In: *2018 24th international conference on pattern recognition (ICPR)*. IEEE, pp. 3180–3185 (cit. on p. 66).
- Dasigi, Pradeep, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A Smith, and Matt Gardner (2021). “A dataset of information-seeking questions and answers anchored in research papers”. In: *arXiv preprint arXiv:2105.03011* (cit. on p. 50).
- Dauphinee, Tyler, Nikunj Patel, and Mohammad Rashidi (2019). “Modular multi-modal architecture for document classification”. In: *arXiv preprint arXiv:1912.04376* (cit. on p. 66).
- Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei (2009). “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255 (cit. on p. 66).
- Denk, Timo I and Christian Reisswig (2019). “Bertgrid: Contextualized embedding for 2d document representation and understanding”. In: *arXiv preprint arXiv:1909.04948* (cit. on pp. 67, 68).
- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). “Bert: Pre-training of deep bidirectional transformers for language understanding”.

- In: *arXiv preprint arXiv:1810.04805* (cit. on pp. 27, 28, 37, 39, 42, 58, 78, 96, 97, 100, 118, 147).
- Ding, Ming, Chang Zhou, Hongxia Yang, and Jie Tang (2020). “Cogltx: Applying bert to long texts”. In: *Advances in Neural Information Processing Systems* 33, pp. 12792–12804 (cit. on p. 50).
- Dong, Haoyu, Shijie Liu, Shi Han, Zhouyu Fu, and Dongmei Zhang (2019). “Tablesense: Spreadsheet table detection with convolutional neural networks”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01, pp. 69–76 (cit. on p. 67).
- Dong, Li, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon (2019). “Unified language model pre-training for natural language understanding and generation”. In: *Advances in neural information processing systems* 32 (cit. on p. 42).
- Dong, Zican, Tianyi Tang, Lunyi Li, and Wayne Xin Zhao (2023). “A survey on long text modeling with transformers”. In: *arXiv preprint arXiv:2302.14502* (cit. on p. 51).
- Fan, Angela, Mike Lewis, and Yann Dauphin (2018). “Hierarchical neural story generation”. In: *arXiv preprint arXiv:1805.04833* (cit. on p. 44).
- Fisher, James L, Stuart C Hinds, and Donald P D’Amato (1990). “A rule-based system for document image segmentation”. In: *[1990] Proceedings. 10th International Conference on Pattern Recognition*. Vol. 1. IEEE, pp. 567–572 (cit. on p. 60).
- Gage, Philip (1994). “A new algorithm for data compression”. In: *C Users Journal* 12.2, pp. 23–38 (cit. on pp. 13, 29).
- Gale, William A and Geoffrey Sampson (1995). “Good-turing frequency estimation without tears”. In: *Journal of quantitative linguistics* 2.3, pp. 217–237 (cit. on p. 16).
- Gao, Liangcai, Yilun Huang, Hervé Déjean, Jean-Luc Meunier, Qinqin Yan, Yu Fang, Florian Kleber, and Eva Lang (2019). “ICDAR 2019 competition on table detection and recognition (cTDaR)”. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, pp. 1510–1515 (cit. on p. 63).
- Gao, Liangcai, Xiaohan Yi, Zhuoren Jiang, Leipeng Hao, and Zhi Tang (2017). “ICDAR2017 competition on page object detection”. In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 1. IEEE, pp. 1417–1422 (cit. on p. 62).
- Gehman, Samuel, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith (2020). “Realtotoxicityprompts: Evaluating neural toxic degeneration in language models”. In: *arXiv preprint arXiv:2009.11462* (cit. on p. 49).
- Göbel, Max, Tamir Hassan, Ermelinda Oro, and Giorgio Orsi (2013). “ICDAR 2013 table competition”. In: *2013 12th International Conference on Document Analysis and Recognition*. IEEE, pp. 1449–1453 (cit. on p. 63).

- Graliński, Filip, Tomasz Stanisławek, Anna Wróblewska, Dawid Lipiński, Agnieszka Kaliska, Paulina Rosalska, Bartosz Topolski, and Przemysław Biecek (2020). “Kleister: A novel task for information extraction involving long documents with complex layout”. In: *arXiv preprint arXiv:2003.02356* (cit. on pp. 64, 105, 124).
- Gu, Zhangxuan, Changhua Meng, Ke Wang, Jun Lan, Weiqiang Wang, Ming Gu, and Liqing Zhang (2022). “Xylayoutlm: Towards layout-aware multimodal networks for visually-rich document understanding”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4583–4592 (cit. on p. 75).
- Guthrie, John T, Tracy Britten, and K Georgene Barker (1991). “Roles of document structure, cognitive strategy, and awareness in searching for information”. In: *Reading Research Quarterly*, pp. 300–324 (cit. on p. 69).
- Ha, Jaekyu, Robert M Haralick, and Ihsin T Phillips (1995). “Recursive XY cut using bounding boxes of connected components”. In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 2. IEEE, pp. 952–955 (cit. on p. 75).
- Hadi, Muhammad Usman, Rizwan Qureshi, Abbas Shah, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, Seyedali Mirjalili, et al. (2023). “Large Language Models: A Comprehensive Survey of its Applications, Challenges, Limitations, and Future Prospects”. In: (cit. on p. 48).
- Harley, Adam W, Alex Ufkes, and Konstantinos G Derpanis (2015). “Evaluation of deep convolutional nets for document image classification and retrieval”. In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, pp. 991–995 (cit. on pp. 61, 63, 79, 94, 95).
- Hauser, Marc D, Noam Chomsky, and W Tecumseh Fitch (2002). “The faculty of language: what is it, who has it, and how did it evolve?” In: *science* 298.5598, pp. 1569–1579 (cit. on p. 7).
- Haviv, Adi, Ori Ram, Ofir Press, Peter Izsak, and Omer Levy (2022). “Transformer language models without positional encodings still learn positional information”. In: *arXiv preprint arXiv:2203.16634* (cit. on p. 106).
- He, Kaiming, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick (2022). “Masked autoencoders are scalable vision learners”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009 (cit. on p. 78).
- He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778 (cit. on p. 33).
- He, Pengcheng, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen (2020). “Deberta: Decoding-enhanced bert with disentangled attention”. In: *arXiv preprint arXiv:2006.03654* (cit. on p. 74).

- Hermann, Karl Moritz, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom (2015). “Teaching machines to read and comprehend”. In: *Advances in neural information processing systems* 28 (cit. on p. 128).
- Hilbert, Martin and Priscila López (2011). “The world’s technological capacity to store, communicate, and compute information”. In: *science* 332.6025, pp. 60–65 (cit. on p. 1).
- Hochreiter, Sepp, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. (2001). *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies* (cit. on p. 23).
- Hochreiter, Sepp and Jürgen Schmidhuber (1997). “Long short-term memory”. In: *Neural computation* 9.8, pp. 1735–1780 (cit. on p. 23).
- Hofmann, Thomas (2001). “Unsupervised learning by probabilistic latent semantic analysis”. In: *Machine learning* 42, pp. 177–196 (cit. on p. 21).
- Hong, Teakgyu, DongHyun Kim, Mingi Ji, Wonseok Hwang, Daehyun Nam, and Sungrae Park (2020). “Bros: A pre-trained language model for understanding texts in document”. In: (cit. on pp. 70, 72, 75, 79).
- Huang, Cheng-Zhi Anna, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck (2018). “Music transformer”. In: *arXiv preprint arXiv:1809.04281* (cit. on p. 38).
- Huang, Yupan, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei (2022). “Layoutlmv3: Pre-training for document ai with unified text and image masking”. In: *Proceedings of the 30th ACM International Conference on Multimedia*, pp. 4083–4091 (cit. on pp. ix, 70, 72, 76, 77, 80).
- Huang, Zheng, Kai Chen, Jianhua He, Xiang Bai, Dimosthenis Karatzas, Shijian Lu, and CV Jawahar (2019). “Icdar2019 competition on scanned receipt ocr and information extraction”. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, pp. 1516–1520 (cit. on pp. 64, 117, 118).
- Hwang, Wonseok, Jinyeong Yim, Seunghyun Park, Sohee Yang, and Minjoon Seo (2020). “Spatial dependency parsing for semi-structured document information extraction”. In: *arXiv preprint arXiv:2005.00642* (cit. on p. 68).
- Jaume, Guillaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran (2019). “Funsd: A dataset for form understanding in noisy scanned documents”. In: *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*. Vol. 2. IEEE, pp. 1–6 (cit. on pp. 61, 63, 117, 118).
- Jiang, Jinhao, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen (2023). “Structgpt: A general framework for large language model to reason over structured data”. In: *arXiv preprint arXiv:2305.09645* (cit. on p. 49).

- Jiao, Xiaoqi, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu (2019). “Tinybert: Distilling bert for natural language understanding”. In: *arXiv preprint arXiv:1909.10351* (cit. on p. 43).
- Jones, Karen Spärck (2005). “Some points in a time”. In: *Computational Linguistics* 31.1, pp. 1–14 (cit. on p. 11).
- Joshi, Mandar, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy (2020). “Spanbert: Improving pre-training by representing and predicting spans”. In: *Transactions of the association for computational linguistics* 8, pp. 64–77 (cit. on pp. 42, 79).
- Jozefowicz, Rafal, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu (2016). “Exploring the limits of language modeling”. In: *arXiv preprint arXiv:1602.02410* (cit. on p. 15).
- Kalchbrenner, Nal, Edward Grefenstette, and Phil Blunsom (2014). “A convolutional neural network for modelling sentences”. In: *arXiv preprint arXiv:1404.2188* (cit. on p. 23).
- Kaplan, Jared, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei (2020). “Scaling laws for neural language models”. In: *arXiv preprint arXiv:2001.08361* (cit. on pp. 28, 48).
- Kardas, Marcin, Piotr Czapla, Pontus Stenetorp, Sebastian Ruder, Sebastian Riedel, Ross Taylor, and Robert Stojnic (2020). “Axccl: Automatic extraction of results from machine learning papers”. In: *arXiv preprint arXiv:2004.14356* (cit. on p. 64).
- Katti, Anoop Raveendra, Christian Reisswig, Cordula Guder, Sebastian Brarda, Steffen Bickel, Johannes Höhne, and Jean Baptiste Faddoul (2018). “Chargrid: Towards understanding 2d documents”. In: *arXiv preprint arXiv:1809.08799* (cit. on p. 67).
- Katz, Slava (1987). “Estimation of probabilities from sparse data for the language model component of a speech recognizer”. In: *IEEE transactions on acoustics, speech, and signal processing* 35.3, pp. 400–401 (cit. on p. 16).
- Kay, Anthony (July 2007). “Tesseract: An Open-Source Optical Character Recognition Engine”. In: *Linux J.* 2007.159, p. 2 (cit. on pp. 71, 95, 108).
- Ke, Guolin, Di He, and Tie-Yan Liu (2020). “Rethinking positional encoding in language pre-training”. In: *arXiv preprint arXiv:2006.15595* (cit. on p. 38).
- Kerroumi, Mohamed, Othmane Sayem, and Aymen Shabou (2021). “VisualWord-Grid: Information extraction from scanned documents using a multimodal approach”. In: *International Conference on Document Analysis and Recognition*. Springer, pp. 389–402 (cit. on pp. 67, 68).
- Khandelwal, Urvashi, He He, Peng Qi, and Dan Jurafsky (2018). “Sharp nearby, fuzzy far away: How neural language models use context”. In: *arXiv preprint arXiv:1805.04623* (cit. on p. 55).

- Kieras, David (1978). "Beyond pictures and words: Alternative information-processing models for imagery effect in verbal memory." In: *Psychological Bulletin* 85.3, p. 532 (cit. on p. 86).
- Kiesel, Johannes, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast (2019). "Semeval-2019 task 4: Hyperpartisan news detection". In: *Proceedings of the 13th International Workshop on Semantic Evaluation*, pp. 829–839 (cit. on p. 55).
- Kim, Geewook, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park (2022). "Ocr-free document understanding transformer". In: *European Conference on Computer Vision*. Springer, pp. 498–517 (cit. on pp. 70, 78, 106, 151).
- Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (cit. on p. 138).
- Kitaev, Nikita, Łukasz Kaiser, and Anselm Levskaya (2020). "Reformer: The Efficient Transformer". In: *arXiv preprint arXiv:2001.04451* (cit. on pp. 54, 93).
- Koh, Huan Yee, Jiaxin Ju, Ming Liu, and Shirui Pan (2022). "An empirical survey on long document summarization: Datasets, models, and metrics". In: *ACM computing surveys* 55.8, pp. 1–35 (cit. on p. 50).
- Kovačević, Aldin and Dino Kečo (2022). "Bidirectional LSTM networks for abstractive text summarization". In: *Advanced Technologies, Systems, and Applications VI: Proceedings of the International Symposium on Innovative and Interdisciplinary Applications of Advanced Technologies (IAT) 2021*. Springer, pp. 281–293 (cit. on p. 23).
- Kress, Gunther R, Theo Van Leeuwen, et al. (1996). *Reading images: The grammar of visual design*. Psychology Press (cit. on p. 86).
- Kumar, Jayant and David Doermann (2013). "Unsupervised classification of structurally similar document images". In: *2013 12th International Conference on Document Analysis and Recognition*. IEEE, pp. 1225–1229 (cit. on p. 63).
- Lai, Guokun, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy (2017). "Race: Large-scale reading comprehension dataset from examinations". In: *arXiv preprint arXiv:1704.04683* (cit. on p. 55).
- Lample, Guillaume and Alexis Conneau (2019). "Cross-lingual language model pretraining". In: *arXiv preprint arXiv:1901.07291* (cit. on p. 43).
- Lan, Zhenzhong, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut (2019). "Albert: A lite bert for self-supervised learning of language representations". In: *arXiv preprint arXiv:1909.11942* (cit. on p. 42).
- Laurençon, Hugo, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander M Rush, Douwe Kiela, et al. (2023). "Obelisc: An open web-scale filtered dataset of

- interleaved image-text documents". In: *arXiv preprint arXiv:2306.16527* (cit. on p. 151).
- Le, Hang, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoit Crabbé, Laurent Besacier, and Didier Schwab (2019). "Flaubert: Unsupervised language model pre-training for french". In: *arXiv preprint arXiv:1912.05372* (cit. on p. 42).
- Lebourgeois, Frank, Zbigniew Bublinski, and Hubert Emptoz (1992). "A fast and efficient method for extracting text paragraphs and graphics from unconstrained documents". In: *11th IAPR International Conference on Pattern Recognition. Vol. II. Conference B: Pattern Recognition Methodology and Systems. Vol. 1*. IEEE Computer Society, pp. 272–273 (cit. on p. 60).
- Leckner, Sara (2012). "Presentation factors affecting reading behaviour in readers of newspaper media: an eye-tracking perspective". In: *Visual Communication* 11.2, pp. 163–184 (cit. on p. 86).
- LeCun, Yann, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel (1989). "Backpropagation applied to handwritten zip code recognition". In: *Neural computation* 1.4, pp. 541–551 (cit. on p. 23).
- Lee, Chen-Yu, Chun-Liang Li, Timothy Dozat, Vincent Perot, Guolong Su, Nan Hua, Joshua Ainslie, Renshen Wang, Yasuhisa Fujii, and Tomas Pfister (2022). "Formnet: Structural encoding beyond sequential modeling in form document information extraction". In: *arXiv preprint arXiv:2203.08411* (cit. on pp. 70, 74, 75).
- Lee, Jinhyuk, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang (2020). "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: *Bioinformatics* 36.4, pp. 1234–1240 (cit. on p. 42).
- Lemarié, Julie, H el ene Eyrolle, and Jean-Marie Cellier (2008). "The segmented presentation of visually structured texts: Effects on text comprehension". In: *Computers in human behavior* 24.3, pp. 888–902 (cit. on p. 86).
- Lewis, David, Gady Agam, Shlomo Argamon, Ophir Frieder, D Grossman, and Jefferson Heard (2006). "Building a test collection for complex document information processing". In: *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 665–666 (cit. on pp. 78, 115).
- Lewis, Mike, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer (2019). "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension". In: *arXiv preprint arXiv:1910.13461* (cit. on pp. 45, 50, 58, 78, 114, 136, 149).

- Li, Chenliang, Bin Bi, Ming Yan, Wei Wang, Songfang Huang, Fei Huang, and Luo Si (2021). “Structurallm: Structural pre-training for form understanding”. In: *arXiv preprint arXiv:2105.11210* (cit. on pp. 70, 72, 78, 81).
- Li, Minghao, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li (2019). “Tablebank: A benchmark dataset for table detection and recognition”. In: *arXiv preprint arXiv:1903.01949* (cit. on p. 63).
- Li, Minghao, Yiheng Xu, Lei Cui, Shaohan Huang, Furu Wei, Zhoujun Li, and Ming Zhou (2020). “DocBank: A benchmark dataset for document layout analysis”. In: *arXiv preprint arXiv:2006.01038* (cit. on pp. 61, 62, 70, 87, 94–96).
- Li, Peizhao, Jiuxiang Gu, Jason Kuen, Vlad I Morariu, Handong Zhao, Rajiv Jain, Varun Manjunatha, and Hongfu Liu (2021). “Selfdoc: Self-supervised document representation learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5652–5660 (cit. on pp. ix, 70, 73, 76, 77, 79–81).
- Lin, Chin-Yew (2004). “Rouge: A package for automatic evaluation of summaries”. In: *Text summarization branches out*, pp. 74–81 (cit. on p. 18).
- Lin, Tsung-Yi, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie (2017). “Feature pyramid networks for object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125 (cit. on p. 77).
- Lin, Weihong, Qifang Gao, Lei Sun, Zhuoyao Zhong, Kai Hu, Qin Ren, and Qiang Huo (2021). “ViBERTgrid: a jointly trained multi-modal 2D document representation for key information extraction from documents”. In: *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part I* 16. Springer, pp. 548–563 (cit. on p. 68).
- Liu, Haotian, Chunyuan Li, Qingyang Wu, and Yong Jae Lee (2023). “Visual instruction tuning”. In: *arXiv preprint arXiv:2304.08485* (cit. on p. 151).
- Liu, Xiaojing, Feiyu Gao, Qiong Zhang, and Huasha Zhao (2019). “Graph convolution for multimodal information extraction from visually rich documents”. In: *arXiv preprint arXiv:1903.11279* (cit. on p. 68).
- Liu, Yinhan, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer (2020). “Multilingual denoising pre-training for neural machine translation”. In: *Transactions of the Association for Computational Linguistics* 8, pp. 726–742 (cit. on pp. 45, 78, 136).
- Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov (2019). “Roberta: A robustly optimized bert pretraining approach”. In: *arXiv preprint arXiv:1907.11692* (cit. on pp. 42, 78).

- Liu, Yixin, Ansong Ni, Linyong Nan, Budhaditya Deb, Chenguang Zhu, Ahmed H Awadallah, and Dragomir Radev (2022). “Leveraging locality in abstractive text summarization”. In: *arXiv preprint arXiv:2205.12476* (cit. on p. 52).
- Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo (2021). “Swin transformer: Hierarchical vision transformer using shifted windows”. In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022 (cit. on p. 78).
- Luo, Ling, Xiang Ao, Yan Song, Feiyang Pan, Min Yang, and Qing He (2019). “Reading like HER: Human reading inspired extractive summarization”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3033–3043 (cit. on p. 128).
- Luo, Siwen, Yihao Ding, Siqu Long, Josiah Poon, and Soyeon Caren Han (2022). “Doc-gcn: Heterogeneous graph convolutional networks for document layout analysis”. In: *arXiv preprint arXiv:2208.10970* (cit. on p. 67).
- Maas, Andrew, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts (2011). “Learning word vectors for sentiment analysis”. In: *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150 (cit. on p. 55).
- Manning, Christopher and Hinrich Schütze (1999). *Foundations of statistical natural language processing*. MIT press (cit. on p. 12).
- Marcus, Mitchell, Beatrice Santorini, and Mary Ann Marcinkiewicz (1993). “Building a large annotated corpus of English: The Penn Treebank”. In: (cit. on p. 47).
- Martin, Louis, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de La Clergerie, Djamé Seddah, and Benoît Sagot (2019). “CamemBERT: a tasty French language model”. In: *arXiv preprint arXiv:1911.03894* (cit. on p. 42).
- Mathew, Minesh, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar (2022). “InfographicVQA”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1697–1706 (cit. on p. 65).
- Mathew, Minesh, Dimosthenis Karatzas, and CV Jawahar (2021). “Docvqa: A dataset for vqa on document images”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2200–2209 (cit. on pp. 61, 65).
- Merity, Stephen, Caiming Xiong, James Bradbury, and Richard Socher (2016). “Pointer sentinel mixture models”. In: *arXiv preprint arXiv:1609.07843* (cit. on p. 47).
- Meyer, Bonnie JF, David M Brandt, and George J Bluth (1980). “Use of top-level structure in text: Key for reading comprehension of ninth-grade students”. In: *Reading research quarterly*, pp. 72–103 (cit. on pp. 69, 86).

- Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean (2013). "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (cit. on pp. 21, 24, 25, 27).
- Mikolov, Tomas, Martin Karafiat, Lukas Burget, Jan Cernocky, and Sanjeev Khudanpur (2010). "Recurrent neural network based language model." In: *Inter-speech*. Vol. 2. 3. Makuhari, pp. 1045–1048 (cit. on p. 23).
- Mindee (2021). *docTR: Document Text Recognition*. <https://github.com/mindee/doctr> (cit. on pp. 108, 116).
- Mohamadi, Salman, Ghulam Mujtaba, Ngan Le, Gianfranco Doretto, and Donald A Adjeroh (2023). "Chatgpt in the age of generative ai and large language models: A concise survey". In: *arXiv preprint arXiv:2307.04251* (cit. on p. 48).
- Nakayama, Hiroki (2018). *seqeval: A Python framework for sequence labeling evaluation*. Software available from <https://github.com/chakki-works/seqeval>. URL: <https://github.com/chakki-works/seqeval> (cit. on p. 121).
- Nguyen, Laura, Thomas Scialom, Benjamin Piwowarski, and Jacopo Staiano (May 2023). "LoRaLay: A Multilingual and Multimodal Dataset for Long Range and Layout-Aware Summarization". In: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Dubrovnik, Croatia: Association for Computational Linguistics, pp. 636–651. URL: <https://aclanthology.org/2023.eacl-main.46> (cit. on p. 125).
- Nguyen, Laura, Thomas Scialom, Jacopo Staiano, and Benjamin Piwowarski (Nov. 2021). "Skim-Attention: Learning to Focus via Document Layout". In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 2413–2427. URL: <https://aclanthology.org/2021.findings-emnlp.207> (cit. on p. 85).
- Nguyen, Thien Huu and Ralph Grishman (2015). "Relation extraction: Perspective from convolutional neural networks". In: *Proceedings of the 1st workshop on vector space modeling for natural language processing*, pp. 39–48 (cit. on p. 23).
- Olive, Thierry and Marie-Laure Barbier (2017). "Processing time and cognitive effort of longhand note taking when reading and summarizing a structured or linear text". In: *Written Communication* 34.2, pp. 224–246 (cit. on p. 86).
- Oliveira, Sofia Ares, Benoit Seguin, and Frederic Kaplan (2018). "dhSegment: A generic deep-learning approach for document segmentation". In: *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, pp. 7–12 (cit. on p. 67).
- Omar, Nazlia and Qasem Al-Tashi (2018). "Arabic nested noun compound extraction based on linguistic features and statistical measures". In: *GEMA Online® Journal of Language Studies* 18.2 (cit. on p. 15).
- Ouyang, Long, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. (2022). "Training language models to follow instructions with human feed-

- back". In: *Advances in Neural Information Processing Systems* 35, pp. 27730–27744 (cit. on p. 46).
- Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu (2002). "Bleu: a method for automatic evaluation of machine translation". In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318 (cit. on pp. 18, 141).
- Park, Seunghyun, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee (2019). "CORD: a consolidated receipt dataset for post-OCR parsing". In: *Workshop on Document Intelligence at NeurIPS 2019* (cit. on pp. 64, 117, 118).
- Parmar, Niki, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran (2018). "Image transformer". In: *International conference on machine learning*. PMLR, pp. 4055–4064 (cit. on p. 55).
- Pasupat, Panupong and Percy Liang (2015). "Compositional semantic parsing on semi-structured tables". In: *arXiv preprint arXiv:1508.00305* (cit. on p. 65).
- Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer (2017). "Automatic differentiation in PyTorch". In: (cit. on pp. 118, 138).
- Paulus, Romain, Caiming Xiong, and Richard Socher (2017). "A deep reinforced model for abstractive summarization". In: *arXiv preprint arXiv:1705.04304* (cit. on p. 44).
- Peng, Qiming, Yinxu Pan, Wenjin Wang, Bin Luo, Zhenyu Zhang, Zhengjie Huang, Teng Hu, Weichong Yin, Yongfeng Chen, Yin Zhang, et al. (2022). "Ernie-layout: Layout knowledge enhanced pre-training for visually-rich document understanding". In: *arXiv preprint arXiv:2210.06155* (cit. on pp. 60, 70, 72, 74, 75, 78, 79, 106, 113).
- Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543 (cit. on p. 25).
- Perot, Vincent, Kai Kang, Florian Luisier, Guolong Su, Xiaoyu Sun, Ramya Sree Boppana, Zilong Wang, Jiaqi Mu, Hao Zhang, and Nan Hua (2023). "LMDX: Language Model-based Document Information Extraction and Localization". In: *arXiv preprint arXiv:2309.10952* (cit. on p. 151).
- Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer (June 2018). "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. URL: <https://aclanthology.org/N18-1202> (cit. on p. 27).

- Petroni, Fabio, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Mail-lard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel (June 2021). “KILT: a Benchmark for Knowledge Intensive Language Tasks”. In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, pp. 2523–2544. URL: <https://aclanthology.org/2021.naacl-main.200> (cit. on p. 11).
- Pham, Hai, Guoxin Wang, Yijuan Lu, Dinei Florencio, and Cha Zhang (2022). “Understanding long documents with different position-aware attentions”. In: *arXiv preprint arXiv:2208.08201* (cit. on pp. 82, 127).
- Pham, Ngoc-Quan, German Kruszewski, and Gemma Boleda (2016). “Convolutional neural network language models”. In: *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 1153–1162 (cit. on p. 24).
- Powalski, Rafał, Łukasz Borchmann, Dawid Jurkiewicz, Tomasz Dwojak, Michał Pietruszka, and Gabriela Pałka (2021). “Going Full-TILT Boogie on Document Understanding with Text-Image-Layout Transformer”. In: *arXiv preprint arXiv:2102.09550* (cit. on pp. 70, 74, 78, 79, 106, 150).
- Pramanik, Subhojeet, Shashank Mujumdar, and Hima Patel (2020). “Towards a Multi-modal, Multi-task Learning based Pre-training Framework for Document Representation Learning”. In: *arXiv preprint arXiv:2009.14457* (cit. on pp. 70, 82, 127).
- Prasad, Devashish, Ayan Gadpal, Kshitij Kapadni, Manish Visave, and Kavita Sultantpure (2020). “CascadeTabNet: An approach for end to end table detection and structure recognition from image-based documents”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 572–573 (cit. on p. 67).
- Qin, Guanghui, Yukun Feng, and Benjamin Van Durme (2022). “The nlp task effectiveness of long-range transformers”. In: *arXiv preprint arXiv:2202.07856* (cit. on pp. 51, 56, 57).
- Radford, Alec, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. (2018). “Improving language understanding by generative pre-training”. In: (cit. on pp. 27, 28, 46, 47).
- Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. (2019). “Language models are unsupervised multitask learners”. In: *OpenAI blog* 1.8, p. 9 (cit. on pp. 47, 147).
- Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu (2020). “Exploring the limits of transfer learning with a unified text-to-text transformer”. In: *The Journal of Machine Learning Research* 21.1, pp. 5485–5551 (cit. on pp. 38, 46, 47, 73, 78, 81, 150).

- Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang (2016). “Squad: 100,000+ questions for machine comprehension of text”. In: *arXiv preprint arXiv:1606.05250* (cit. on p. 11).
- Ramesh, Aditya, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever (2021). “Zero-shot text-to-image generation”. In: *International Conference on Machine Learning*. PMLR, pp. 8821–8831 (cit. on p. 80).
- Ramshaw, Lance A and Mitchell P Marcus (1999). “Text chunking using transformation-based learning”. In: *Natural language processing using very large corpora*. Springer, pp. 157–176 (cit. on pp. 105, 120).
- Ranzato, Marc’Aurelio, Sumit Chopra, Michael Auli, and Wojciech Zaremba (2015). “Sequence level training with recurrent neural networks”. In: *arXiv preprint arXiv:1511.06732* (cit. on p. 44).
- Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun (2015). “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *Advances in neural information processing systems* 28 (cit. on pp. 66, 67, 73, 76).
- Rosenfeld, Ronald (2000). “Two decades of statistical language modeling: Where do we go from here?” In: *Proceedings of the IEEE* 88.8, pp. 1270–1278 (cit. on p. 16).
- Sage, Clément, Alex Aussem, Véronique Eglin, Haytham Elghazel, and Jérémy Espinas (2020). “End-to-end extraction of structured information from business documents with pointer-generator networks”. In: *Proceedings of the fourth workshop on structured prediction for NLP*, pp. 43–52 (cit. on p. 106).
- Sanh, Victor, Lysandre Debut, Julien Chaumond, and Thomas Wolf (2019). “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”. In: *arXiv preprint arXiv:1910.01108* (cit. on p. 43).
- Scao, Teven Le, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. (2022). “Bloom: A 176b-parameter open-access multilingual language model”. In: *arXiv preprint arXiv:2211.05100* (cit. on p. 48).
- Schwenk, Holger (2007). “Continuous space language models”. In: *Computer Speech & Language* 21.3, pp. 492–518 (cit. on p. 23).
- Scialom, Thomas, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano (2020). “MLSUM: The multilingual summarization corpus”. In: *arXiv preprint arXiv:2004.14900* (cit. on p. 128).
- Scialom, Thomas, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano (2019). “Answers unite! unsupervised metrics for reinforced summarization models”. In: *arXiv preprint arXiv:1909.01610* (cit. on p. 44).
- See, Abigail, Peter J Liu, and Christopher D Manning (2017). “Get to the point: Summarization with pointer-generator networks”. In: *arXiv preprint arXiv:1704.04368* (cit. on p. 106).

- Shaham, Uri, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, et al. (2022). “Scrolls: Standardized comparison over long language sequences”. In: *arXiv preprint arXiv:2201.03533* (cit. on p. 56).
- Sharma, Eva, Chen Li, and Lu Wang (2019). “BIGPATENT: A large-scale dataset for abstractive and coherent summarization”. In: *arXiv preprint arXiv:1906.03741* (cit. on pp. 50, 128, 134).
- Shaw, Peter, Jakob Uszkoreit, and Ashish Vaswani (2018). “Self-attention with relative position representations”. In: *arXiv preprint arXiv:1803.02155* (cit. on p. 38).
- Shazeer, Noam, Ryan Doherty, Colin Evans, and Chris Waterson (2016). “Swivel: Improving embeddings by noticing what’s missing”. In: *arXiv preprint arXiv:1602.02215* (cit. on pp. 21, 25).
- Shazeer, Noam and Mitchell Stern (2018). “Adafactor: Adaptive learning rates with sublinear memory cost”. In: *International Conference on Machine Learning*. PMLR, pp. 4596–4604 (cit. on p. 138).
- Shen, Zejiang, Kyle Lo, Lucy Lu Wang, Bailey Kuehl, Daniel S Weld, and Doug Downey (2022). “VILA: Improving structured content extraction from scientific PDFs using visual layout groups”. In: *Transactions of the Association for Computational Linguistics* 10, pp. 376–392 (cit. on pp. 70, 73).
- Sheng, Emily, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng (2021). “Societal biases in language generation: Progress and challenges”. In: *arXiv preprint arXiv:2105.04054* (cit. on p. 49).
- Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts (2013). “Recursive deep models for semantic compositionality over a sentiment treebank”. In: *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642 (cit. on p. 55).
- Soto, Carlos and Shinjae Yoo (2019). “Visual detection with context for document layout analysis”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3464–3470 (cit. on p. 67).
- Sun, Simeng, Kalpesh Krishna, Andrew Mattarella-Micke, and Mohit Iyyer (2021). “Do long-range language models actually use long-range context?” In: *arXiv preprint arXiv:2109.09115* (cit. on pp. 55, 57).
- Sutskever, Ilya, Oriol Vinyals, and Quoc V Le (2014). “Sequence to sequence learning with neural networks”. In: *Advances in neural information processing systems* 27 (cit. on p. 29).
- Tanaka, Ryota, Kyosuke Nishida, and Sen Yoshida (2021). “Visualmrc: Machine reading comprehension on document images”. In: *Proceedings of the AAAI*

- Conference on Artificial Intelligence*. Vol. 35. 15, pp. 13878–13888 (cit. on pp. 65, 70, 78).
- Tang, Yuqing, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan (2020). “Multilingual translation with extensible multilingual pretraining and finetuning”. In: *arXiv preprint arXiv:2008.00401* (cit. on p. 136).
- Tang, Zineng, Ziyi Yang, Guoxin Wang, Yuwei Fang, Yang Liu, Chenguang Zhu, Michael Zeng, Cha Zhang, and Mohit Bansal (2023). “Unifying vision, text, and layout for universal document processing”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19254–19264 (cit. on pp. 70, 73, 77, 78, 80, 150).
- Tay, Yi, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler (2020a). “Long Range Arena: A Benchmark for Efficient Transformers”. In: *arXiv preprint arXiv:2011.04006* (cit. on p. 55).
- Tay, Yi, Mostafa Dehghani, Dara Bahri, and Donald Metzler (2020b). “Efficient Transformers: A Survey”. In: *arXiv preprint arXiv:2009.06732* (cit. on p. 51).
- Thede, Scott M and Mary Harper (1999). “A second-order hidden Markov model for part-of-speech tagging”. In: *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, pp. 175–182 (cit. on p. 16).
- Thirunavukarasu, Arun James, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting (2023). “Large language models in medicine”. In: *Nature medicine*, pp. 1–11 (cit. on pp. 12, 48).
- Toprak, Elif and Gamze Almacioğlu (2009). “Three reading phases and their applications in the teaching of english as a foreign language in reading classes with young learners”. In: *Journal of language and Linguistic Studies* 5.1 (cit. on p. 128).
- Touvron, Hugo, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. (2023). “Llama 2: Open foundation and fine-tuned chat models”. In: *arXiv preprint arXiv:2307.09288* (cit. on pp. 46, 48, 147).
- Townsend, Benjamin, Eamon Ito-Fisher, Lily Zhang, and Madison May (2021). “Doc2dict: Information extraction as text generation”. In: *arXiv preprint arXiv:2105.07510* (cit. on p. 106).
- Turing, A. M. (1950). “Computing Machinery and Intelligence”. English. In: *Mind*. New Series 59.236, pp. 433–460. URL: <http://www.jstor.org/stable/2251299> (cit. on p. 7).
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). “Attention is all you need”. In: *Advances in neural information processing systems* 30 (cit. on pp. 3, 28, 29, 32, 37, 69, 106, 127, 147).

- Villalobos, Pablo, Jaime Sevilla, Lennart Heim, Tamay Besiroglu, Marius Hobbhahn, and Anson Ho (2022). "Will we run out of data? An analysis of the limits of scaling datasets in Machine Learning". In: *arXiv preprint arXiv:2211.04325* (cit. on p. 1).
- Von Platen, Patrick (2020). "How to generate text: using different decoding methods for language generation with Transformers". "Website". <https://huggingface.co/blog/how-to-generate#how-to-generate-text-using-different-decoding-methods-for-language-generation-with-transformers> (cit. on p. 44).
- Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman (2018). "GLUE: A multi-task benchmark and analysis platform for natural language understanding". In: *arXiv preprint arXiv:1804.07461* (cit. on pp. 11, 55).
- Wang, Dongsheng, Natraj Raman, Mathieu Sibue, Zhiqiang Ma, Petr Babkin, Simerjot Kaur, Yulong Pei, Armineh Nourbakhsh, and Xiaomo Liu (2023). "DocLLM: A layout-aware generative language model for multimodal document understanding". In: *arXiv preprint arXiv:2401.00908* (cit. on p. 151).
- Wang, Guoxin, Yijuan Lu, Lei Cui, Tengchao Lv, Dinei Florencio, and Cha Zhang (2022). "A Simple yet Effective Learnable Positional Encoding Method for Improving Document Transformer Model". In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2022*, pp. 453–463 (cit. on p. 75).
- Wang, Hanchen, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. (2023). "Scientific discovery in the age of artificial intelligence". In: *Nature* 620.7972, pp. 47–60 (cit. on pp. 12, 48).
- Wang, Jiapeng, Lianwen Jin, and Kai Ding (2022). "Lilt: A simple yet effective language-independent layout transformer for structured document understanding". In: *arXiv preprint arXiv:2202.13669* (cit. on pp. 70, 74).
- Wang, Jiapeng, Chongyu Liu, Lianwen Jin, Guozhi Tang, Jiaxin Zhang, Shuaitao Zhang, Qianying Wang, Yaqiang Wu, and Mingxiang Cai (2021). "Towards robust visual information extraction in real world: New dataset and novel solution". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 4, pp. 2738–2745 (cit. on p. 64).
- Wang, Junjie, Yuchao Huang, Chunyang Chen, Zhe Liu, Song Wang, and Qing Wang (2023). "Software testing with large language model: Survey, landscape, and vision". In: *arXiv preprint arXiv:2307.07221* (cit. on p. 48).
- Wang, Sinong, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma (2020). "Linformer: Self-Attention with Linear Complexity". In: *arXiv preprint arXiv:2006.04768* (cit. on p. 54).

- Wang, Zhiguo, Wael Hamza, and Radu Florian (2017). “Bilateral multi-perspective matching for natural language sentences”. In: *arXiv preprint arXiv:1702.03814* (cit. on p. 55).
- Wang, Zilong, Yiheng Xu, Lei Cui, Jingbo Shang, and Furu Wei (2021). *LayoutReader: Pre-training of Text and Layout for Reading Order Detection*. arXiv: 2108.11591 [cs.CL] (cit. on pp. 75, 108, 116).
- Wang, Zilong, Yichao Zhou, Wei Wei, Chen-Yu Lee, and Sandeep Tata (2023). “Vrdu: A benchmark for visually-rich document understanding”. In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5184–5193 (cit. on p. 64).
- Wehrmann, Joonatas, Willian Becker, Henry EL Cagnini, and Rodrigo C Barros (2017). “A character-based convolutional neural network for language-agnostic Twitter sentiment analysis”. In: *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 2384–2391 (cit. on p. 13).
- Wei, Hao, Micheal Baechler, Fouad Slimane, and Rolf Ingold (2013). “Evaluation of SVM, MLP and GMM classifiers for layout analysis of historical documents”. In: *2013 12th International Conference on Document Analysis and Recognition*. IEEE, pp. 1220–1224 (cit. on p. 60).
- Wei, Jason, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le (2021). “Finetuned language models are zero-shot learners”. In: *arXiv preprint arXiv:2109.01652* (cit. on p. 151).
- Welbl, Johannes, Pontus Stenetorp, and Sebastian Riedel (2018). “Constructing datasets for multi-hop reading comprehension across documents”. In: *Transactions of the Association for Computational Linguistics* 6, pp. 287–302 (cit. on p. 55).
- Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. (2019). “Huggingface’s transformers: State-of-the-art natural language processing”. In: *arXiv preprint arXiv:1910.03771* (cit. on pp. 99, 118, 138).
- Wong, Kwan Y., Richard G. Casey, and Friedrich M. Wahl (1982). “Document analysis system”. In: *IBM journal of research and development* 26.6, pp. 647–656 (cit. on p. 60).
- Wright, Patricia (1999). “The psychology of layout: Consequences of the visual structure of documents”. In: *American Association for Artificial Intelligence Technical Report FS-99-04*, pp. 1–9 (cit. on pp. 69, 86).
- Wu, Te-Lin, Cheng Li, Mingyang Zhang, Tao Chen, Spurthi Amba Hombaiah, and Michael Bendersky (2021). “Lampret: Layout-aware multimodal pretraining for document understanding”. In: *arXiv preprint arXiv:2104.08405* (cit. on pp. 70, 73, 77, 79).

- Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. (2016). “Google’s neural machine translation system: Bridging the gap between human and machine translation”. In: *arXiv preprint arXiv:1609.08144* (cit. on pp. 14, 29).
- Xie, Saining, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He (2017). “Aggregated residual transformations for deep neural networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500 (cit. on p. 77).
- Xiong, Lee, Chuan Hu, Chenyan Xiong, Daniel Campos, and Arnold Overwijk (2019). “Open domain web keyphrase extraction beyond language modeling”. In: *arXiv preprint arXiv:1911.02671* (cit. on p. 55).
- Xu, Frank F, Uri Alon, Graham Neubig, and Vincent Josua Hellendoorn (2022). “A systematic evaluation of large language models of code”. In: *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, pp. 1–10 (cit. on pp. 12, 48).
- Xu, Yang, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. (2020). “LayoutLMv2: Multimodal Pre-training for Visually-Rich Document Understanding”. In: *arXiv preprint arXiv:2012.14740* (cit. on pp. 70, 73, 76, 78, 79, 106).
- Xu, Yiheng, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou (2020). “Layoutlm: Pre-training of text and layout for document image understanding”. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1192–1200 (cit. on pp. ix, 2, 60, 70, 71, 75, 76, 78, 79, 89, 96, 97, 100, 106, 123, 136, 147).
- Xu, Yiheng, Tengchao Lv, Lei Cui, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, and Furu Wei (2021). “Layoutxlm: Multimodal pre-training for multilingual visually-rich document understanding”. In: *arXiv preprint arXiv:2104.08836* (cit. on p. 81).
- Xu, Yiheng, Tengchao Lv, Lei Cui, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, and Furu Wei (May 2022). “XFUND: A Benchmark Dataset for Multilingual Visually Rich Form Understanding”. In: *Findings of the Association for Computational Linguistics: ACL 2022*. Dublin, Ireland: Association for Computational Linguistics, pp. 3214–3224. URL: <https://aclanthology.org/2022.findings-acl.253> (cit. on pp. 64, 70).
- Yang, Xiao, Ersin Yumer, Paul Asente, Mike Kraley, Daniel Kifer, and C Lee Giles (2017). “Learning to extract semantic structure from documents using multimodal fully convolutional neural networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5315–5324 (cit. on p. 66).
- Yang, Zhilin, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le (2019). “Xlnet: Generalized autoregressive pretraining for

- language understanding". In: *Advances in neural information processing systems* 32 (cit. on p. 53).
- Yang, Zhilin, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning (2018). "HotpotQA: A dataset for diverse, explainable multi-hop question answering". In: *arXiv preprint arXiv:1809.09600* (cit. on p. 55).
- Yao, Yunzhi, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang (2023). "Editing Large Language Models: Problems, Methods, and Opportunities". In: *arXiv preprint arXiv:2305.13172* (cit. on p. 49).
- Ye, Junjie, Xuantang Chen, Nuo Xu, Can Zu, Zekai Shao, Shichun Liu, Yuhan Cui, Zeyang Zhou, Chao Gong, Yang Shen, et al. (2023). "A comprehensive capability analysis of gpt-3 and gpt-3.5 series models". In: *arXiv preprint arXiv:2303.10420* (cit. on p. 49).
- Ye, Qinghao, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. (2023). "mplug-owl: Modularization empowers large language models with multimodality". In: *arXiv preprint arXiv:2304.14178* (cit. on p. 151).
- Yu, Wenwen, Ning Lu, Xianbiao Qi, Ping Gong, and Rong Xiao (2021). "PICK: processing key information extraction from documents using improved graph learning-convolutional networks". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, pp. 4363–4370 (cit. on p. 68).
- Zaheer, Manzil, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. (2020). "Big bird: Transformers for longer sequences". In: *Advances in Neural Information Processing Systems* 33 (cit. on pp. 51, 52, 58, 78, 137–139).
- Zhang, Chong, Ya Guo, Yi Tu, Huan Chen, Jinyang Tang, Huijia Zhu, Qi Zhang, and Tao Gui (2023). "Reading Order Matters: Information Extraction from Visually-rich Documents by Token Path Prediction". In: *arXiv preprint arXiv:2310.11016* (cit. on p. 73).
- Zhang, Jingqing, Yao Zhao, Mohammad Saleh, and Peter Liu (2020). "Pegasus: Pre-training with extracted gap-sentences for abstractive summarization". In: *International Conference on Machine Learning*. PMLR, pp. 11328–11339 (cit. on pp. 45, 136–138).
- Zhang, Ming, Chengzhang Li, Meilin Wan, Xuejun Zhang, and Qingwei Zhao (2024). "ROUGE-SEM: Better evaluation of summarization using ROUGE combined with semantics". In: *Expert Systems with Applications* 237, p. 121364 (cit. on p. 141).
- Zhao, Wayne Xin, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. (2023). "A

- survey of large language models”. In: *arXiv preprint arXiv:2303.18223* (cit. on p. 48).
- Zhong, Xu, Elaheh ShafieiBavani, and Antonio Jimeno Yepes (2020). “Image-based table recognition: data, model, and evaluation”. In: *European conference on computer vision*. Springer, pp. 564–580 (cit. on p. 63).
- Zhong, Xu, Jianbin Tang, and Antonio Jimeno Yepes (2019). “Publaynet: largest dataset ever for document layout analysis”. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, pp. 1015–1022 (cit. on pp. 62, 94, 95).
- Zhu, Yukun, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler (2015). “Aligning books and movies: Towards story-like visual explanations by watching movies and reading books”. In: *Proceedings of the IEEE international conference on computer vision*, pp. 19–27 (cit. on p. 40).