Incorporating expert knowledge in deep neural networks for domain adaptation in natural language processing

Guilhem Xavier Piat

université PARIS-SACLAY

Incorporating expert knowledge in deep neural networks for domain adaptation in natural language processing

Intégration de connaissances expertes dans des modèles neuronaux profonds pour l'adaptation au domaine dans le traitement automatique de la langue

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°580 : sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat : Informatique
Graduate School : Informatique et Sciences du Numérique. Référent : Faculté des sciences d'Orsay

Thèse préparée dans l'unité de recherche **Institut LIST (Université Paris-Saclay, CEA)**, sous la direction d'**Alexandre ALLAUZEN**, Professeur, et le co-encadrement de **Nasredine SEMMAR**, Chercheur, et de **Julien TOURILLE**, Chercheur.

Thèse soutenue à Paris-Saclay, le 11 Décembre 2023, par

Guilhem Xavier PIAT

Composition du jury

Membres du jury avec voix délibérative

Aurélie NÉVÉOL Directrice de Recherche CNRS, Université Paris-Saclay	Présidente
Reinhard RAPP Professeur, Magdeburg-Stendal University of Applied Sciences / Athena R.C.	Rapporteur & Examinateur
Xavier TANNIER Professeur, Sorbonne Université	Rapporteur & Examinateur
Élise BONZON Maîtresse de Conférences, Université Paris Cité	Examinatrice

THESE DE DOCTORAT
NNT : 2023UPASG087

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Title: Incorporating expert knowledge in deep neural networks for domain adaptation in natural language processing

Keywords: Natural language processing, Expert knowledge, Domain adaptation, Neural networks, Deep learning, Transformers

Abstract: Current state-of-the-art Language Models (LMs) are able to converse, summarize, translate, solve novel problems, reason, and use abstract concepts at a near-human level. However, to achieve such abilities, and in particular to acquire "common sense" and domain-specific knowledge, they require vast amounts of text, which are not available in all languages or domains. Additionally, their computational requirements are out of reach for most organizations, limiting their potential for specificity and their applicability in the context of sensitive data.

Knowledge Graphs (KGs) are sources of structured knowledge which associate linguistic concepts through semantic relations. These graphs are sources of high quality knowledge which pre-exist in a variety of otherwise low-resource domains, and are denser in information than typical text. By allowing LMs to leverage these information structures, the burden of memorizing facts can be removed from LMs, potentially reducing the amount of text and computation required to train them and allowing us to update their knowledge with little to no additional training by updating the KGs, therefore broadening their scope of applicability and making them more democratizable.

Various approaches have succeeded in improving Transformer-based LMs using KGs. However, most of them unrealistically assume the problem of Entity Linking (EL), *i.e.* determining which KG concepts are present in the text, is solved upstream. This thesis covers the limitations of handling EL as an upstream task. It goes on to examine the possibility of learning EL jointly with language modeling, and finds that while this is a viable strategy, it does little to decrease the LM's reliance on in-domain text. Lastly, this thesis covers the strategy of using KGs to generate text in order to leverage LMs' linguistic abilities and finds that even naïve implementations of this approach can result in measurable improvements on in-domain language processing.

Titre: Intégration de connaissances expertes dans des modèles neuronaux profonds pour l'adaptation au domaine dans le traitement automatique de la langue

Mots clés: Traitement automatique de la langue, Connaissances expertes, Adaptation au domaine, Réseaux de neurones, Apprentissage profond, Transformers

Résumé: Les Modèles de Langage (LMs) de pointe sont capables de converser, résumer, traduire, résoudre des problèmes inédits, raisonner, et manipuler des concepts abstraits à un niveau quasi-humain. Cependant, pour acquérir ces capacités, et en particulier pour acquérir une forme de "bon sens" ou des connaissances spécifiques à un domaine, ils requièrent de vastes quantités de texte, qui ne sont pas disponibles pour toutes les langues ou tous les domaines. De surcroît, leurs besoins en puissance de calcul ne sont atteignables que par quelques organisations, limitant leur spécificité ainsi que leur applicabilité aux données sensibles.

Les Graphes de Connaissances (GCs) sont des sources de connaissances structurées qui associent des concepts linguistiques entre eux par le biais de relations sémantiques. Ces graphes sont des sources de connaissances de haute qualité, préexistantes dans une variété de domaines même peu dotés en ressources, et plus denses en informations que du texte. En permettant aux LMs d'exploiter ces structures d'information, ils sont délestés de la responsabilité de mémoriser les informations factuelles, réduisant la quantité de ressources textuelles et calculatoires nécessaires à leur entraînement, et nous permettant de mettre à jour leurs connaissances à moindre coût, élargissant leur cadre d'application et augmentant leur potentiel de démocratisation.

Diverses approches pour l'amélioration de LMs par intégration de GCs ont démontré leur efficacité. Elles reposent cependant sur la supposition rarement vérifiée que le problème de Désambiguïsation d'Entités Nommées (DEN) est résolu en amont. Cette thèse couvre les limitations de cette approche, puis explore l'apprentissage simultané de modélisation de langue et de DEN. Cette démarche s'avère viable mais échoue à réduire considérablement la dépendance du LM sur le texte issu du domaine. Enfin, cette thèse aborde la stratégie de générer du texte à partir de GCs de manière à mieux exploiter les capacités linguistiques des LMs. Il en ressort que même une implémentation naïve de cette approche peut se solder par de considérables progrès en modélisation de langue dans des domaines de spécialité.

"Everything is burning, Emma, because we told it to set fire to everything. This is the workings of the obedient servant. [...] This is the good product."
Said Polat (2023)

To a bright future for humanity.

Acknowledgements

This thesis is the fruit borne by the tree grown from four years of labor, over the course of which I have invested more of myself than I knew I had to offer. Despite the caffeine addiction and sleep lost, despite the hardships and repetitive strain injuries sustained, despite the paperwork and relationships neglected, I look back on this section of my life with few regrets, for life can only thrive under stress. However, much like a tree only stands strong against the storm thanks to its roots, I have my own roots to thank for supporting me throughout this shared skybound journey.

I would first like to thank my parents, the brightest minds I've ever had the chance to probe, and whose legacy I can only hope to live up to one day. To my father, in whose footsteps I seem to have followed despite my best efforts, I am grateful for introducing me to computer science and machine learning, as well as for his support in these shared interests. To my mother, I am grateful for instilling in me the value of hard work, and for teaching me to write, particularly in our elegant mother tongue. To both, I am especially grateful for their continuous moral and logistical support, without which I could never have come this far.

I would also like to thank my friends, who have stood by me despite my occasional lack of care in maintaining our bonds. In particular, I would like to thank my dear friend and sister in arms, Wafa Aissa, without whom I would not have had the opportunity to pursue this doctorate.

I am grateful to my colleagues from the CEA for the positive and laid back work environment they fostered, the interest they took in my work, and the valuable insights they gave.

Lastly, I would like to thank my supervisors and doctoral committee. In particular, I would like to extend my utmost gratitude for the continuous guidance and encouragement of my main supervisor and mentor, Nasredine Semmar. To Julien Tourille, I am grateful for consistently taking time out of his busy schedule to lend me his technical expertise in all things biomedical and knowledge-graph-related. I take this opportunity to acknowledge Hassane Essafi who, unsolicited, took interest in my work to the point of insisting on taking part in every meeting, and helped come up with many new ideas and directions for research. I would like to thank Profs. Rapp and Tannier for agreeing to review this dissertation. Finally, I would like to extend my thanks to my director, Alexandre Allauzen, for his supervision, often in the form of characteristically specific, pertinent and insightful suggestions, as well as for his aid in having me selected as a part-time teaching assistant at Dauphine University, thereby helping me discover my passion for teaching.

Though these paltry acknowledgements do not begin to do justice to all the support I have received from friends, family, colleagues, peers, and even pets, I will have to cut them short lest I become a primary driver of deforestation by use of paper. *With gratitude to all those who have been my roots so far.*

Contents

List of Figures

List of Tables

1 - Introduction

1.1 . Context & Motivation

In the field of Artificial Intelligence (AI), the most ambitious goal may be to create an agent capable of performing many or all human cognitive tasks at a human or super-human level. This is Artificial General Intelligence (AGI). There are multiple conceptions for how an AGI might be structured, but Natural Language Processing (NLP) holds a privileged role in most of them as the most obvious tool for humans to give instructions to a flexible system, and perhaps the most universal human cognitive task.

The relationship between AGI and language modeling dates back to the inception of the field, with the proposal of *The Imitation Game* (Turing, 1950), also commonly known as *Turing test* which posits that the question "*Can digital computers mimic human written language well enough to be indistinguishable from true human writing in a conversational setting?*" (A) functionally entails the question "*Can machines think?*" (B). That is, if the answer to A is *yes*, then the answer to B must also be *yes* provided "thinking" is not specifically defined in a way that precludes machines from having this ability. This relies on the functionalist argument that the inner workings of an object should have no bearing on the way it is defined, only its influence on its environment should. One interpretation follows from the solipsistic *problem of other minds*: only one's mind is known to be true for certain; we have no reliable method for rejecting the possibility that the world may be an illusion and that, rather than equals, other minds could be confabulations of our own or automatons mimicking sapience. Consequently, there is no objective basis to discriminate between such automatons and what might be true minds other than one's own, as such a thing cannot be proven to exist.

Furthermore, the idea that fooling a human into thinking they are exchanging with another human requires a form of higher-order reasoning that may be described as "thinking" is not without merit. As language is the most versatile and universal tool we use to communicate ideas, it appears sensible that proficiency in the communication medium would entail proficiency in reasoning about the concepts being handled. In reality, the lack of specifics in terms of allotted time for the test have led to many unambiguously unthinking language models to arguably pass the test, such as Cleverbot (Carpenter, 2008; Aron, 2011). Consequently, it seems neither consciousness nor the ability to handle a wide range of cognitive tasks are necessary to fool a human into thinking they are conversing with another human in a practical, time-constrained setting.

Despite this failing of the test, the link between mastery of reasoning about abstract concepts, ability to carry out instructions and mastery of natural language remain relevant. As text (often in the form of natural language, but also program source code, mathematical equations, *etc.*) is the most common representation of inputs and outputs for cognitive tasks and descriptions of non-cognitive processes, it has been my belief since the beginning of my graduate studies that it is critical for an AGI to master this modality, and if no new fundamental breakthroughs are necessary, the first AGI will be largely based on language modeling. This belief has driven and shaped my work thus far.

While this belief may seem to stem from an absurd leap in logic, given a large enough model, the best policy to predict consequences from the description of a process is to implement an internal simulation of the process (in humans, this "simulation" maps to several cognitive processes such as *imagination* and *reasoning*). For instance, the best policy for learning to predict the sum of two numbers is to learn addition. Given examples of physics experiments, the best policy to learn to predict the results is to derive the laws of physics and simulate the experiments. Given the formulation of a cognitive task, the best policy to predict the answer is to simulate the cognitive process. Hence, the best policy for predicting arbitrary text completions, *i.e.* modeling language, is to create internal representations of the entities involved and their interactions. As universal approximators (Hornik *et al.*, 1989), it is not unexpected that given enough parameters, data, and time training, the ability to perform this at a human or super-human level could emerge in neural models. It has been proposed however, that language may be insufficient, as true understanding must be grounded in the world, and therefore AGI must be multimodal (Huang *et al.*, 2023) and/or embodied (Kremelberg, 2019).

In the era of neural networks, especially since the advent of the Transformer architecture, the main approach to progress on language modeling has been to increase the amount of parameters in the model, of training data, and of computational resources used. Projects such as ChatGPT (Schulman *et al.*, 2022) have demonstrated that this approach can achieve outstanding results, particularly for language understanding and generating grammatically correct sentences. However, it does not tend to model knowledge well or efficiently, typically having either low certainty on common-sense knowledge, and/or high certainty on untrue facts. This is likely one of the causes of "*hallucinations*" (Ji *et al.*, 2023), a phenomenon affecting large transformer-based conversational language models which leads them to confidently disseminate misinformation. An example of this behavior is provided in figure 1.1, in which ChatGPT provides a correct article title which is relevant to the query, but confidently attributes it to incorrect authors.

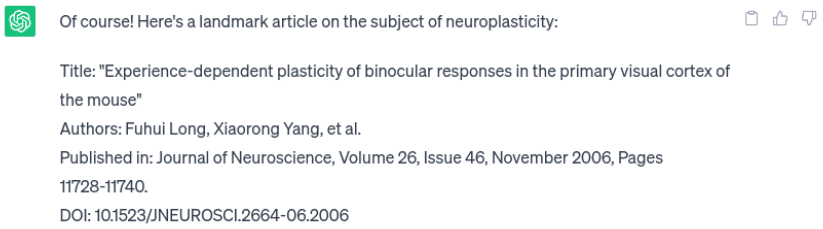


Figure 1.1: Example of "hallucination": article reference provided by ChatGPT. The article title refers to a real article which is relevant (Gordon and Stryker, 1996), but the metadata (authors, date, DOI) are incorrect.

Another significant weakness of this method is its inefficiency in terms of data and computation requirements, particularly at scale. A good basis for scaling comparison is OpenAI's series of GPT models, as they share a basic architecture and training policy, and differ mostly in the number of parameters and training data:

- GPT-1 (Radford *et al.*, 2018) has approximately 117 million parameters, was trained on 4.5 GB of text (approximately 7000 books, more text than most people read in their entire lives), and required 1.7×10^{19} FLoating-point OPerations, or 17 exaFLOP[1] to train (Epoch, 2022; Sevilla *et al.*, 2022). It generated mostly grammatically correct text, but often made mistakes and had trouble remaining consistent for more than a few sentences.
- GPT-2 (Radford *et al.*, 2019) is approximately ten times larger and trained on ten times more data for an approximately hundred times larger training budget (1.5×10^{21} FLOP), and succeeds in reliably generating a few paragraphs of grammatically correct, consistent text in a variety of contexts. Most significantly, GPT-2 demonstrated the ability to handle a variety of linguistic tasks such as summarization, creative writing, and question answering with carefully constructed prompts which typically include a few examples of the task.
- GPT-3 (Brown *et al.*, 2020) is over a hundred times larger than GPT-2 with 175 billion parameters requiring 800GB of storage, and trained on nearly twenty times more data (950 GB), and was approximately 200 times more expensive to train (3.1×10^{23} FLOP). GPT-3 can handle significantly larger amounts of text than GPT-2, and was the first model to be able to follow natural language instructions (Ouyang *et al.*, 2022) (*e.g.* "*Extract all place names from the article below:*" followed by a news article) and the first general language model to write computer programs from a list of requested features.

[1]FLOP, FLOPs, or floating-point operations, are a measure of amount of computation for a task, which is independent of time and hardware. It is not to be confused with FLOPS, FLOP/s, or floating-point operations per second, which are a measure of computation speed of a piece of computer hardware.

- Little is known about the exact technical details of GPT-4 (OpenAI, 2023), but it is known to have at least approximately a trillion parameters, it is trained on a portion of the 19.5PB Common Crawl (Elbaz *et al.*, 2012) dataset, and required approximately 2.1×10^{25} FLOP to train according to estimations by Epoch. Given the findings of Kaplan *et al.* (2020) on optimal relative scaling of LM size versus corpus size, we can expect that the corpus size should exceed 3.45 TB. Based on the computation time ratio compared to GPT-3, the corpus size could exceed 11.26 TB. Its most notable improvements over GPT-3 seem to be its general reasoning skills and its world model, as it is able to make more accurate predictions relating to real-world objects in hypothetical situations.

In summary, this mostly quantitative change in size and training data led to a major qualitative change in potential applications and usefulness. These abilities make GPT-3 and its successors unprecedented tools with the potential to assist cognitive tasks and significantly increase productivity in many white-collar professions. However, this naïve approach is highly inefficient in terms of data and computational requirements. This inefficiency has three major drawbacks, detailed below.

Computing at the billion-parameter scale and above is expensive. The cost of training a model the size of GPT-3 is measured in millions of dollars[2], and running it on a server for collaborative use would cost hundreds of thousands per year[3]. This is an expense that most organizations cannot afford. Yet, contracting a large technology corporation such as Alphabet or OpenAI to license their language models is often not an admissible option due to privacy concerns. AI corporations have a vested interest in keeping logs of user interactions in order to improve their products, have a poor track record regarding respect of privacy, and wield disproportionate economic (and therefore litigation) power. Potential client corporations dealing with sensitive data such as personal medical data, military strategic information, defendants' personal information in court proceedings and confidential intellectual property therefore cannot fully trust AI corporations to respect the confidentiality of the information. In addition to the privacy concerns, client corporations may also be interested in adapting and fine-tuning language models to their needs with internal additional data and tasks. This use case, in addition to the prohibitive cost, would typically require very large amounts of data and many man-hours to complete due to the low data-efficiency of large language models, adding yet another barrier to adoption.

Large language models may be reaching a scaling limit. As demonstrated by Kaplan *et al.*, language model performance increases (loss decreases) logarithmically with respect to the log of the number of FLOPs, the size of the training

[2] https://lambdalabs.com/blog/demystifying-gpt-3

[3] https://neoteric.eu/blog/how-much-does-it-cost-to-use-gpt-models-gpt-3-pricing-explained/

corpus, and number of model parameters. Assuming this relationship holds indefinitely and error converges asymptotically to 0, a practical scaling limit nonetheless lies within the available computational power and corpora which will likely fail to meet the exponential demand. While the idea of approaching these upper bounds may seem absurd, as of the writing of this thesis in 2023 and since 2020, a multifactorial global silicon shortage has been causing computer hardware prices to increase dramatically. It is unclear if and when this issue will be resolved, potentially causing stagnation in available computing power, even for AI-focused corporations. Furthermore, the Common Crawl dataset, the largest text dataset available which preserves a large section of the World-Wide Web, may already be utilized nearly to its full useful extent. While it is nearly 20PB large, versions with markup and low quality content filtered out are much smaller, with one of the largest being the 9.7TB multilingual Colossal Clean Crawled Corpus (Raffel *et al.*, 2020; Dodge *et al.*, 2021)[4]. This lines up with the amount of text apparently used to train GPT-4. Substantially increasing the amount of training data will likely significantly reduce its quality, or involve acquiring more data through expensive means such as licensing costly content (*e.g.* Google Books, scientific articles) and digitizing books.

Lastly, further increasing performance on the current language modeling tasks with current architectures may actually be counterproductive. As outlined by Zhuang and Hadfield-Menell (2020), when utility functions are misspecified with respect to our true goals, as the performance of AI systems increase on their training task, their performance on real-world tasks increases, plateaus, and subsequently decreases as the model learns to exploit the specificities in their proxy utility function. In conversational assistants for instance, providing false but plausible information in response to a difficult request, thereby tricking the user into believing the assistant is helpful, typically has higher utility than not attempting to give useful information, which creates a perverse incentive to deceive. As demonstrated by Perez *et al.* (2022a), this type of misalignment is already widespread in conversational language models, and as their capabilities increase, these types of failures may render their use actively harmful in subtle ways that may be difficult to detect. It is currently difficult to determine the impact of language model misalignment, and therefore difficult to determine how much further they can be scaled up before they become actively harmful.

Concerns over the environmental impact of the AI industry have also been raised due to energy use for computation. Although lack of traceability makes estimating the global carbon impact of neural network training difficult, we know that large models individually cause substantial carbon emissions (Heikkilä, 2022). For instance, the training of GPT-3 alone has consumed the equivalent of 500 metric tons of CO_2 in terms of power. This is only approximately fifteen billionths of the global emissions, but scaling power consumption by the amount of FLOPs

[4] https://www.tensorflow.org/datasets/catalog/c4
https://github.com/allenai/allennlp/discussions/5056

in training, we can estimate the training of GPT4 to have caused 34000 tons of emissions, or a millionth of the world's consumption, without taking into account the cost of running the models as part of the ChatGPT service. As more corporations pursue this type of AI research and continue to scale models up, and as more organizations train similar models, large language models may become a substantial driver of carbon emissions.

In summary, while language models can be made substantially more powerful by scaling up their size and training corpus, soon, we will likely have to turn to other improvement methods to drive progress in the field. Furthermore, large language models are not realistically democretizable due to the monetary and environmental costs of training and running them; they are inefficient learners that have difficulty assimilating facts; and they are prone to having incorrect beliefs and/or deceiving humans in order to avoid missing out on rewards when discussing topics which they know little about. In essence, we would like address these limitations by increasing language models' crystallized intelligence at low computational cost. We decide to take inspiration from the way we deal with this issue in humans: knowledge augmentation using external resources (see Fig. 1.2). We therefore attempt to leverage external, pre-existing sources of knowledge, by integrating them into Transformer-based language models. In this dissertation, I will discuss the various approaches I have undertaken in the pursuit of this objective, as well as my observations and conclusions, and we will consider directions for future research on this and related topics.

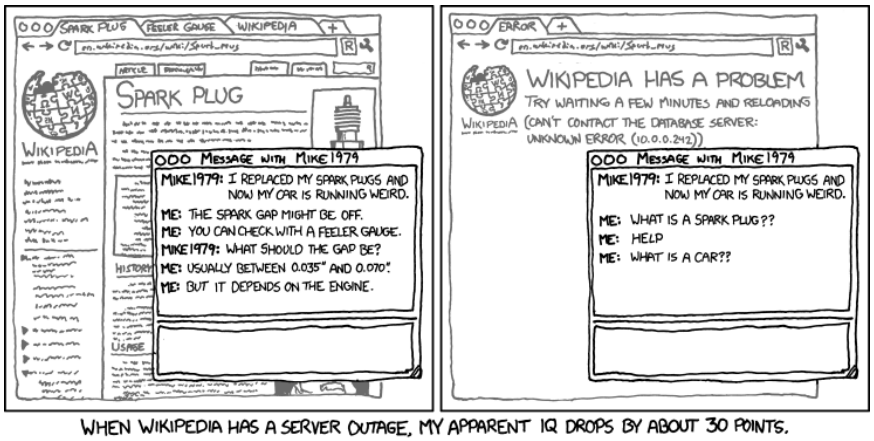


Figure 1.2: Illustration of the use of an external resource for knowledge augmentation in humans. Comic by Randall Munroe, *"Extended Mind"*. xkcd.com #903.

1.2 . Thesis Statement

Language modeling consists of evaluating how likely any sequence of tokens[5] (typically words, word pieces, or characters) is to appear in language. The typical approach to achieving this is to learn to predict the probability of a token given the tokens in its context on a large corpus. However, as demonstrated by Zipf (1935), language is dominated by a few common words, and vocabulary increases logarithmically as the corpus grows. In all natural language corpora, there are therefore many rare words which occur insufficiently often to satisfactorily estimate these probabilities. This is especially true in domain-specific text, which often contains words for highly specific concepts. As we would like to avoid incurring the aforementioned penalties that come with increasing corpus size exponentially, we seek a form of *a priori* knowledge which provides us with an alternative source of information.

Several candidate sources of information may come to mind, such as:

- **Grammar rules** which, among other things, sort words into different parts of speech (PoS) and define which constructions are sensible. These rules are generally imperfect and riddled with exceptions, but could conceivably be used to bootstrap sentence-structure understanding and/or as a soft filter to dampen or amplify the probabilities of various tokens depending on their part of speech, identified with rule-based dependency parsing.

- **Weaker language models** used in a reverse-distillation setting, *i.e.* before self-supervision on text, a larger, newer model is pre-trained on the outputs of an older, smaller model, which creates lower-quality but more informative data.

- **Dictionaries and encyclopediæ**, which can be added to most corpora and trained on for multiple epochs[6] or queried to add context during inference.

- **Ontologies and Knowledge Graphs**[7] (KGs) which arrange concepts in a web of relations which capture probabilistic dependencies between concepts (and, therefore, tokens).

[5] Token: "A *token* is an instance of a sequence of characters in some particular document that are grouped together as a useful semantic unit for processing." (Manning *et al.*, 2008, p. 22)

[6] Epoch: complete pass over the linguistic resource

[7] Ontologies are a type of Knowledge Base (KB), but not all Knowledge Bases are technically Ontologies. Knowledge Graphs are a common way to represent an Ontology, but not necessarily the only way. In practice however, these nuances are not critical. Henceforth, we will generally be using the terms 'Knowledge Base', 'Ontology', and 'Knowledge Graph' interchangeably to mean "an ontology which is represented as a knowledge graph for our practical purposes". We tend to prefer 'Knowledge Base' when speaking in general terms and 'Knowledge Graph' when discussing the technical details of an approach which is specific to graphs.

Knowledge graphs are of particular interest as an under-utilized source of high-quality information. Experts in many fields have carefully constructed such knowledge graphs as WorldKG (Dsouza *et al.*, 2021), a geographically-oriented KG; WordNet (Miller, 1995), a general knowledge graph representing synonymy and other lexical relations between words; and the Unified Medical Language System (UMLS), which indexes almost all biomedical concepts, along with their various spellings and different types of mutual relationships.

In this dissertation, we explore the various ways in which we might leverage knowledge graphs in support of Transformer-based language models for the purposes of alleviating the previously stated drawbacks of the naïve approach. Specifically, we will be using UMLS as a basis for our work: while using multiple knowledge graphs in the exploration of our methods for leveraging them would likely yield more nuanced results and a deeper understanding, adapting our methods to each specific KG would cost a significant amount of time which we believe is better spent broadening the scope of our exploration to multiple knowledge integration techniques. We specifically chose UMLS based on the set of challenges that it does and does not pose:

- Lack of text is its own difficult problem to tackle in the context of Transformer-based language models. It is therefore desirable to work within a field where this is not an issue as, if a low-resource scenario is relevant to explore in any given work, this can still be simulated in a non-resource-constrained domain. The biomedical domain has no shortage of textual resources, including millions of Open Access articles (as shown in Figure 1.3), making it a good candidate field to focus on.

- UMLS specifically is the most extensive KG in the biomedical domain, not only affording the most options, but also being large enough to be challenging to manage. Systematic text-based term searches are not tractable at the scale of language model training corpora, requiring specific optimization efforts which lead our research in a productive direction. This also ensures our methods are likely to be tractable on almost all other knowledge graphs, as there are very few knowledge graphs which are comparable in size to UMLS.

- Although this is subjective, we find working on the biomedical domain helps envision applications for our work which may help patients or researchers, and find these prospects particularly inspiring.

Despite our exclusive use of UMLS and focus on the biomedical domain however, the concepts discussed in this thesis are meant to apply to any Ontology and any domain.

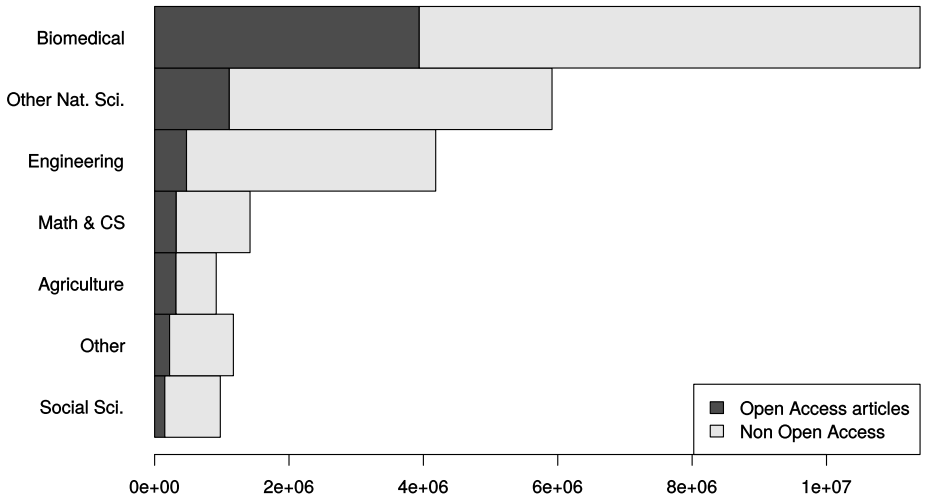


Figure 1.3: Open Access articles by field
Number of Open Access and Non Open Access articles published by field of science between 2009 and 2016. Source of data: European Commission.

In summary, we will explore the integration of knowledge from knowledge graphs into Transformer-based LMs for the purpose of adapting LMs to specific domains while reducing the computational (and non-computational) costs in comparison to the naïve approach. We will closely examine three major approaches in particular, report our findings, and discuss future directions for research on knowledge integration, domain adaptation, and data efficiency in language modeling.

1.3 . Outline

In the following chapter, we will review the literature on the topics of domain adaptation in language modeling and knowledge integration, as well as the prerequisites in the various neural approaches to language modeling and the task of Entity Linking, which is a pivotal task for many approaches to knowledge integration.

Chapters 3 through 5 will be dedicated to my original contributions to the field. First will be discussed the importance of Entity Linking (EL) as a precursor to knowledge integration methods such as ERNIE (Zhang *et al.*, 2019), UmlsBERT (Michalopoulos *et al.*, 2021) or BERT-MK (He *et al.*, 2020), which are based on the identification of entities mentioned in the text, and the extraction of corresponding entity embeddings. In chapter 4, we will examine ways to work around the Entity Linking problem within the entity-embedding-based methods, in particular applying

the KnowBert (Peters *et al.*, 2019) method, which introduces the idea of fuzzy entity linking, to the biomedical domain with UMLS. We will discuss the inherent limitations of this type of approach, and in chapter 5 will consider alternative classes of approaches and explore data augmentation through text generation as a specific instance.

Lastly, chapter 6 will conclude by summarizing these contributions; discussing future avenues for investigation including potential improvements and different approaches to knowledge integration; and reexamining the current direction of Machine Learning and Language Modeling research, the problems faced in the field, and the potential roles of knowledge integration and domain adaptation in that future.

2 - Background & Related Work

2.1 . Language Modeling

Computational language modeling traces its roots to the work of al Kindi (9th century) on the statistical study of language in cryptography. National defense and Intelligence were the primary drivers of progress in what would eventually be known as Natural Language Processing during the mid-twentieth century, with applications in cryptography during the war, most notably Alan Turing's work at Bletchley Park, and in Machine Translation (MT) afterwards and into the early 1960s (Weaver, 1949/1955), until the publication of the unfavorable ALPAC report (Pierce *et al.*, 1966). Influenced by Turing's idea of *the imitation game* (Turing, 1950) as the litmus test for true machine intelligence, the development of the conversational Rogerian psychotherapist chatbot ELIZA (Weizenbaum, 1966) marks a pivotal event in applied computational linguistics, bridging the gap with Chomskyist theory (Chomsky, 1957). Since the 1970s, many ontologies and KBs have been developed as, among other applications, a supporting framework to expand the range of capabilities of symbolic NLP systems influenced by ELIZA such as MARGIE (Schank *et al.*, 1973) and TALE-SPIN (Meehan, 1977). Many of these knowledge bases contain valuable information, distilled to a relatively dense representation, and some remain in use. A notable example of this is the Unified Medical Language System (UMLS) (Lindberg *et al.*, 1993), of which we have made ample use in the context of this research.

Starting in the 1970s, with the introduction of the concept of inverse document frequency (IDF) in particular (Jones, 1973), statistical approaches to computational linguistics have gained traction. As an increasing number of inconsistencies within and across natural languages were found, rule-based models have become increasingly complex. The apparent increasing complexity of the hypothetical set of rules defining the universal grammar posited by Chomsky as the underlying implementation of the human language acquisition device, in addition to the continuous improvement of computer hardware, led to the dominance of these statistical approaches in the 1990s. Improvements to statistical tools for text mining and information retrieval such as LSA (Deerwester *et al.*, 1990) and a breakthrough in corpus-based machine translation spearheaded by IBM with their *alignment models* (Brown *et al.*, 1993) in particular laid the foundation for modern statistical NLP. This coincided with development of the modern *World Wide Web*, which provided both an increasingly large amount of easily harvested text, enabling easy statistical study of language, as well as an incentive to develop these tools in order to organize, index, search, and study the web and its users.

While neural networks were long criticized for their lack of interpretability and computational cost, interest in their applications to text increased throughout the 2000s due in large part to the work of Bengio and Hinton in neural language modeling (Bengio *et al.*, 2000; Morin and Bengio, 2005; Mnih and Hinton, 2008).

Graphics processing hardware and firmware improvements in the late 2000s and early 2010s dramatically increased the time efficiency of training neural networks, and led to increased interest by the machine learning community. The application of Recurrent Neural Networks (RNNs) to language modeling (Mikolov *et al.*, 2010) and subsequent advancements in neural word embeddings attributable to Word2Vec (Mikolov *et al.*, 2013) and GloVe (Pennington *et al.*, 2014) marked a turning point in neural NLP leading to significant progress in all sub-fields, perhaps most notably in sequence-to-sequence tasks such as Neural Machine Translation (NMT) (Bahdanau *et al.*, 2015a) and syntactic parsing (Vinyals *et al.*, 2015) in the case of RNN models and information retrieval and classification tasks (*e.g.* sentiment analysis, topic detection, word and document similarity in the case of word embeddings.

Despite the improvements of pretrained neural word embeddings, the major remaining limitations were the limited vocabulary and 1-to-1 mapping of vectors to words in that vocabulary leading to limited ability to model word polysemy. In the late 2010s, Peters *et al.* (2018) addressed these issues using an RNN-based LM, ELMo, to contextualize distributed word representations using a character-based input. As a result, any character sequence, even if it had never been seen in training, could generate an embedding which took context into account and thus differentiated between the possible meanings of a word.

Another independently-developed major addition to language modeling in the late 2010s was the use of the *self-attention* mechanism to allow models to assign various levels of importance to various input tokens depending on the context. Rooted in the work of Schmidhuber (1992) on *fast weights* in neural networks (weights which are variable at inference), attention mechanisms were popularized in the context of language modeling by Bahdanau *et al.* (2015a) and entirely replaced recurrent architectures in the wake of the publication of *Attention Is All you Need* by Vaswani *et al.* (2017). We discuss the new, fixed-context-size non-recurrent attention-based architectures, called Transformers, in Section 2.3.

2.2 . Recurrent Neural Networks

Whilst in theory, simple Multilayer Perceptrons (MLPs) with fixed inputs and outputs can approximate any function (Cybenko, 1989), they are not capable of handling variable-size inputs such as images, sound, data streams, time series, and text. This problem has been addressed in multiple different ways depending on the application. Scaling (using an algorithm such as nearest-neighbor or bicubic interpolation) and cropping are common approaches in computer vision, while sliding context windows and input padding are common approaches for most other applications. These methods have the drawback of corrupting the input in some way and are limited in the amount of context that can be drawn from. The Recurrent Neural Network (RNN) architecture, based on the work of Rumelhart *et al.* (1986) and Robinson and Fallside (1987) but proposed in its most common form by Elman (1990), is an alternative solution to this problem, allowing for sequential data to be fed *ad libitum* into a network over the course of multiple steps,

during which the network maintains and updates a state vector which memorizes the useful features of the data which has been supplied up to that point.

Many RNN architectures exist, which share the basic trait that the neuron graph contains cycles. The Elman network architecture is the most common and simply includes one hidden layer connected to itself. To resolve the backwards connections, they are considered an output at each time step t, and an input at each time step $t+1$ as shown in Fig. 2.1. Formally, the output $\mathbf{o}_t$ and hidden layer $\mathbf{h}_t$ are expressed as:

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{h}_t + \mathbf{b}_o)$$

$$\mathbf{h}_t = \sigma(\mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_x \mathbf{x}_t + \mathbf{b}_h)$$

Where $\mathbf{x}_t$ is the input at step t; matrices $\mathbf{W}$ and vectors $\mathbf{b}$ are (respectively) connection weights and biases to be learned; and σ is the non-linear activation function (traditionally sigmoid). $\mathbf{h}_0$ is typically initialized to the $\vec{0}$ vector.

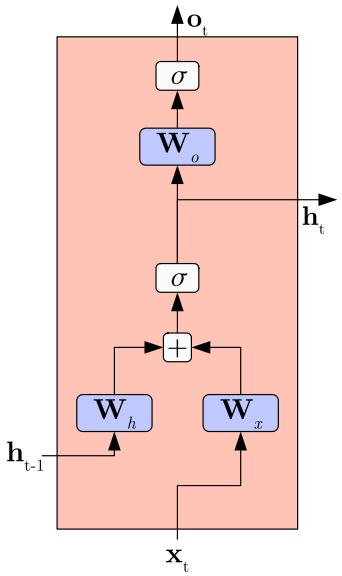


Figure 2.1: Structure of the Elman network RNN viewed at time step t. Bias terms omitted for legibility. Tunable parameters marked in purple.

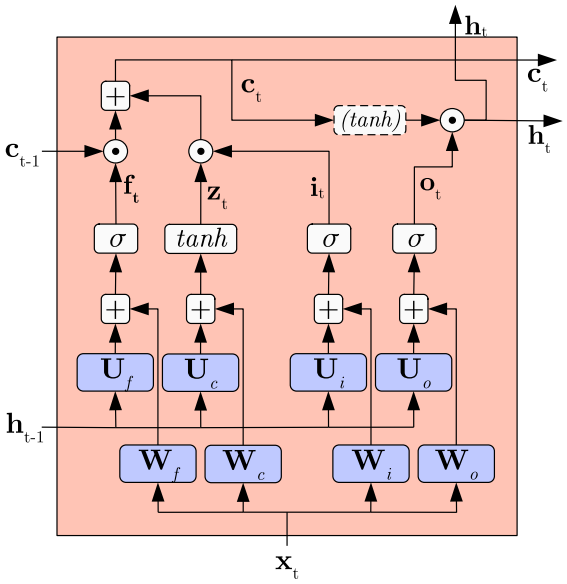


Figure 2.2: Structure of the LSTM viewed at time step t. Bias terms omitted for legibility. Tunable parameters marked in purple. Dashed boxes represent optional steps.

The Recurrent Neural Network architecture can theoretically encode information over any length of context[1] (and is in fact Turing-complete ; Hyötyniemi, 1996), limited in the resolution of the information only by the size of h. However,

[1]For a trivial example (albeit of limited usefulness), it could encode the first token t_0 in the sequence into its representation $\mathbf{h}_0$ and set $\mathbf{h}_{t+1} = \mathbf{h}_t \quad \forall t \geq 0$. More formally, $\mathbf{h}_{t+1} = \sigma(\mathbf{W}_h \mathbf{h}_t + \mathbf{W}_x \mathbf{x}_{t+1} + \mathbf{b}_h)$ where $\mathbf{h}_0 = \mathbf{x}_0$, $\mathbf{W}_h = I$ the identity matrix, and $\mathbf{W}_x = O$ the os matrix.

in its classical form, this architecture is met with difficulty scaling to large numbers of inputs as shown by Bengio *et al.* (1994) and Pascanu *et al.* (2013). In practical NLP, this means RNN-based language models have trouble scaling to multiple sentences. This is fundamentally due to a property of chaining multiplications, that is, that repeated multiplications of numbers whose absolute value is smaller or larger than 1 will tend to 0 or $\pm\infty$ respectively as the number of operations increases, such as is the case with gradient Backpropagation Through Time (BPTT) and iterated forward propagation of inputs. This has two consequences. The first is that as the length of the context grows, gradient descent ceases to be effective at updating the weights of the network, as the gradient approaches 0 ("vanishes") or diverges ("explodes"). The second is that, as there is no mechanism allowing for an explicit, binary decision to be made regarding whether to conserve or discard a piece of information, the slightest degradations to the information will inevitably compound as context size increases. The Long Short-Term Memory (LSTM) network architecture introduced by Hochreiter and Schmidhuber (1997), as well as the architectures it has inspired such as the Gated Recurrent Units (GRU) (Cho *et al.*, 2014) attempt to address these limitations by introducing various sub-networks, or "gates", which explicitly control the information that should be retained and lost, and cause the gradient to be computed in such a way that it is less likely to vanish or explode.

As shown in Fig. 2.2, the LSTM unifies the hidden state vector $\mathbf{h}$ with the output, and adds to it a recurrent explicit *context* memory vector $\mathbf{c}$, which is updated with a candidate information vector $\mathbf{z}$, a *memorization* gate $\mathbf{i}$ (which rescales the components of the candidate vector $\mathbf{z}$), and a *forget* gate $\mathbf{f}$ (which rescales the components of the memory vector $\mathbf{c}$). Additionally, an *output* gate $\mathbf{o}$ is trained to decide which parts of the memory vector to incorporate into the output. Formally, the forward pass is computed thusly:

$$\begin{aligned} \mathbf{h}_t &= f(\mathbf{c}_t) \odot \mathbf{o}_t & \mathbf{z}_t &= tanh(\mathbf{U}_c\ \mathbf{h}_t + \mathbf{W}_c\ \mathbf{x}_t + \mathbf{b}_c) \\ \mathbf{c}_t &= \mathbf{c}_{t-1} \odot \mathbf{f}_t + \mathbf{z}_t \odot \mathbf{i}_t & \mathbf{f}_t &= \sigma(\mathbf{U}_f\ \mathbf{h}_t + \mathbf{W}_f\ \mathbf{x}_t + \mathbf{b}_f) \\ \mathbf{o}_t &= \sigma(\mathbf{U}_o\ \mathbf{h}_t + \mathbf{W}_o\ \mathbf{x}_t + \mathbf{b}_o) & \mathbf{i}_t &= \sigma(\mathbf{U}_i\ \mathbf{h}_t + \mathbf{W}_i\ \mathbf{x}_t + \mathbf{b}_i) \end{aligned} \tag{2.1}$$

Where $\odot$ is the element-wise product, and f is usually $tanh$, constraining the output to the $[-1, 1]$ range, but can be replaced by the $identity$ function.

Thanks in particular to the reliance on addition rather than multiplication for updates to the memory vector, this architecture minimizes the number of successive multiplications in the gradient, reducing its risk of exploding or vanishing. LSTM networks and related gating mechanisms thus prove to be effective at increasing the allowable context size of RNNs. However, this comes at a significant computational cost due to the increased number of parameters to tune.

Additionally, since long-term dependencies must be learned over a number of steps proportional to the distance between terms, these models typically fail to learn to handle context sizes in excess of a few paragraphs of text, which is highly restrictive in terms of the scope of the types of text that can be processed. Another

shortcoming of the RNN architecture resides in its inherently serial structure, *i.e.* the result of time step t must be known to compute the result of time step $t + 1$. Therefore, processing and learning over long sequences, even where possible, is not highly parallelizable.

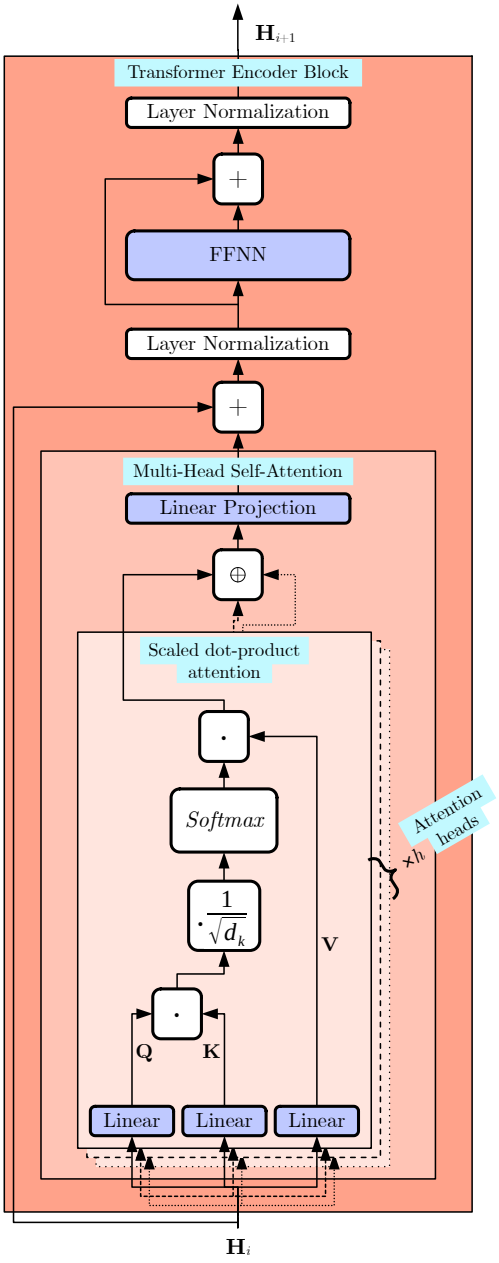


Figure 2.3: Structure of the ith Encoder block in Vaswani *et al.*'s Transformer language model. The tunable parameters are marked purple. $\oplus$ denotes vector concatenation.

The Transformer architecture was introduced by Vaswani *et al.* (2017) as an alternative to Recurrent Neural Networks for language modeling, specifically as a sequence-to-sequence architecture which is trained on the machine translation task. In their landmark paper *"Attention Is All You Need"*, Vaswani *et al.* (2017) demonstrate that the attention mechanism described by Bahdanau *et al.* (2015b), combined with a token representation which includes token ordering information, is sufficient to model dependencies between tokens, and decide which information is relevant for the training objective, without having to resort to a recurrent architecture. As such, the Transformer does not have the benefit of a theoretically unlimited context. However, with a fixed context size, it is no more prone to suffer from vanishing or exploding gradients than traditional FFNNs, and the entire context can contribute equally to the output, meaning that long-range dependencies can be learned just as easily as short-range dependencies provided that the neural network accepts sufficiently many inputs. In addition, the computations are highly parallelizable. Therefore, more computational resources can be leveraged over a shorter time to process larger context sizes more effectively when compared to RNN-based architectures. In this section, we go over the main components of the Transformer architecture, the main language models in the literature, and their contributions to the current state of the art.

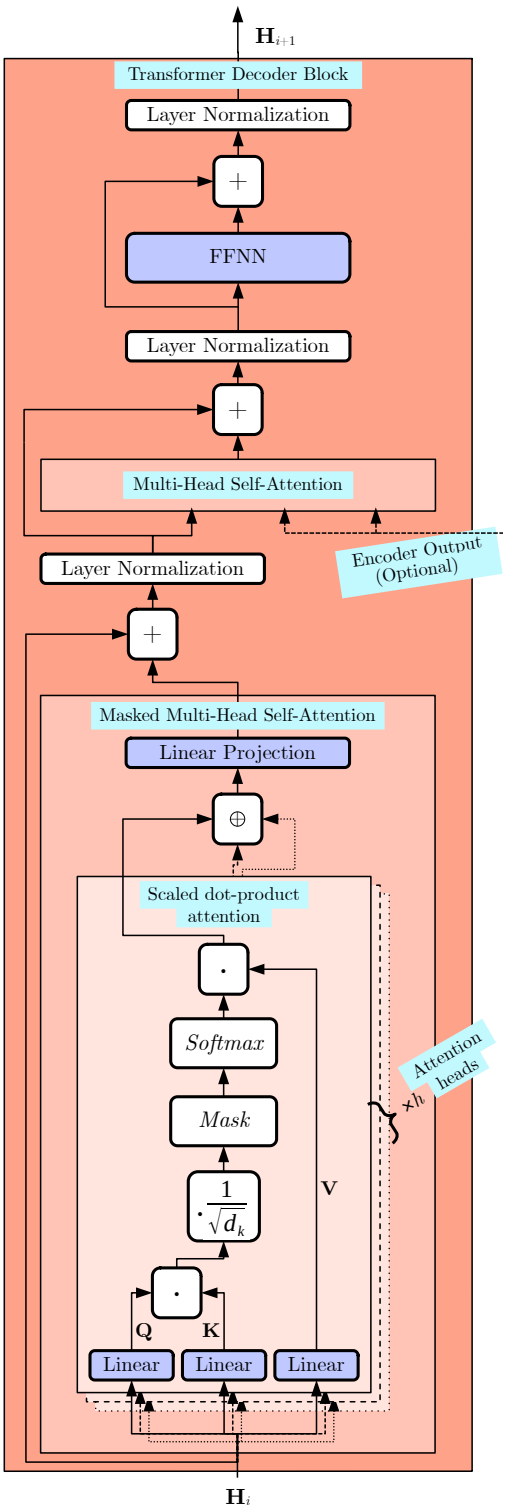


Figure 2.4: Structure of the ith Decoder block in Vaswani *et al.*'s Transformer language model. The tunable parameters are marked purple. $\oplus$ denotes vector concatenation. For Multi-Head Self-Attention details, see Fig. 2.3

Vaswani *et al.*'s Transformer is a translation language model which uses Transformer Encoder blocks (Fig. 2.3) to encode input text from a source language and Transformer Decoder blocks (Fig. 2.4) to generate a sentence in the target language, informed by the representation given by the encoders. The decoder block is almost identical to the encoder block but includes a Masked Multi-Head Self-Attention module before the Multi-Head Self-Attention module, and the Multi-Head Self-Attention module accepts the output of the Encoder blocks as input. The Masked Multi-Head Self-Attention module differs from the (non-masked) Multi-Head Self-Attention module in the fact that it prevents the attention mechanism from propagating information backwards in the sequence — for each word, information from its predecessors can be attended to, but its successors are "masked". The original model uses 6 encoder blocks, 6 decoder blocks and $h = 8$ attention heads. This architecture works well for translation and similar tasks, but is purely a sequence-to-sequence architecture and is fairly complex, requiring many parameters. The Transformer uses byte-pair encoding (Sennrich *et al.*, 2016), which works around the problem of out-of-vocabulary words and is resilient to typos in input text without limiting context size as much as character-based tokenization by representing common character combinations as single tokens. Subsequent Transformer-based architectures tend to use the similar WordPiece tokenization (Wu *et al.*, 2016).

ELMo (Peters *et al.*, 2018), though it was based on the LSTM architecture, introduced the concept of *contextualized embeddings*. That is, rather than assigning a fixed vector to every word in a predetermined vocabulary as per Word2Vec, each token in a sequence can be assigned a vector which accounts for its general use and its local context. Therefore, polysemic words can be differentiated.

This approach can also detect when a word is used in a novel context which has not been seen in training. This was achieved by training the output representations to optimize for the autoregressive language modeling objective, which consists of predicting the next word in a sequence. To be more specific, ELMo does this bidirectionally using a bidirectional LSTM. Optimizing for this objective yields token representations which model grammatical and semantic information for that token given its context, which can then be used for a variety of downstream tasks such as Named Entity Recognition (NER) (*i.e.* tagging words in a sequence which mention an entity of interest and typically assigning it a type such as "location" or "person"), Part of Speech (PoS) tagging (*i.e.* tagging words based on their grammatical function in the text), or Natural Language Inference (*i.e.* recognizing the type of relation between two sequences such as "contradiction" or "entailment"). Peters *et al.* (2018) further introduce the idea of fine-tuning the language model for a specific task. This approach is further expanded upon and standardized by the ULMFiT approach (Howard and Ruder, 2018).

GPT (Radford *et al.*, 2018) eschews the encoded context sequence used by Vaswani *et al.*'s translation model and adopts a purely generative architecture based on the Transformer Decoder block (Fig. 2.3) which is optimized for unidirectional, autoregressive language modeling (*i.e.* predicting the next word in a sequence). As it does not integrate Encoder output, it also simplifies the Decoder block by removing the Multi-Head Self-Attention module. This yields a model that is particularly well-suited to generative tasks, such as conversational agents and creative writing. Crucially, as shown by GPT-2 and Instruct-GPT (Radford *et al.*, 2019; Brown *et al.*, 2020; Ouyang *et al.*, 2022), many tasks can be formulated as generative and/or conversational tasks, such as summarization and translation. With 12 transformer decoder blocks and 12 attention heads, GPT comprises approximately 117 million parameters.

BERT (Devlin *et al.*, 2019) adapts the ELMo approach of word embedding contextualization to Transformers. It takes the opposite approach to GPT and uses only Transformer Encoder blocks (Fig. 2.3) to derive contextualized distributed word representations. It adapts ELMo's bidirectional objective to the transformer by introducing the Masked Language Modeling (MLM) objective: words in the input sequence are masked with some probability (15% in the case of BERT), and the model must learn to predict which words were masked. In addition to these adaptations, BERT includes a special token called `[CLS]` which models the entire sequence for sequence classification purposes. Lastly, Devlin *et al.* (2019) introduce the Next Sentence Prediction (NSP) objective which consists of feeding pairs of sequences to the model and teaching it to differentiate between sentences which are contiguous in the original text from sentences which are unrelated to each other. BERT is particularly effective in the same contexts as ELMo, *i.e.* Natural Language Understanding (NLU) tasks (which typically involve classification). However, while not as effective at creating coherent text as the GPT approach, by masking the last word(s) in a sequence, BERT can be used as a generative

language model. With 12 transformer encoder blocks and 12 attention heads (the same as GPT), $\text{BERT}_{\text{BASE}}$ comprises approximately 110 million parameters. Devlin *et al.* (2019) also released a 24-encoder 16-head 340 million parameter model dubbed $\text{BERT}_{\text{LARGE}}$. When not specified, "BERT" generally refers to $\text{BERT}_{\text{BASE}}$.

Input text in transformers is generally represented as a sum of multiple embeddings for each token:

- Each token in the vocabulary is associated with a *token embedding* which may be pretrained or learned as part of the language model training.
- *Position embeddings* are token-invariant embeddings assigned to the positions in the input, generally based on a mixture of sine waves at different frequencies.
- In models which may take multiple input sequences such as BERT, *sequence embeddings* (which are generally trained in tandem with the LM objective) may be added to differentiate them.

These are the most standard types of input embedding components, but some approaches such as UmlsBERT (Michalopoulos *et al.*, 2021) may introduce additional components.

BERT is highly effective in NLU tasks and being a small enough model to study for most organizations, the barrier to entry is low in studying BERT-based architectures, making it a prime choice for many NLP research applications. GPT-based architectures only display high enough performance to be useful at the scale of GPT-2, which as approximately 1.5 billion parameters — approximately 14 times larger than $\text{BERT}_{\text{BASE}}$. Many improvements have thus been made to BERT in the NLP literature. Notably, it has been found that in practice, the NSP objective is ineffective, or even counter-productive (Yang *et al.*, 2019; Liu *et al.*, 2019; Lan *et al.*, 2019). Notable contributions to the BERT literature (sometimes referred to as "BERTology" (Rogers *et al.*, 2021)) include the findings of Liu *et al.* (2019) on transformer hyperparameterization and the effects of corpus size, Yang *et al.* (2019)'s autoregressive extension of BERT, and improvements on BERT's scaling by Lan *et al.* (2019) (particularly in terms of memory requirements). For further review and examples of landmark Transformer-based language models in the context of the encoder-based and decoder-based families, readers are directed to the work of Yang *et al.* (2023).

2.4 . Domain Adaptation

In the wake of the advent of the Transformer architecture introduced by Vaswani *et al.* (2017), language processing tasks have increasingly been handled by neural language models based upon this architecture such as GPT (Radford *et al.*, 2018) and BERT (Devlin *et al.*, 2019). Many projects have sought to leverage the Transformer architecture in specific fields ; for instance in the process of *discovery*, in the legal domain (Yang *et al.*, 2022), which has traditionally required lawyers to sift through large numbers of documents, or patient risk modeling for disease prognosis in the medical domain (Li *et al.*, 2022). Due to the distributional shift

between the language used in specific fields and the types of text typically used to train these models however, general language models tend to underperform in these fields (Yang *et al.*, 2022; Lee *et al.*, 2019).

2.4.1 . Definition

Most definitions of domain adaptation, such as the one provided by Ruder (2019), describe it as a type of transfer learning characterized by the source and target settings of the model differing in data distribution rather than task, as opposed to fine-tuning, which focuses on adapting a model to a new task. For instance, in computer vision, training a model to categorize dogs and cats before re-training it to categorize cancerous versus non-cancerous cells would be an example of domain adaptation, whereas training a model to reconstruct corrupted images (a task which seeks to teach general features of images) before changing the model objective to classification would be an example of pretraining and fine-tuning.

However, on one hand, transfer learning fundamentally seeks to *retrain* an existing model to perform well on a certain task for a certain distribution of data which may differ from the task and data for which the original model was optimized. On the other, domain adaptation seeks simply to efficiently re-use a model, which was trained on a given task and source distribution, with a new target distribution. The latter does not inherently involve additional learning and therefore does not entail the former. Consequently it is arguably inaccurate to describe domain adaptation as a type of transfer learning in theory, despite most practical implementations of domain adaptation also being instances of transfer learning. We thus broaden the scope of domain adaptation and define it thus:

> *Domain Adaptation is any process by which an existing model, optimized for a source data distribution, is altered so that it may better perform on a target data distribution.*

Desirable characteristics for domain adaptation techniques include:

- Increased data efficiency in comparison with training the model from scratch on the target domain
- Increased computational efficiency in comparison with training the model from scratch on the target domain
- Applicability to models not specifically intended for domain adaptation
- Non-necessity of taking target domain into account for the creation of the source-domain model
- Reverse-compatibility with the source distribution

2.4.2 . Historical approaches

In its most basic form, domain adaptation consists of correcting a biased statistical classifier trained on N_s sample-label pairs (x, y) drawn from the source joint distribution $D_s(X, Y)$ and deployed on data from the target joint distribution $D_t(X, Y)$. It may also be trained on N_t pairs drawn from the target joint distribution $D_t(X, Y)$ with $0 < N_t \ll N_s$. This is a fundamental problem in statistics (Shimodaira, 2000) and a classic problem in econometrics (Heckman (1979), Zadrozny (2004)), but has been studied under the name of *domain adaptation* as a machine learning problem since the mid-2000s with the work of Daumé III and Marcu (2006); Blitzer *et al.* (2006); and Jiang and Zhai (2007).

The classical approach to solving this problem as described by Jiang (2008) is to insert a bias correction term in the training of the classifier. The optimal set of parameters θ^* for the model is obtained by the following equation:

$$\theta_t^* = \underset{\theta \in \Theta}{\operatorname{argmin}} \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} P_t(x, y) \mathcal{L}(x, y, \theta) \tag{2.2}$$

$$= \underset{\theta \in \Theta}{\operatorname{argmin}} \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} \frac{P_t(x, y)}{P_s(x, y)} P_s(x, y) \mathcal{L}(x, y, \theta) \tag{2.3}$$

Where Θ is the parameter space, $P_t(x, y)$ (resp. $P_s(x, y)$) is the joint probability that the sample-label pair (x, y) occurs in the target (resp. source) distribution, and $\mathcal{L}(x, y, \theta)$ is the loss of the model on that pair given the parameterization θ. This can be approximated as

$$\hat{\theta}_t = \underset{\theta \in \Theta}{\widetilde{\operatorname{argmin}}} \sum_{i=1}^{N_s} \frac{P_t(x_i, y_i)}{P_s(x_i, y_i)} \mathcal{L}(x_i, y_i, \theta) \tag{2.4}$$

Where $\widetilde{\operatorname{argmin}}$ is an optimization algorithm such as SGD. While $P_s(x_i, y_i)$ can be estimated using the observed sample $\tilde{P}_s(x_i, y_i)$ from the source distribution, estimating $\lambda \leftarrow \frac{P_t(x_i, y_i)}{P_s(x_i, y_i)}$ is not straightforward because (x_i, y_i) are from the source domain and the target distribution sample $\tilde{P}_t$ is small. Many domain adaptation techniques consist of finding satisfactory estimates for λ.

Class Imbalance: Under this special case, we assume the conditional distributions of the samples given the labels are the same in both domains, but the marginal distributions of the labels may differ. *I.e.* $D_s(X|Y = y) = D_t(X|Y = y)$ but $D_s(Y) \neq D_t(Y)$. Integrating this assumption into the formulation for λ:

$$\frac{P_t(x, y)}{P_s(x, y)} = \frac{P_t(y)}{P_s(y)} \frac{P_t(x|y)}{P_s(x|y)} \tag{2.5}$$

$$= \frac{P_t(y)}{P_s(y)} \tag{2.6}$$

This ratio of label occurrences can typically be satisfactorily estimated using $\lambda \approx \frac{\tilde{P}_t(y)}{\tilde{P}_s(y)}$ (Lin *et al.*, 2002). This is roughly equivalent to resampling the source distribution in order to match the target label distribution.

Covariate Shift: This special case is the reverse of the former, in which we assume the conditional distributions of the *labels* given the *samples* are the same in both domains, but the marginal distributions of the *samples* may differ. *I.e.* $D_s(Y|X = x) = D_t(Y|X = x)$ but $D_s(X) \neq D_t(X)$. In traditional single-step training procedures (as opposed to multi-objective and pretrain/fine-tune procedures), this problem is only relevant if the model cannot perfectly fit the entire distribution and must fit the dense regions of X preferentially (Shimodaira, 2000). Following the same bayesian substitution process, we derive:

$$\frac{P_t(x, y)}{P_s(x, y)} = \frac{P_t(x)}{P_s(x)} \tag{2.7}$$

As each x typically appears rarely in the dataset, estimating $\frac{P_t(x)}{P_s(x)}$ is not straightforward. Multiple methods have been proposed such as adding λ as a learned parameter of the model (Bickel *et al.*, 2007) and non-parametric kernel density estimation (Shimodaira, 2000).

For a more in-depth overview of this class of methods, we refer readers to the work of Jiang (2008). However, these approaches are not typically applicable in the context of language modeling. To illustrate this, let us take the basic case of autoregressive language modeling, which models the probability of each token as being conditioned by its preceding context (*i.e.* left-hand context in left-to-right languages). In this case, $\mathcal{X}$ is the set of all contexts and $\mathcal{Y}$ is the set of all tokens. In theory, context may be infinite, but in practice is bounded — however, larger contexts are preferable as they allow for the modeling of long-range dependencies. As context size increases, the number of possible contexts increases, $P_t(x)$ and $P_s(x)$ converge towards 0, and $\frac{P_t(x,y)}{P_s(x,y)}$ is thus undefined.

Other methods include the proposal by Daumé III (2007) to learn a mapping from a k-dimensional input space to a $(n+1)k-$dimensional input space where n is the number of domains. In essence, this approach learns domain-specific functions for words, *e.g.* the word *short* would likely be predominantly used as an adjective in the general domain, but a verb in the domain of economics.

Maximum Mean Discrepancy (MMD) is a method for estimating the difference between two distributions from a set of samples, defined as the difference between distributions in the reproducing kernel Hilbert space (Gretton *et al.*, 2006). This has been used as a basis for transfer component analysis (Pan *et al.*, 2010), which projects sample representations into a lower-dimensional space which minimizes MMD, and as a loss for domain regularization in order for the model to learn domain-invariant latent representations (Ma *et al.*, 2019). However, MMD by construction reduces feature discriminability (Wang *et al.*, 2021a).

Grassmann Manifolds have been used by Gong *et al.* (2012) and Gopalan *et al.* (2011) to find domain-invariant representations. This is done by embedding the domains in lower-dimensional subspaces, constructing a geodesic curve in the Grassmannian, and either sampling subspaces from that curve in order to perform projections of the data and construct a unified embedding through concatenation (Gopalan *et al.*, 2011) or using this geodesic to compute a kernel (subsequently used to build a kernel-based classifier) (Gong *et al.*, 2012).

Autoencoders are neural networks which are trained to output their own input (or an uncorrupted version of a corrupted input) but contain a small hidden layer so as to create a representation bottleneck forcing the model to learn to compress the input. They have been used in a similar fashion to Grassmann manifolds (Glorot *et al.*, 2011; Chen *et al.*, 2012): the objective remains to find a smaller, domain-invariant representation of the input. Neural Structural Correspondence Learning (NSCL) was introduced by Ziser and Reichart (2017) as a variation on the autoencoder approach which adds support for an explicit mechanism for learning to leverage both cross-domain and domain-specific features in tandem. In practice however, Lin *et al.* (2020) do not find NSCL to be helpful in the context of Transformer-based LMs.

Approaches based on minimizing MMD, Grassmann manifolds or autoencoders, which seek to find domain-invariant representations, display a significant limitation, as they essentially consist of finding the general features which apply across domains. This can be seen as a compromise between source and target domain, improving performance on the target domain at the cost of source-domain performance by neglecting source-domain-specific features. This performance reduction, in traditional single-stage model training curricula, is simply *underfitting*, whereas in more complex settings such as the pretrain/fine-tune paradigm, this problem is referred to as *catastrophic forgetting*. In an ideal scenario, we would prefer to be able to leverage the specificities of both domains. This is by definition easy (and in fact the default behavior) in the source domain, but inherently requires, if not data, some *a priori* knowledge of the target domain. Additionally, these techniques require knowledge of the target domain before the model is optimized for the source domain, reducing their scope of applicability.

Another major drawback of these domain adaptation methods is that they must be applied for each individual sub-task, with most of them being restrained to classification tasks such as part of speech tagging or named entity recognition in NLP. In a pretrain/fine-tune paradigm, these methods would therefore have to be applied at the fine-tuning stage, and would require sets of labeled data for both the source and target domain within each task, multiplying the need for annotated data and further narrowing their range of relevancy. In addition, these methods are limited to leveraging the information in the source domain and a small amount of information from the target domain to perform domain adaptation. In Natural Language Processing, this may not be sufficient as relationships between entities can often only be learned from resources in the target domain; *e.g.* information on the relationship between pancreatic β-cells, insulin production and diabetes is

unlikely to occur in general-domain text, and no corrective factor or other mathematical trick could induce modeling of these relationships by the language model if the information is not presented in some form. We therefore seek to make use of additional sources of information while minimizing the negative effects of increasing corpus size.

2.4.3 . In computer vision

Domain adaptation for deep neural networks was largely popularized by the success of transfer learning in computer vision (Yosinski *et al.*, 2014; Girshick *et al.*, 2014), where fine-tuning image classification models trained on natural image datasets such as ImageNet (Deng *et al.*, 2009) yielded highly superior results to models trained from scratch on domain-specific image datasets, which often were comparatively small. The most notable examples of this were in the biomedical domain (Greenspan *et al.*, 2016), and particularly for detection of cancerous growths (Esteva *et al.*, 2017) and lung disease (Shin *et al.*, 2016), where human or arguably super-human level has been reached (Rajpurkar *et al.*, 2017; Goh *et al.*, 2020).

2.4.4 . In-domain extended pre-training in Transformers

Transformer-based language models, when trained mostly on general text such as is the case with BERT (Devlin *et al.*, 2019) and GPT (Radford *et al.*, 2018), for which the bulk of the training corpus is formed by the *Books* corpus (Zhu *et al.*, 2015), do not perform well on tasks involving specific domains such as medicine, patents, or law. This is to be expected, as many of the writing conventions differ by field and make assumptions on the knowledge of the reader. The obvious way to expand the capabilities of a language model to a specific domain is therefore to include in-domain text in the model's training corpus. This has indeed been the typical approach, and has been quite successful, with its performance having been investigated by Gururangan *et al.* (2020), and with notable models such as GeoBERT (Gao *et al.*, 2022), LEGAL-BERT (Chalkidis *et al.*, 2020), or PatentBERT (Lee and Hsiang, 2020). The biomedical domain in particular has inspired a variety of models which all incorporate various amounts and proportions of biomedical text in their pre-training phase, *e.g.* BioBERT (Lee *et al.*, 2019), BlueBERT (Peng *et al.*, 2019), BioMed-RoBERTa (Gururangan *et al.*, 2020), SciBERT(Beltagy *et al.*, 2019), and ClinicalBERT (Alsentzer *et al.*, 2019).

However, not only is this type of pre-training sensitive to catastrophic forgetting — *i.e.* it usually leads models to underperform on general-domain text as demonstrated by Arumae and Bhatia (2020) and Xu *et al.* (2020); large models with attention mechanisms such as BERT are also notoriously computationally expensive to pre-train. Furthermore, the ability for a model to associate concepts (*e.g.* "*COVID-19*" and "*respiratory failure*") is predicated on these concepts appearing in the pre-training corpus, leading to difficulty adapting to some forms of distributional shift. These limitations have lead to interest in different methods of adapting these models to specific domains.

2.4.5 . Catastrophic Forgetting

As models trained on one task and data distribution are repurposed for a new task (transfer learning) and/or a new distribution (domain adaptation), the fine-tuning process has no incentive to allow the model to continue performing well on the source task/domain, and critical parameters can be modified. This not only lowers performance of the model on the source task/domain (which may be problematic for large models which are expensive to run or host as a service), but also may lead optimizers such as Stochastic Gradient Descent (SGD) or Adam to converge to poor local optima on the target task/distribution. This problem, which we've touched on as a weakness of some existing domain adaptation techniques, is known as Catastrophic Forgetting (CF).

Experience Replay (Isele and Cosgun, 2018; Arumae and Bhatia, 2020) may be the most obvious way to address CF. Experience Replay is a type of Multi-task learning which specifically re-uses the training data from the source task in order to alternate source and target task and/or distribution in the model's training curriculum. This provides the model with the missing incentive to conserve performance on the source task/distribution, but as the objectives are alternated rather than united, the Experience Replay mechanism by construction moreso undoes the damage done by the fine-tuning process than protects previously acquired knowledge, creating an inefficient optimization path in the loss landscape. Combined with the inherent additional computation of replaying past data, it is a computationally expensive way to counter CF.

Adapters (Houlsby *et al.*, 2019; Pfeiffer *et al.*, 2020) mitigate CF by freezing all model parameters after pretraining and adding small layers of new parameters within Transformer blocks which take on the responsibility of fitting to the target fine-tuning task. While originally formulated in the context of transfer learning for fine-tuning pretrained models to specific tasks, it has also proven effective in protecting against catastrophic forgetting and improving data efficiency in domain adaptation (Zhang *et al.*, 2021).

Elastic Weight Consolidation (EWC) (Kirkpatrick *et al.*, 2017) is a CF mitigation technique which consists of penalizing updates to parameters proportionally to how critical they are for the original task and data distribution for which the model was optimized. This has been used to some success in Transformers (Arumae and Bhatia, 2020; Xu *et al.*, 2020), but is by construction less effective at reducing CF than Adapters for the benefit of better parameter efficiency.

2.5 . Knowledge Integration

Knowledge-based systems have been used in conjunction with neural networks since the beginnings of multi-layer neural networks, for general purposes (Oliver *et al.*, 1988) as well as for a variety of specific tasks such as robotic agent control (Handelman *et al.*, 1990) and gene recognition in DNA sequences (Towell *et al.*, 1990; Noordewier *et al.*, 1990). This approach has mostly fallen out of favor since the deep learning resurgence of 2012 as progress has continued on neural network architectures and training procedures. Some work on this topic has continued however, and influenced the current landscape of the field of knowledge integration as it undergoes a resurgence of its own.

In contrast to typical domain adaptation techniques, knowledge integration, sometimes referred to as Knowledge-Infused Learning (Kursuncu *et al.*, 2019; Sheth *et al.*, 2019; Wang *et al.*, 2020; Yuan *et al.*, 2022), relies on making use of available *a priori* knowledge. The core trait of this type of approach is that the neural model must *make use of* the prior knowledge, improving the bottom-up learning procedure with a top-down knowledge-based component. These approaches therefore do not include:

- using prior knowledge of a problem to decompose the learning objective into multiple sub-tasks as described by Oliver *et al.* (1988)
- neuro-symbolic approaches such as proposed by Kroshchanka *et al.* (2021) which augment relatively general knowledge-based architectures with narrower neural subsystems

More generally, we do not consider that using prior knowledge of the problem in order to devise a fully bottom-up model architecture or overall top-down AI system which is more effective or data-efficient than learning the task end-to-end constitutes Knowledge *Integration*.

In addition to its applicability to the enhancement of general language models at the pre-training stage, this approach has multiple advantages. Depending on the implementation of the integration, the model may be able to take advantage of knowledge base concepts and relations never encountered in training, allowing the model to progress without additional training as KBs are updated and reducing the amount of in-domain concepts that must be covered by the training corpus. As KBs are a much denser source of information than raw text, effectively leveraging them may also potentially reduce the amount of in-domain text required compared to that of extended-pre-training-based methods. In addition, models which require less in-domain training data are not only applicable to a wider variety of domains and less costly to deploy, they are also less sensitive to catastrophic forgetting (Piat *et al.*, 2022a). Lastly, use of structured knowledge can help with explainability (Gaur *et al.*, 2021), which is critical in high-stakes applications such as healthcare, finance and education.

Knowledge may take multiple forms such as rules or heuristics, but the most common approach is to use pre-existing KBs. Typically, this is done by leveraging the KB at the pre-training stage, meaning the domain adaptation step needs only

be carried out once to benefit all of the various downstream tasks. In general language models, this is usually achieved by identifying knowledge base concept occurrences (or "mentions") in the input text, extracting relevant concept and/or relation embeddings from the knowledge graph which are precomputed using a graph embedding algorithm such as TransE (Bordes *et al.*, 2013), and using the knowledge embeddings to improve the language model's distributed word representations.

There are three main factors which characterize knowledge integration methods:

- The type of prior knowledge: typically rules, heuristics, or knowledge graphs.
- The method of integration: typically as additional input to the network, as an external resource, or as a mechanism for altering the network architecture or loss-function.
- The representation of the knowledge, which is most commonly vectorial, but often tailored to the model based on the previous two factors.

In this section, we delve into the various approaches that have been attempted and further elaborate on the landmark articles which have influenced the direction of our work and of the field as a whole. First, we will go over non-KG-based approaches and the relevant knowledge representations. Next, as KGs are almost exclusively used in language modeling, we will go over the dominant approaches to KG integration in RNN based LMs. Lastly, we will discuss the approaches used in Transformer-based LMs.

2.5.1 . Overview of common knowledge types and related approaches

In *The Integration of* A Priori *Knowledge into a Go Playing Neural Network*, Enzenberger (1996) provide a good example of the use of rules and heuristics in the form of "experts" to augment the capabilities of a Go-playing neural network. An "external expert" implements simple known patterns and strategies which can override the decisions of the network in well-documented board configurations. A "feature expert" alleviates the burden of learning useful patterns by implementing prior known types of patterns and feeding a higher-level representation directly to the neural network. Lastly, a "relation expert" imposes a direct relation between neurons and elements on the board, allowing connections only between neurons corresponding to elements that are adjacent on the board, which change as the game progresses. Trained with self-play, this approach was shown to be more effective than both neural networks and rule-based systems of the time. This type of knowledge integration which informs or alters the very structure of the neural network has been explored in other projects (Neal, 1995; Piat and Stamou, 1999; Towell and Shavlik, 1994; Tan, 1997) but this type of approach is more difficult to put in place with the worse explainability of current, larger models.

Some domain knowledge may take the form of logic rules. These have also been used to construct neural network topologies such as for the *Knowledge-Based Artificial Neural Networks* proposed by Towell and Shavlik (1994). In this approach,

the key step is the *rules-to-network translator*, which defines the topology of the neural network as well as the parameters such that the predictions of the neural network are consistent with the rules. Additional parameters are then added, to account for unknown rules, and the resulting neural network is trained as per normal with the exception of some parameters being frozen if they correspond to known hard rules which must not be altered. Fu (1993) puts forth a similar idea, with a reverse *network-to-rules* translation element which additionally enables extraction of new symbolic rules from the neural network. Tan (1997) proposes a roughly equivalent approach, but in the context of Adaptive Resonance Theory.

Inductive Logic Programming (ILP) (Muggleton, 1991) is a similar machine learning approach which uses prior logical knowledge, typically expressed as Horn clauses, including positive statements, negative statements and constraints, in order to produce a logic program which results in all positive statements being true and negative statements being false, with constraints typically implemented as part of the network architecture (d'Avila Garcez and Zaverucha, 1999). ILP is often tackled using neural networks, which require a vector as input. The vectorization scheme for clauses is typically propositionalization, which is thematically similar to *embedding* (Lavrač *et al.*, 2020) but uses a boolean value range. While initially based on manually engineered features, automated feature construction for propositionalization was introduced by Lavrač *et al.* (1991).

Domain Specific Languages (DSLs) are computer languages which are designed for a specific goal, such as HTML for describing web pages, or AWK for text processing. Some DSLs are constructed to be used specifically by a program-induction algorithm (and as such constitute a form of prior knowledge), some of which are based on neural networks such as the work of Lake *et al.* (2015) on handwritten digit recognition and generation, and of Devlin *et al.* (2017) on robotic motion. DREAMCODER (Ellis *et al.*, 2018) additionally learns to construct the appropriate DSL.

Some knowledge integration makes use of their prior knowledge in the loss function. The simplest way to achieve this is generally to introduce a loss penalty informed by prior knowledge (Fischer *et al.*, 2019; Muralidhar *et al.*, 2018). On an abstract level, one would use prior knowledge to formulate constraints and penalize model predictions which violate these constraints. This is often done using knowledge graphs. For instance, Rocktäschel *et al.* (2014) extract hypernymy information from WordNet and induce the modeling of these relations by penalizing scenarios where statements which are true are predicted to be false when the entities mentioned are replaced by hypernyms. Takeishi and Akimoto (2018) provide a loss function regularization for continuous data which assumes a knowledge graph exists which covers the various features as concepts and that related features have similar distributions in the dataset.

Other approaches use logic rules, such as work by Xu *et al.* (2018), for instance, who devise a penalty term which models the likelihood that a given model output satisfies a set of logical constraints. In a similar setup to Iterated Distillation and Amplification (Christiano *et al.*, 2018; Hinton *et al.*, 2015), whereby a model is trained to imitate an amplified version of itself, Hu *et al.* (2016) propose to train

a model to imitate a rule-constrained version of itself in order to distill rules into the neural architecture.

Prominent examples of the approach by loss function alteration in Transformer-based language models include UmlsBERT (Michalopoulos *et al.*, 2021), which we will go over in Section 2.5.4, and KEPLER (Wang *et al.*, 2021b) which optimizes both a masked language modeling objective and a knowledge embedding objective which uses text descriptions of concepts to construct concept embeddings and enforces desirable geometric properties to derive relation embeddings similarly to TransE (Bordes *et al.*, 2013) (see Section 2.5.2).

2.5.2 . KG-based approaches

Knowledge Graphs exist for many domains and generally contain widely applicable prior knowledge. They are also generally reliable, fairly standardized and non-proprietary. This makes them good candidates for knowledge integration, particularly for domain-specific tasks, and are thus used in a variety of methods and applications in the literature.

Perhaps the most obvious lead in terms of processing Knowledge Graphs would be using Graph Neural Networks (GNNs), a family of neural networks whose architectures are designed to process data organized as graphs. This approach has, for instance, been used to pre-train a discriminator in a Generative Adversarial Network (GAN) setup for drug design (Dash *et al.*, 2021); as a knowledge-based component of a multi-expert system predicting the answer to a Visual Question-Answering (VQA) problem (Marino *et al.*, 2021); or to fuse text and KG-based inputs in Transformers (Zhang *et al.*, 2022; Yasunaga *et al.*, 2022). GNNs are most popular in graph-oriented tasks however, as they lack flexibility in terms of their ability to interface with other modalities. In the context of knowledge graphs, they are therefore more suited to tasks such as knowledge graph completion, concept classification, or knowledge graph merging, making them a generally less compelling processing option than graph embeddings when supporting other tasks for which the knowledge graph is not the central component. For further review of the literature, we direct readers to the work of Ye *et al.* (2022) for the use of GNNs with Knowledge Graphs, and of Wu *et al.* (2023) for a more comprehensive overview of the use of GNNs in NLP.

In *Leveraging Lexical Resources for Learning Entity Embeddings in Multi-Relational Data*, Long *et al.* (2016) use pretrained Word2Vec embeddings (Mikolov *et al.*, 2013) and Knowledge Base concept descriptions as an initialization for the TransE knowledge graph embedding algorithm (Bordes *et al.*, 2013). As such, they are effectively modifying word embeddings so as to take information from the knowledge base into account. TransE achieves this by imposing that, for a triple (c_1, r, c_2) where c_1 and c_2 are concept nodes and r a relation edge, their corresponding vectors are bound by the quasi-equality $\mathbf{c}_1 + \mathbf{r} \approx \mathbf{c}_2$. This is consistent with existing known properties of the Word2Vec embedding space, *e.g.* $\overrightarrow{king} - \overrightarrow{man} + \overrightarrow{woman} \approx \overrightarrow{queen}$ or $\overrightarrow{france} + \overrightarrow{capital} \approx \overrightarrow{paris}$ where $\overrightarrow{word}$ denotes the embedding for "word". The concept and relation embeddings can therefore be used in place of their pretrained word embedding counterparts in addition to func-

tioning as node and vertex embeddings. Celikyilmaz *et al.* (2015) follow a similar method to achieve the same objective, but find that adding a concurrent language modeling objective and adding neighboring concepts to the context improves performance. No mechanistic explanation for this improvement is given, but it is likely that the language modeling objective reduces catastrophic forgetting. Wang *et al.* (2015) attempt to achieve a similar result by integrating KG relation information in the word embedding objective. These approaches are limited however, as the cases for multi-word concepts, synonyms of concepts unaccounted for in the KB, homographs, *etc.* are non-trivial to account for and can be quite common depending on the knowledge graph. Furthermore, with fixed-vocabulary models, not all concepts in the knowledge graph may have a corresponding pretrained embedding, limiting the usefulness of the method in a domain adaptation setting. While this approach, unlike many others, does not require Knowledge Base concepts to be identified in the text, the reverse problem applies: one must still automate the matching of KB concepts and relations with existing embeddings, which would likely introduce many mistakes of the same variety. Lastly and most significantly, these methods rely on embeddings being static and are therefore not directly applicable to contextualized word embeddings.

For further in-depth discussion on the aforementioned approaches, readers are directed toward the work of Dash *et al.* (2022), Von Rueden *et al.* (2021), Kim *et al.* (2021) and Colon-Hernandez *et al.* (2021).

2.5.3 . In RNN LMs

2.5.3.1 NKLM

In *A Neural Knowledge Language Model*, Ahn *et al.* (2016) propose a mechanism which enables a generative LSTM-based LM to choose whether to generate a word from its vocabulary or from a Knowledge Graph. The KG is represented as a set of "facts" $\mathcal{F}$, *i.e.* triples $a = (c_{a,1}, r_a, c_{a,2})$ where nodes $c_{a,1}$ and $c_{a,2}$ are concepts (respectively the "subject" and "object" of the fact) and edge r is a relation. Embeddings for concepts and relations are derived using TransE (Bordes *et al.*, 2013) and fixed in order to allow the model to generalize to new facts. They are packed into a fact matrix $\mathbf{F}$ where each column $\mathbf{a}_i = c_{a_i,1} \oplus r_{a_1} \oplus c_{a_i,2}$ where $\oplus$ is the vector concatenation operator.

As shown in Figure 2.5, the Neural Knowledge Language Model (NKLM) uses a regular LSTM architecture and uses the state vector $\mathbf{h}_t$ at time-step t to determine which fact to select from $\mathbf{F}$ using an MLP implementing the $f_{\text{factkey}}(\mathbf{h}_t, \mathbf{e})$ function where $\mathbf{e}$ is a mean-pooling of $\mathbf{F}$. This returns a fact index a_t which can be used to retrieve the most relevant fact embedding $\mathbf{a}_t$. This fact embedding is then used, in conjunction with $\mathbf{h}_t$, to decide whether the next word should be generated from the vocabulary or the knowledge base using $f_{\text{copy}}(\mathbf{h}_t, \mathbf{a}_t)$, similarly implemented as an MLP. In either case, $\mathbf{h}_t$ and $\mathbf{a}_t$ are fed to an MLP implementing an index look-up similarly to f_{factkey}: f_{voca} and f_{poskey} for the vocabulary- and knowledge-based generation respectively.

f_{voca} straightforwardly yields a key to look-up the next word in the word embedding matrix (alternatively, this can be seen as predicting the embedding for

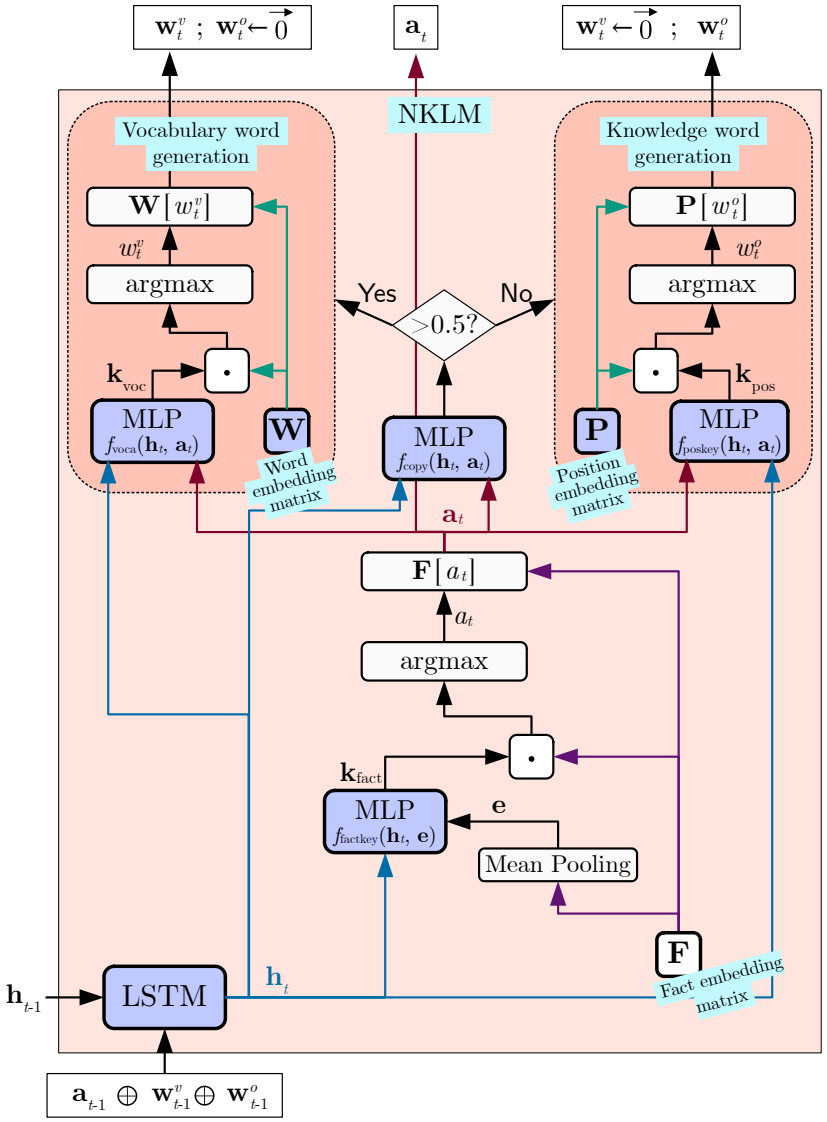


Figure 2.5: Simplified NKLM architecture proposed by Ahn *et al.*. The tunable parameters are marked purple. Arrows of a single color represent the same matrix or vector being used in multiple places.

the next word and retrieving the closest true word embedding), and f_{poskey} has the equivalent function in knowledge-based word indexing, using trained *position embeddings* which refer to words by their position in the description of $c_{a_t,2}$, the *object* of fact a_t. As knowledge-word position embeddings do not share the same vector space as regular word embeddings, both a word embedding and position embedding, in addition to the previously generated fact embedding $\mathbf{a}_t$, are fed to the LSTM at every time step t, with the embedding that was *not* generated at step $t-1$ initialized to the $\vec{0}$ vector.In their paper, Ahn *et al.* add a layer of probabilistic interpretation to the retrieval mechanism which yields $\mathbf{a}_t$, $\mathbf{w}_t^o$ and $\mathbf{w}_t^v$ by applying a $Softmax$ before argmax. We omit this from figure 2.5 for simplicity as $Softmax$ has no effect on the output of argmax.

This approach, by using position embeddings, treats the KB as an external resource which can be queried. It has the advantage of allowing the model to generalize to new facts and words added to the KB after training and, in a traditional word-based vocabulary, to predict out-of-vocabulary words. It suffers from many shortcomings, however. In particular, each word can only be associated with one fact. Ahn *et al.* present a limited situation in which the text is focused on one known topic τ which is the *subject* ($c_{a,1} \forall a \in \mathcal{F}_\tau$ with $\mathcal{F}_\tau$ the set of facts related to topic τ) and therefore cannot have multiple facts per word unless the knowledge graph has pairs of concepts that share more than one relation (that is, pairs of vertices that share more than one edge in a given orientation). In deployment, the ability to change topic focus mid-text or perhaps focus on multiple topics simultaneously is highly desirable. Furthermore, the indexing approach for knowledge-based token generation likely works because the KGs used have relatively few relation types and concept descriptions tend to be short and homogeneous. This approach is therefore unlikely to scale to larger, more complex KGs well. Lastly, this method heavily relies on string matching in order to construct supervised training samples and prompts. Depending on the knowledge base and topic, this may be unsatisfactory as, in addition to the potentially high computational cost of the approach, homographs may match incorrect concepts and synonyms or paraphrases may fail to match mentioned concepts. This problem is known as Entity Linking and current methods continue to struggle in all but the most trivial of cases (see section 2.6).

2.5.3.2 KBLSTM

In *Leveraging Knowledge Bases in LSTMs for Improving Machine Reading*, Yang and Mitchell (2017) propose the Knowledge-enhanced Bidirectional LSTM (KBLSTM), a concept-embedding-based method which relaxes the problem of Entity Linking by retrieving multiple *candidate* concepts per word from the KG. Specifically, candidate concepts which may be related to the current word are retrieved, and their embeddings are pooled with the output of the Bi-LSTM at each time step. The relevance to the current context is estimated using attention, in addition to a "sentinel" vector which allows the model to focus on the context and ignore the contextless background knowledge where relevant by deciding how much weight to assign to KG information versus the current state.

The KG is represented as a set of triples (c_1, r, c_2) with c_1, c_2 being any two concepts (nodes) with relationship (edge) r. Knowledge Graph Embeddings $\mathbf{v}_c$ for each concept c are derived following Yang *et al.* (2015).

As depicted in figure 2.6, the KBLSTM replaces the hidden state vector $\mathbf{h}_t$ with:

$$\hat{\mathbf{h}}_t = \mathbf{h}_t + \mathbf{m}_t$$

where $\mathbf{m}_t$ is a "knowledge state vector" which contextualizes knowledge vectors with respect to $\mathbf{h}_t$. It is computed as a mixture model, allowing for a better tradeoff between the impact of background knowledge and information from the context:

$$\mathbf{m}_t = \sum_{i \in V(x_t)} \alpha_{ti} \mathbf{v}_i + \beta_t \mathbf{s}_t$$

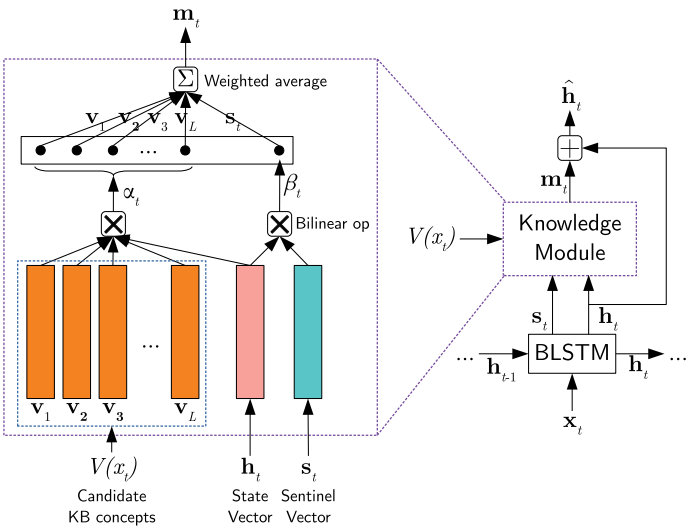


Figure 2.6: KBLSTM structure at each time step. Reproduced from Yang and Mitchell (2017).

Where $V(x_t)$ is the set of candidate concepts from the KB matching x_t, α_{ti} acts as an attention weight which reflects the relevance of the concept i, and β_t is the weighting term for the local context. Together, α_{ti} and β_t form a probability distribution and are defined as:

$$\alpha_{ti} \propto exp(\mathbf{v}_i^\top \mathbf{W}_v \mathbf{h}_t)$$
$$\beta_t \propto exp(\mathbf{s}_t^\top \mathbf{W}_s \mathbf{h}_t)$$
$$\sum_{i \in V(x_t)} \alpha_{ti} + \beta_t = 1$$

$\mathbf{s}_t$ is the aforementioned sentinel vector which encodes the context information and determines the knowledge/context tradeoff, defined as[2]:

$$\mathbf{b}_t = \sigma(\mathbf{W}_b \mathbf{h}_{t-1} + \mathbf{U}_b \mathbf{x}_t)$$
$$\mathbf{s}_t = \mathbf{b}_t \odot tanh(\mathbf{c}_t)$$

The word embeddings $\mathbf{x}_t$ are pretrained following Wieting *et al.* (2016) and fine-tuned during training. The KBLSTM outperformed BiLSTM in *entity extraction* on the ACE2005 and OntoNotes 5.0 corpora and in *event extraction* on ACE2005.

This article marks the first significant usage of a KG to improve a language model at the embedding level. This approach elegantly matches entity mentions to KB concepts and learns to combine the information from both modalities. It does, however, have several weaknesses. First, as $\mathbf{m}_t$ is not explicitly part of the $\mathbf{h}_t$ space, the responsibility for alignment of vector spaces falls on the optimization of the entire model for each task. It is generally preferable to learn projections between vector spaces explicitly, as it would make this method more easily applicable to fine-tuning pretrained language models and incentivize parameter specialization in the model. Furthermore, the candidate concept generation process does not work well for multi-word concepts ; while the approach to Entity Linking described has the advantage of being fuzzy thus allowing the model to learn which candidates are likely to be relevant, the single-word search is likely to produce low-precision results making learning difficult, and the lack of context in the search is likely to cause relatively low recall on some KBs. Despite these shortcomings, the approach has been influential, inspiring approaches such as KnowBert (Peters *et al.*, 2019), KAGNET (Lin *et al.*, 2019) and SenticLSTM (Ma *et al.*, 2018).

2.5.4 . In Transformer LMs

In the interest of reducing the pre-training burden of Transformer-based LMs such as BERT and expanding the range of concepts and relationships that they can accurately predict without resorting to additional freeform text, multiple methods for incorporating knowledge in contextualized word embeddings have been developed. One of the main categories of approaches is to rely on the Transformer's attention mechanism to combine entity and word information, as do ERNIE (Zhang *et al.*, 2019), K-Adapters (Wang *et al.*, 2020) and KnowBert (Peters *et al.*, 2019). K-Adapters in particular are compelling as they use Adapters (see Section 2.4.5) which, by construction, prevent catastrophic forgetting. Another common type of approach is to train a neural network or the language model itself to align entity representations with token representations, whether they be the input WordPiece representations such as E-BERT (Poerner *et al.*, 2020) or contextualized output representations such as KEPLER (Wang *et al.*, 2021b) and CODER (Yuan *et al.*, 2022). UmlsBERT (Michalopoulos *et al.*, 2021), in contrast, does not fit into the aforementioned categories as it mainly consists of biasing the BERT input vectors for entity mentions with a topic vector, and changing the Cross-Entropy loss in the Masked Language Modeling (MLM) objective to a Binary Cross Entropy Loss, setting all synonyms of a medical term as valid targets.

[2]See equation 2.1 for definition of $\mathbf{c}_t$

More specifically, E-BERT learns a projection of entity embeddings derived from the KB to the input word embedding space which preserves KG relation information, enabling their use in input text and providing the model with more information. UmlsBERT, on the other hand, explicitly adds semantic group embeddings to words found in UMLS. ERNIE and BERT-MK (He *et al.*, 2020) learn a fused representation of contextualized word and entity representations. K-Adapters (Wang *et al.*, 2020) are a computationally cheaper alternative to the ERNIE process which avoid the problem of catastrophic forgetting by using Adapters (Houlsby *et al.*, 2019), *i.e.* adding layers within the transformer which can be enabled or disabled and take on the burden of fine-tuning while the rest of the model weights remain frozen. While these methods have largely been successful, one drawback which we discuss in section 2.6 is that they all require a separate upstream Entity Linking (EL) step to be made at inference, limiting their performance and scope of applicability.

Some approaches do not suffer from the need for an EL step. KEPLER (Wang *et al.*, 2021b) is one such model, which introduces a knowledge embedding loss as an objective for language model pre-training, aligning contextual word representations with entity description representations. As such, it does not require an EL step at inference. The KnowBert model developed by Peters *et al.* (2019), on the other hand, grafts a KB-specific EL module into a transformer-based pretrained LM such as BERT, in order to jointly perform EL and contextualized word representation enrichment, making it also a standalone method requiring no upstream EL with the additional benefit of explicitly identifying the entities present in the text.

This distinction between methods which do or do not require an upstream EL step to be performed, that is, whether they require entities mentioned in the input text to be precisely identified, is an important one. This property is in fact one of the most important criteria for selecting a knowledge integration method in the biomedical case, and with UMLS in particular, as EL is currently an unsolved problem. On the MedMentions corpus (Mohan and Li, 2019), for instance, the best performing models are MedCat (Kraljevic *et al.*, 2021) and Bhowmik *et al.*'s dual encoder (Bhowmik *et al.*, 2021) which achieve 0.448 and 0.534 F_1 score respectively. The Entity Linker used by UmlsBERT is cTAKES (Savova *et al.*, 2010), which reaches an F_1 score of 0.178 on MedMentions, as reported by Kraljevic *et al.*. The majority of the aforementioned knowledge integration approaches are subject to this limitation. Whilst KEPLER and CODER are exceptions which do not require an upstream EL step at inference, CODER does require this in training, and KEPLER is trained on descriptions of each entity in the knowledge base, which are not available for some KBs such as UMLS. KnowBert relaxes the EL requirement, calling only for candidate entity mentions, and is trained both on Language Modeling and Entity Linking. KnowBert is thus less limited by the EL performance than other knowledge-based models, and does not require as much in-domain text as pre-training-based approaches. It therefore has considerable potential to effectively utilize the KB and is unlikely to suffer from CF on general text.

Several models are worth reviewing in greater detail due to their influence on the field as well as my work.

ERNIE (Zhang *et al.*, 2019) (not to be confused with the identically named model by Sun *et al.* (2019) which uses knowledge to simply preferentially mask entities in the MLM objective) is one of the first, simplest and most influential approaches, having inspired approaches such as BERT-MK (He *et al.*, 2020), GREASELM (Zhang *et al.*, 2022) and KnowBert (Peters *et al.*, 2019). It is based on the BERT model, but substitutes some of the basic transformer encoder blocks with knowledge-enhanced encoders called "K-Encoders" (see Fig. 2.7). K-Encoders accept additional inputs and outputs in the form of entity embeddings corresponding to entities mentioned in the sequence. The entity embeddings are fed through a multi-head attention mechanism which is identical to but separate from the multi-head attention applied to tokens. These recontextualized entity embeddings $\tilde{\mathbf{e}}_k$ are then paired with their corresponding recontextualized token embeddings $\tilde{\mathbf{w}}_j$ and fed to a token-entity information fusion mechanism which projects the embeddings into a unified space, performs their sum, and projects the fused representation back to token and entity space resulting in text-enriched entity representations $\mathbf{e}_k$ and knowledge enriched token representations $\mathbf{w_j}$ (see Eq. 2.8).

$$\mathbf{h}_j = \sigma(\widetilde{\mathbf{W}}_t\tilde{\mathbf{w}}_j + \widetilde{\mathbf{W}}_e\tilde{\mathbf{e}}_k + \mathbf{b}) \tag{2.8}$$

$$\tilde{\mathbf{w}}_j = \sigma(\mathbf{W}_t\mathbf{h}_j + \mathbf{b}_t) \tag{2.9}$$

$$\mathbf{e}_k = \sigma(\mathbf{W}_e\mathbf{h}_j + \mathbf{b}_e) \tag{2.10}$$

Hou *et al.* (2022) introduce Graph Convolution Simulation as a way to probe LMs for knowledge integration. They find ERNIE and K-Adapter (Wang *et al.*, 2020) successfully integrate only a marginal amount of knowledge.

GREASELM (Zhang *et al.*, 2022) is a similar approach which, rather than relying on pre-existing entity embeddings, extracts a relevant subgraph of the KG given the input text, and substitutes an attention-enabled GNN (Veličković *et al.*, 2018) for ERNIE's entity recontextualization step. Subsequently, rather than performing information fusion on matched token and entity embeddings, a sequence embedding similar to the `[CLS]` token in BERT[3] and an equivalent "node interaction embedding" for the graph are used to perform the information fusion indirectly. This side-steps the difficulty of matching multi-word entity names to multiple input tokens, promotes information mobility, and is computationally more efficient at inference. Both ERNIE and GREASELM suffer from the requirement of an upstream Entity Linking step, however GREASELM's constraint is weaker than ERNIE's due to the lack of necessity to match token and entity embeddings. DRAGON (Yasunaga *et al.*, 2022) significantly improves on GREASELM by introducing a graph completion self-supervised pre-training objective which masks relations in the knowledge sub-graph, similarly to the standard MLM objective.

[3] see Section 2.3

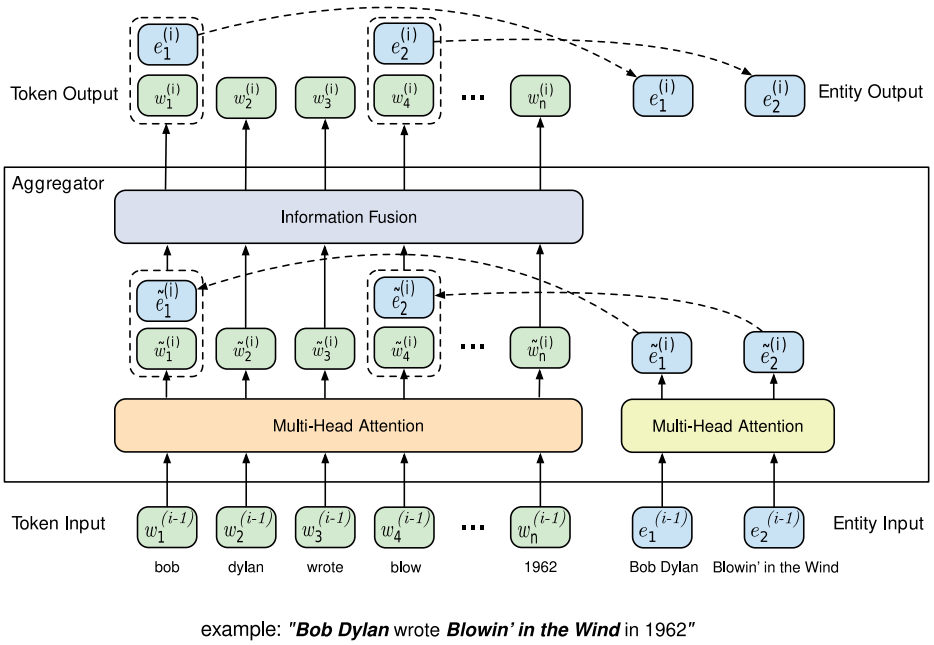


Figure 2.7: ERNIE's K-Encoder structure. Reproduced from Zhang *et al.* (2019).

UmlsBERT (Michalopoulos *et al.*, 2021) is, as its name suggests, a BERT-based language model which integrates knowledge from UMLS. It uses two knowledge integration mechanisms in tandem: knowledge augmentation of input embeddings, and a knowledge-informed multi-label MLM objective. The first mechanism consists of classifying each clinical input token into one of UMLS' "semantic types"[4] (groups of semantically related concepts) and adding to the token, segment and position embeddings[3] the relevant semantic type embedding. This effectively stretches input embeddings of the same semantic type in the same direction in the input space, giving BERT information on the relatedness of the tokens, which is particularly useful with uncommon words. The second mechanism uses UMLS' thesaurus feature, which associates clinical concepts to a list of names. Rather than using BERT's cross-entropy loss in its MLM objective to generate a probability distribution on the vocabulary, UmlsBERT uses binary cross-entropy — which gives an independent probability for each token to be an acceptable completion — and sets every clinical synonym as defined by UMLS to be an acceptable completion. For instance, for the masked word "lung", "pulmonary" would also be a positive label. This leads the model to learn to associate clinically related words, but should degrade the model's grammatical accuracy. It is not made explicit how the semantic type embeddings are computed, nor how the multi-label objective interacts with WordPiece tokenization and multi-word entity mentions. This model obviously suffers from reliance on accurate Entity Linking, but also uses Clinical-

[4]There are 127 semantic types in UMLS. UmlsBERT uses 44, but it is not made explicit which semantic types were chosen or why.

Bert as a pretrained backbone. Having been trained on approximately six times more biomedical text than BERT, accurately quantifying the added value of its knowledge integration is not straightforward.

2.6 . Entity Linking

Entity Linking (EL) is a task which consists of identifying spans of input text which refer to specific entities or concepts, and pairing them with the appropriate target they refer to in a knowledge base. Take, for example, the following sentence:

> *The council of the EU intends to force the operators of secure messaging apps such as WhatsApp or Signal to allow intelligence services to gain access to encrypted conversations through backdoors.*

Assume we use Wikipedia as our KB. We thus wish to identify any entity which has a Wikipedia page, and a unique identifier from which we can look up the page in question (for our purposes, we will use the page URL) for each mentioned entity. The entities in question for the aforementioned example are *the council of the EU* (1), *the European Union* (2), *WhatsApp* (3), *Signal* (4), *Intelligence Agencies* (5), and *Backdoor* (6). As a result of the Entity Linking process, we would like the occurrences of the aforementioned entities to be identified and labeled. We illustrate this in the following sentence by enclosing entity mentions in boxes and adding corresponding numbers (1-6) as subscripts.

> *The* $\boxed{\textit{council of the } \boxed{\textit{EU}}_{2}}_{1}$ *intends to force the operators of secure messaging apps such as* $\boxed{\textit{WhatsApp}}_{3}$ *or* $\boxed{\textit{Signal}}_{4}$ *to allow* $\boxed{\textit{intelligence services}}_{5}$ *to gain access to encrypted conversations through* $\boxed{\textit{backdoors}}_{6}$.

Each of the labels (1-6) should be matched to a URL as per Table 2.1.

Label	URL
1	https://en.wikipedia.org/wiki/Council_of_the_European_Union
2	https://en.wikipedia.org/wiki/European_Union
3	https://en.wikipedia.org/wiki/WhatsApp
4	https://en.wikipedia.org/wiki/Signal_(software)
5	https://en.wikipedia.org/wiki/List_of_intelligence_agencies
6	https://en.wikipedia.org/wiki/Backdoor_(computing)

Table 2.1: Correspondence table between labels in the example sentence and corresponding entity URLs.

In summary, in our definition[5], Entity Linking consists of two separate tasks: Mention Detection (MD; a KB-specific NER task which does not require an entity type) and Entity Disambiguation (ED; also referred to as Entity Normalization). The additional task of Candidate Generation (CG), *i.e.* a pre-selection of potentially relevant concepts, is often inserted between MD and ED. Crucially, an Entity Linking task is accompanied by a corresponding KB which defines which entities must be marked. For all practical purposes, regardless of the KB, we may assume that each entity is associated with at least one plain text descriptive "name". We may not, however, assume that entities systematically have a textual description. Because EL is dependent on a specific KB, it is difficult to systematically compare performance across methods. We will therefore go over the main approaches in general terms, before focusing on specific contributions for the biomedical domain, contextualizing the work described in subsequent chapters of this thesis.

The naïve approach to EL consists of matching every substring of the input text to every name for every entity in the KB using a string similarity measure such as Levenshtein distance or Jaccard similarity and selecting a threshold above which the span is linked to the relevant entity. This is the approach followed, for instance, by QuickUMLS (Soldaini and Goharian, 2016). RysannMD (Cuzzola *et al.*, 2017) and Rao *et al.* (2013) use string matching as a basis for candidate selection before applying more sophisticated final selection and disambiguation procedures. Several approaches (Han *et al.*, 2011; Hoffart *et al.*, 2011a; Alhelbawy and Gaizauskas, 2014) propose to generate candidate matches in this way, extract the subgraph of mentioned entities from the KG, and differ in their method for selecting only the most highly related nodes. While many of these approaches show fairly compelling results, this exhaustive approach works only for small amounts of text and small KBs, and quickly becomes intractable as we discuss in Chapter 4. Additionally, the use of string similarity as a basis for selecting candidate mentions yields poor results for many KBs which do not provide sufficiently many different names for their entities. Using our previous example, it is unlikely that an occurrence of the word "EU" could be accurately matched to the page of the "European Union" using string similarity.

Kolitsas *et al.* (2018) use an ELMo-like BiLSTM language model to represent their input text as contextualized word embeddings, which they pool into span embeddings for all sub-sequences of the input text exhaustively. They then retrieve a number of candidate entity embeddings from a pretrained probabilistic map provided by Ganea and Hofmann (2017) for each span embedding. They then measure the similarity between the span embeddings projected in the entity-embedding space and the candidate embeddings and use this similarity score as a basis for their EL decision. This approach is essentially a vectorial version of the aforementioned approaches, and as such benefits from better hardware acceleration, but shares the same fundamental computational complexity. Furthermore,

[5]Some definitions omit the MD sub-task. However, there is to my knowledge no better term for the coupling of both Mention Detection and Entity Disambiguation. We do not go over disambiguation-only approaches as their actual EL performance is heavily reliant on the accuracy of the upstream MD step.

the pretrained span-to-entity-embedding map must be created for every knowledge graph, which poses its own concerns in terms of tractability and requirement for annotated text.

Van Hulst *et al.* (2020) use ELMo-based contextualized word embeddings to power the Mention Detection step, modeled as a simple NER task. Candidate entities are chosen in the same way as Kolitsas *et al.* (2018) and Ganea and Hofmann (2017) and/or by cosine similarity between entity embeddings and the sum of the words in the context of the detected mention ("contextual similarity") (which assumes entity and word embeddings are in the same space). Entities are disambiguated according to a score computed as a function of prior probability, contextual similarity, and "coherence" (Ganea and Hofmann, 2017; Le and Titov, 2018) (a measure of relatedness between different entity candidates for the span). The lower complexity of the MD step and the fewer candidates selected compared to previous methods improve the complexity of this approach, but it is specific to the Wikipedia KB and heavily relies on the work of Ganea and Hofmann (2017), making it difficult to adapt to other KBs.

The obvious way to avoid the need for a span-to-entity-embedding map such as the one created by Ganea and Hofmann (2017) is to embed entities and tokens in the same space. Moreno *et al.* (2017), for instance, propose an extension of Word2Vec which applies to corpora containing entity mentions, as well as several similarity measures to perform ED. These approaches rely on the existence of such annotated corpora however, which are not available for most knowledge bases.

Ravi *et al.* (2021) uses two Transformer-based LMs in sequence. The first LM performs MD as a typical NER task, feeding the resulting spans to a wikipedia-specific search engine which incorporates a lookup table for related terms and a page ranking algorithm. The search engine returns candidate entities, and the second LM performs ED by modeling it as a sequence classification task: it is fed the detected mention, the context sequence, and a description of the candidate entity, and must predict whether the candidate entity matches the mention. While the approach is efficient and performs well on Wikipedia, the reliance on its search engine and assumption that entities are accompanied by descriptions makes it inapplicable to most other KBs. Additionally, it requires an annotated corpus to train the ED task, which is rarely available.

E-BERT (Poerner *et al.*, 2020), which is originally a Knowledge-Integration approach and hypothetically requires an upstream EL step, proposes an ED approach which leverages its concept-embedding-to-word-embedding projection such that it can perform its own EL (assuming the MD and CG steps are already handled). While E-BERT normally expects concept embeddings to be projected to the WordPiece input space and fed as input simultaneously with regular input WordPiece embeddings (all the while assuming that the relevant knowledge is not lost in the projection process), Poerner *et al.* propose to repurpose the mechanism in reverse. Specifically, they suggest to find concepts by replacing their embeddings in the input with an average of the candidate concepts' embeddings, and training the model to output the embedding for the correct concept. While at first glance this approach may seem like it relies on the language model's knowledge

thus defeating the purpose of knowledge integration, the upstream CG process and the gap between the true and predicted concept embedding contribute knowledge that the model does not necessarily possess. This approach unfortunately relies on MD and CG tools as well as large amounts of text with annotated KB concepts, reducing its scope of applicability to the Wikipedia KB, and few if any others.

End-to-end neural Entity Linking had historically not been heavily considered until the introduction of the Transformer. This approach has since been attempted by Broscheit (2020) and Chen *et al.* (2020), which we will go over in Chapter 3. Transformers have also enabled approaches for matching entity mentions to entity embeddings in a shared vector space (Humeau *et al.*, 2019; Logeswaran *et al.*, 2019; Wu *et al.*, 2020), assuming that textual descriptions of mentions exist. This assumption is not verified for all KBs, and notably not for UMLS; furthermore, these approaches tend to be computationally expensive due to the necessity of mining hard negative samples, though Wiatrak *et al.* (2022) have introduced a loss regularization which helps relax this constraint.

While Entity Linkers are known not to be perfectly reliable, Biomedical Entity Linking on the UMLS KB is a particularly difficult task. One of the most widely used biomedical EL systems is cTAKES (Savova *et al.*, 2010), which reaches an F_1score of 0.178 on MedMentions, as reported by Kraljevic *et al.* Some of the best performing models include RysannMD (Cuzzola *et al.*, 2017), which achieves 0.436 F_1 on the CRAFT corpus, and the dual encoder architecture proposed by Bhowmik *et al.* (2021) which achieves 0.534 F_1 score (with 0.529 Precision and 0.538 Recall) on the MedMentions corpus (Mohan and Li, 2019). In the latter case, this translates to slightly more than half of the actual entities mentioned being correctly identified, and nearly half of the entities found by the linker being false positives.

3 - Biomedical Entity-Linking with Transformers

3.1 . Introduction

Whilst knowledge integration has been fairly well explored with projects such as NKLM (Ahn *et al.*, 2016), ERNIE (Zhang *et al.*, 2019), or UmlsBERT (Michalopoulos *et al.*, 2021), most of the proposed methods assume knowledge-base entities mentioned in the text are known. Requiring human annotation of these concepts defeats the purpose of automation, meaning entity mentions must be automatically identified and labeled upstream of knowledge integration. This task is known as Entity Linking (EL) and, depending on the concepts present in the knowledge base, is typically a difficult task (see Section 2.6).

An obvious approach to EL is to attempt to use a Transformer-encoder language model such as BERT to perform EL by modeling it as a fine-grained Named Entity Recognition (NER) task. That is, given a KB with N concepts, each token in a sequence is classified (based on its output contextualized representation) as one of the N concepts — or as not a mention of any concept. To handle multi-token concept names, the IOB2 scheme is typically applied, with each token being classified as Inside (`I`) a concept mention, Outside (`O`) of any concept mention (not-a-KB-concept), or the Beginning (`B`) of a concept mention[1]. Accounting for the `B` and `I` tags which can apply to any of the N concepts, the resulting classification has $2N + 1$ classes. See Table 3.1 for an example of IOB2 EL tagging of a sequence. This approach has the advantage of being easier to put in place than those mentioned in Section 2.6 and, as it does not contain any string-search or other major CPU-bound operations, can be relatively fast at inference with hardware acceleration, but is limited by a requirement for labeled training data on all concepts and is expensive to train.

INPUT	Pseudomonas	aeruginosa	(Pa)	infection	in
OUTPUT	B-C0854135	I-C0854135	I-C0854135	I-C0854135	O
INPUT	cystic	fibrosis	(CF)	patients	is
OUTPUT	B-C0010674	I-C0010674	B-C0010674	O	O

Table 3.1: Space-tokenized input sequence and its IOB2 tagging with UMLS concepts as output sequence. Each tag is either O for tokens *Outside* any entity mention, or of the form `[B/I]-[UMLS Concept Unique Identifier (CUID)]`.

[1]This annotation scheme assumes entity mentions are continuous. Handling discontinuous entities requires complexification of the annotation scheme which can be counterproductive. This is a topic of ongoing research (Dirkson *et al.*, 2021).

Another approach, as the classification process produces a sequence and concept mentions follow a Zipfian distribution much like a vocabulary, is to treat the problem as a sequence-to-sequence task. Given the variety of concepts, each label can be considered an entry in the model's output vocabulary, and the task can be tackled similarly to a translation task for which the target output sequence is the IOB2 "translation" of the input sequence. This is the approach followed by Boguslav *et al.* (2021).

To evaluate subsequence classification, we use SeqEval (Nakayama, 2018), which assigns true positives only to perfectly predicted subsequences. Any predicted sequence which does not exactly match a reference sequence from the dataset is considered a false positive, and any reference sequence which does not have a matching prediction is considered a false negative. See Table 3.2 for an example. As SeqEval criteria are stringent, numerically low performance can conceal a large number of near-correct predictions, therefore we also use token-level F_1 (which does not take sequence into account) as an evaluation metric.

Sentence	in	cystic	fibrosis	(CF)	patients	is
Expected: 2 TP	O	B-C0010674	I-C0010674	B-C0010674	O	O
FP & 2 FN	O	B-C0010674	I-C0010674	I-C0010674	O	O
TP & FN	O	B-C0010674	I-C0010674	O	O	O
TP & FP & FN	B-C0010674	I-C0010674	I-C0010674	B-C0010674	O	O

Table 3.2: Examples of various subsequence classification errors according to the SeqEval criteria.

3.2 . Training from scratch in a resource-constrained setting

While we originally considered fine-tuning a pretrained model such as BERT, lack of computational power required for fine-tuning BERT-sized language models led us to attempt to train a smaller model from scratch. Based on our computational budget, we set the size of our models to vary circa 8 million parameters (approximately on fourteenth that of $\text{BERT}_{\text{BASE}}$). Expecting poor performance on the MLM task with such a small language model, and expecting full semantic NLU to be unnecessary (as, in order to compete with our baseline string-matching-based approaches such as QuickUMLS (Soldaini and Goharian, 2016), we were expecting token-based fuzzy pattern-matching and some basic intuitions about parts of speech to be sufficient), we initially did not pretrain our models (either on MLM or autoregressively). We attempted the two main aforementioned approaches: the token embedding classification approach, and a sequence-to-sequence approach based on Vaswani *et al.*'s translation model.

Faced with initial difficulties with models performing marginally better than chance, we created a training curriculum: models were to first be trained on Men-

tion Detection (MD) (*i.e,* the IOB sequence classification task with no entity differentiation required), then on typed NER using UMLS' 127 semantic types (to which each of the concepts in UMLS belong) as entity types, and finally on the Entity Linking task. Word-based, wordpiece-based and character-based tokenization were all attempted, with wordpiece-based tokenization outperforming the alternatives.

Despite Boguslav *et al.*'s subsequent success with the translation-based approach, likely due to their larger pretrained model, experiments with the sequence-to-sequence model were halted early due to it being more computationally expensive and performing worse than its non-generative counterpart. As shown in Table 3.3, our best performing model performed only marginally better than chance at the MD task. With poorer performance on this simpler task than state of the art entity linkers on the EL task, it became clear that the architecture was not large enough and/or the training data not plentiful enough to learn EL in this manner.

Model	Mention-F_1	Token-F_1
Random predictor	0.00	0.33
Ours	0.01	0.47

Table 3.3: Mention-based (subsequence-sensitive) and Token-based (non-subsequence-sensitive) F_1score on Mention Detection for a non-pretrained 8 million-parameter Transformer (encoders only).

3.3 . Entity Linking with BioBERT

After having acquired additional computing resources, we attempted to fine-tune an existing pretrained biomedical language model, BioBERT, to each of the tasks in the aforementioned curriculum (independently, as following the curriculum led to worse results, presumably due to catastrophically forgetting language modeling when learning MD). While MD performance is difficult to evaluate as there exists no baseline for the MedMentions corpus, BioBERT's MD performance exceeded the EL performance of the state of the art model (Bhowmik *et al.*, 2021), as depicted in Table 3.4; failure cases mostly involved abbreviations (*e.g.* "(Pa)" and "(CF)" in the example sequence in Table 3.1).

The Semantic Type NER task appears not to be significantly more challenging than MD for BioBERT. However, increasing the number of classes from 255 (owing to the 127 semantic types tagged in IOB2 scheme) to over 16,000 (with approximately 8,000 concepts in the corpus) revealed that BioBERT's ability to discriminate between UMLS concepts was low. Approximately 94.6% of unique concepts which appear in the MedMentions test corpus were not found as they appeared too rarely in the training data. When taking into account only the twenty most common concepts, the micro-F_1score of BioBERT is 0.42.

Task	Precision	Recall	Micro-F_1
SOTA Entity Linking	0.53	0.54	0.53
TaggerOne	0.47	0.44	0.45
Mention Detection	0.66	0.71	0.68
Semantic Type NER	0.60	0.63	0.60
Entity Linking	0.22	0.21	0.21

Table 3.4: BioBERT performance on MD, NER (semantic types) and EL on MedMentions, with State Of The Art (SOTA) entity linker (Bhowmik *et al.*, 2021) and TaggerOne (Leaman and Lu, 2016, referenced in the original MedMentions paper by Mohan and Li) as baselines for comparison.

3.4 . Subsequent literature

Wiatrak and Iso-Sipila (2020) follow a similar approach to ours, using hierarchical multi-task training to learn MD, entity typing, and ultimately Entity Linking in an end-to-end fashion. Their approach differs from ours in several meaningful ways. First, they use SciBERT (Beltagy *et al.*, 2019) as a pretrained backbone. Second, the training curriculum follows Sanh *et al.* (2019) and alternates tasks at random, with the probability for a given task to be chosen next being proportional to its relative dataset size. Third, bidirectional LSTMs are inserted between (1) the output of SciBERT and the MD layer, and (2) between the aforementioned LSTM and the entity typing and EL layers. Skip-connections bypassing the LSTMs are added so as not to create an information bottleneck. This approach seems to successfully build on its understanding of MD and entity typing to achieve considerable performance in EL, all the while largely overcoming the catastrophic forgetting phenomenon, likely thanks to the different training curriculum. However, it falls short of the state of the art, with approximately 0.44 F_1.

While our work has focused on UMLS, similar approaches have been attempted on Wikipedia concurrently with and subsequently to our work, and likely due to the greater abundance of training data, have been met with a greater degree of success.

In *Investigating Entity Knowledge in BERT with Simple Neural End-To-End Entity Linking*, Broscheit (2020) attempts to use BERT to perform EL by modeling it as a fine-grained NER task, much like the approach followed with BioBERT. The KB is the top 700 000 Wikipedia entities with the most other Wikipedia pages linking to them, and rather than being annotated using IOB2, it is assumed that two contiguous mentions of a concept never occur. Therefore the classifier has 700 001 classes. This approach yields worse results than the alternatives mentioned in Section 2.6, but is much faster at inference. Chen *et al.* (2020) employ a similar approach, but train a MD classifier and an ED classifier separately, which achieves somewhat better results. They also find adding candidate sets following Hoffart *et al.* (2011b) to limit the search space for ED yields significantly better results.

3.5 . Conclusions

While the Transformer-based end-to-end neural approach to EL does work in some circumstances, it encounters issues when scaling and requires large amounts of annotated data which are not available for most KBs. It cannot handle rare concepts, especially those that are not part of the model's training data, limiting its potential for applications, especially considering the fact that rare concepts are often the ones for which it is most useful to retrieve additional structured knowledge, since statistical information is scarce during language model training.

Approaches to Entity Linking which perform reasonably well across the entire range of concepts are largely based on string matching and are thus computationally expensive. Transformer-based EL is cheaper to run at scale and has better grammatical awareness, but is limited to common concepts or makes strong assumptions about the KB (*e.g.* all concepts must have a text description). Finding a compromise which combines traditional EL approaches with the grammatical proficiency, contextual awareness and inference speed of transformers may be more effective than either approach in isolation. By decomposing the EL task into MD, CG and ED, we may be able to leverage the LM's grammatical ability for MD and its understanding of context for ED, whilst CG could be performed as a classical EL approach. By increasing the focus on recall compared to precision and thanks to the LM's high performance on MD (thus necessitating few mention spans to be checked), it may be possible to make CG tractable.

Lastly, the effectiveness of knowledge integration being inherently limited by the prior Entity Linking step, the possibility of performing knowledge enrichment without Entity Linking becomes attractive. Alternatively, learning both tasks jointly may allow EL (or ED) to benefit from knowledge integration and vice versa.

4 - KnowBert-UMLS

4.1 . Introduction

The most notable approach to performing knowledge enrichment jointly with Entity Linking is KnowBert (Peters *et al.*, 2019), an enhanced BERT model which integrates a mechanism for matching the embeddings of concepts in a Knowledge Base, as given by a graph embedding algorithm such as TransE (Bordes *et al.*, 2013) in the case of Knowledge Graphs, to the partially contextualized token representations yielded by the N^{th} Transformer encoder block of BERT. It is similar in essence to the KBLSTM described in Section 2.5.3.2, applied to the Transformer architecture. While this EL mechanism does require an initial candidate generation step, it enables the use of a powerful context- and grammar-aware language model to aid in selecting the most relevant candidate concept(s) as it performs knowledge integration similarly to ERNIE (see Section 2.5.4).

This approach has been shown by Peters *et al.* (2019) to be effective at increasing BERT's performance on general tasks when used with the WordNet general KG (Miller, 1995), which includes mainly semantic relations between words (*i.e.* synonymy, hypernymy, hyponymy, *etc.*) and the Wikipedia KB (using document embeddings derived from page contents). UMLS, on the other hand, is a domain-specific KB which maps a domain that is unfamiliar to BERT. Additionally, while it comprises fewer concepts than Wikipedia, UMLS records many more concept names[1] (as each concept is associated with various textual representations) and does not provide text descriptions for most concepts.

We therefore investigate whether KnowBert, as a highly effective knowledge integration technique which works around the greatest drawback of competing approaches (*i.e.* the need for an upstream EL step), is suitable for *domain adaptation* specifically, and whether it scales to a KG as large and complex as UMLS.

4.2 . Enriching contextualized representations with biomedical ontologies

Currently, biomedical document processing is mostly human work. Software solutions which attempt to alleviate this burden exist but generally do not perform well enough to be helpful in many applications. Concurrently, there exist projects which organize concepts in the biomedical field. Therefore, we seek to leverage existing structured knowledge resources to improve biomedical language modeling. In this section, we integrate the UMLS knowledgebase into a BERT-based language model, aiming to improve its performance in biomedical Named Entity Recognition. To achieve this, we extend KnowBert, a model architecture designed to integrate knowledge into language models. Preliminary results reveal

[1]There approximately 6.7 million Wikipedia articles versus 3.3 million concepts and 15 million concept names in UMLS as of the writing of this dissertation

the challenges of applying KnowBert to the biomedical domain given the number and subtlety of different concepts in UMLS. Going forward, addressing these challenges and combining this with other approaches such as BioBERT may help expand the range of usefully automatable biomedical language processing tasks.

With over a million articles published every year in the biomedical field and the large number of patient records generated by hospitals, it is increasingly difficult for healthcare professionals to keep up to date on research, carry out systematic reviews, or search for patient information. There is thus a demand for language processing tools able to identify and extract meaningful information from these texts.

For this reason, multiple knowledge bases such as the Unified Medical Language System (UMLS) and the OpenTargets LIterature coNcept Knowledge base (LINK) have been created to make information more searchable. Our objective is to enable Transformer-based pretrained neural Language Models to make explicit use of this knowledge, in order to improve their performance, interpretability, and data efficiency.

Other projects such as BioBERT (Lee *et al.*, 2019) and ClinicalBert (Alsentzer *et al.*, 2019) have successfully specialized language models to the biomedical domain. However, their approach has typically not explicitly leveraged structured knowledge sources such as UMLS. A notable exception is UmlsBERT (Michalopoulos *et al.*, 2021), which leverages UMLS as a thesaurus to explicitly teach BERT synonymy and enriches biomedical word representations with semantic type embeddings (see Section 2.5.4 for a more in-depth explanation).

We follow the method described by Peters *et al.* (2019) known as KnowBert to integrate knowledge derived from the UMLS Knowledge Base into a BERT-based language model. We thus call this model KnowBert-UMLS. Our approach using KnowBert differs significantly from previous specialized models in that it makes use of the full vocabulary of UMLS to enrich word representations, and jointly performs entity linking. It is also fairly indifferent to the pretrained LM used as a base, whereas UmlsBert is limited to biomedically pre-specialized models such as ClinicalBert (Alsentzer *et al.*, 2019).

In the context of large and specialized knowledge bases such as UMLS, we find the approach proposed by Peters *et al.* (2019) to be computationally unrealistic with current tools for most organizations. Preliminary biomedical Named Entity Recognition evaluations of our model trained on a small subset of our training corpus demonstrate a decrease in performance with respect to models with non-enriched word representations. We investigate the reasons for this, and propose ways to alleviate this computational burden.

The remainder of the section is organized as follows. In Subsection 4.2.1, we overview the architecture of KnowBert and discuss the specifics of our extension of it to the UMLS Knowledge Base. The preliminary experimental results are reported and discussed in Subsection 4.2.2. Finally, we present in Subsection 4.2.3 our conclusions and future work.

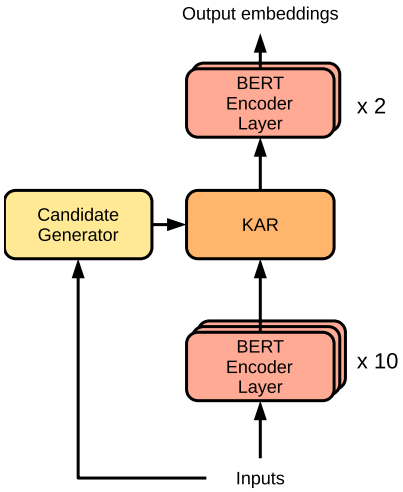


Figure 4.1: Abstraction of the KnowBert architecture. KnowBert extends BERT by adding a Knowledge Attention and Recontextualization module (KAR) between two transformer encoder layers; in this case between layers 10 and 11.

4.2.1 . KnowBert-UMLS

The core idea of KnowBert-UMLS, much like the KBLSTM introduced by Yang and Mitchell (2017), is to overcome the hurdle of Entity Linking which approaches such as ERNIE (Zhang *et al.*, 2019) and UmlsBERT suffer from by selecting multiple *candidate* concepts for each entity mentioned in the input text, increasing recall at the cost of precision, and learning a soft, probabilistic, and contextual entity linking in order to identify the most relevant candidates. The implementation of this idea diverges significantly from the KBLSTM, as the token-by-token functioning of the LSTM naturally lent itself to single-word entity mentions. In a Transformer-based language model, an additional layer of complexity is added in the form of *candidate spans*, *i.e.* pieces of the input text which may or may not contain an entity mention and each of which will have an associated set of candidate concepts.

As shown in Fig. 4.1, the KnowBert architecture is composed of three main components: a pretrained Language Model backbone, a Knowledge Attention and Recontextualization Module (or KAR) which performs Entity Linking and knowledge enrichment of word representations, and a candidate mention generator.

4.2.1.1 Pretrained BERT

While the KnowBert method can apply to most Transformer-based pretrained language models, we focus on BERT as it was the pretrained backbone used by Peters *et al.* (2019). BERT models comprise L Transformer encoder layers. For a sequence of N tokens, each layer i takes as input an $N \times H$-dimensional sequence representation $\mathbf{H}_{i-1}$ and outputs a representation $\mathbf{H}_i$ which integrates more contextual information by applying a multi-headed attention mechanism to $\mathbf{H}_{i-1}$ followed by a Multi Layer Perceptron. In the case of $\text{BERT}_{\text{BASE}}$, we have

$L = 12$ and $H = 768$. The final output of each token is thus a contextualized representation in $\mathbb{R}^H$.

4.2.1.2 *Ontology & candidate generator*

The KnowBert method ties a pretrained language model to a Knowledge Base, specifically an Ontology. For our purposes, we define a Knowledge Base $\mathcal{K}$ as a set of $J_\mathcal{K}$ entities e_j, each with a vectorial representation $\mathbf{e}_j \in \mathbb{R}^K$. We use the UMLS Knowledge Base, with each entity corresponding to a Concept Unique Identifier. The entity embeddings we use are computed according to the adversarial method provided by Maldonado *et al.* (2019) with $K = 50$.

To perform the Entity Linking step, the KAR requires a candidate generator to create a list $\mathcal{C}$ of candidate mentions. Specifically, a set $\mathcal{S}$ of S *candidate spans* is associated to each sequence, which may or may not contain an entity mention. Each candidate span is then assigned a corresponding list of *candidate entities*, including a null entity representing the lack of an entity mention within the candidate span. Formally, we have:

$$\mathcal{C} = \{(s, \{e_{s,1}, \ldots, e_{s,J_s}\}) | s \in \mathcal{S}\} \tag{4.1}$$

with each candidate span s being associated to a set of J_s candidate entities, and each entity e_j having a corresponding vector $\mathbf{e}_j \in \mathbb{R}^K$.

These candidates are produced by a candidate generator which follows rules specific to the KB being used. KnowBert as specified by Peters *et al.* (2019) implements compatibility with two KBs, namely WordNet and Wikipedia.

The primary challenge in crafting a mention generator for the biomedical domain, and specifically UMLS, is the variability of formulations for each concept. In the case of UMLS, each concept is associated to a list of strings (called "names") that may represent it. For instance, the concept for lung cancer is associated to 97 different forms, including "pulmonary carcinoma".

To leverage these names, we have attempted several methods based on string similarity and cosine similarity of vectorial word representations. We have found the most effective option for our purpose to be the QuickUMLS python library (Soldaini and Goharian, 2016) which, given some text, identifies candidates in the form `(span_start, span_end, concept_ID)`. We then aggregate candidate entities by candidate span, derive an empirical estimate of the prior probabilities for each entity from MedMentions, and find the relevant entity embeddings as described in Fig. 4.2. Finally, we feed the output of this candidate generation process to the KAR.

In practice, matching each of the approximately 180M (million) sequences in our training corpus to the 16M names in UMLS on-demand is prohibitively computationally expensive in training. In order to achieve this, we precompute the candidates for each of our sequences ahead of time and create a lookup table for each file in our corpus. This needs to be done only once and is parallelizable, but nonetheless 3.5% of our corpus took six days to process across seven nodes of a computing cluster, each equipped with two Xeon 36-thread processors with a

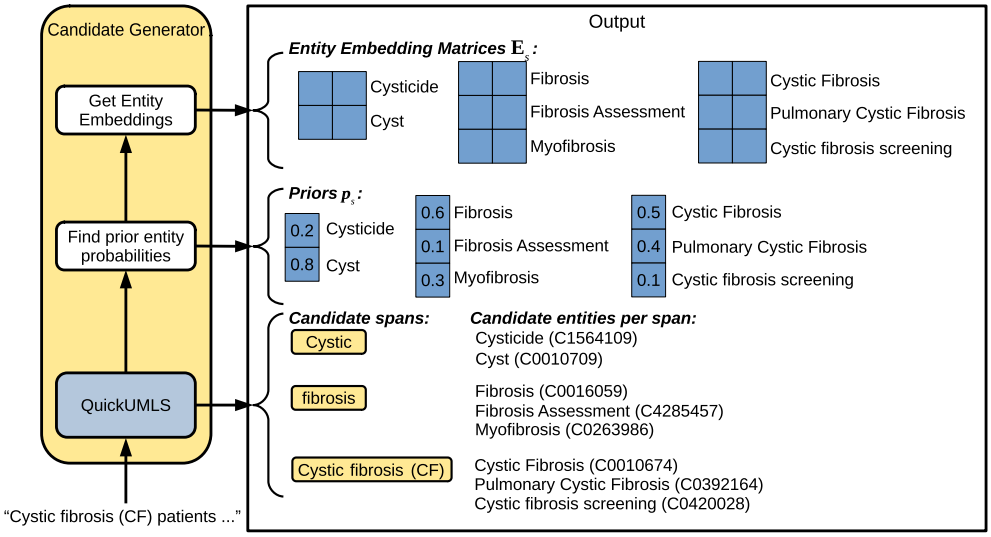


Figure 4.2: Detailed structure and output of the UMLS candidate generator.

clock speed of 3GHz, and required us to favor speed over recall and precision when considering QuickUMLS settings.

Depending on which similarity measure and threshold are chosen, QuickUMLS trades off between recall and execution time. We settled on Jaccard similarity, with a threshold of 0.7 as the best compromise we could find.

In our experience, the computational impact at inference is fairly low for on-demand low-volume applications, as the candidate generator typically takes fractions of a second to process a sequence.

4.2.1.3 KAR

The KnowBert approach adds a KB-specific "Knowledge Attention and Recontextualization module", or KAR, between two transformer layers in a pretrained BERT model. This module is a relatively inexpensive addition to the pretrained model, with in our case only approximately 0.3% as many trainable parameters as $\text{BERT}_{\text{BASE}}$.

Multiple KBs can be used in tandem: theoretically, a KAR can be inserted between every pair of layers in the transformer. In practice, the insertion of a KAR too close to the input layer causes too much perturbation to the flow of information and prevents the model from recovering during training. As suggested by Peters *et al.* (2019), in order to minimize the language model's perplexity[2], we insert the KAR between the tenth and eleventh layers of $\text{BERT}_{\text{BASE}}$ as per Fig. 4.1.

This module performs entity linking on the intermediate contextualized word representations and pools them with the relevant entity embeddings. This results in contextualized word representations which are enriched with information extracted

[2]Perplexity is computed as the exponential of the cross-entropy loss, and is a standard measure of how well the language model predicts samples.

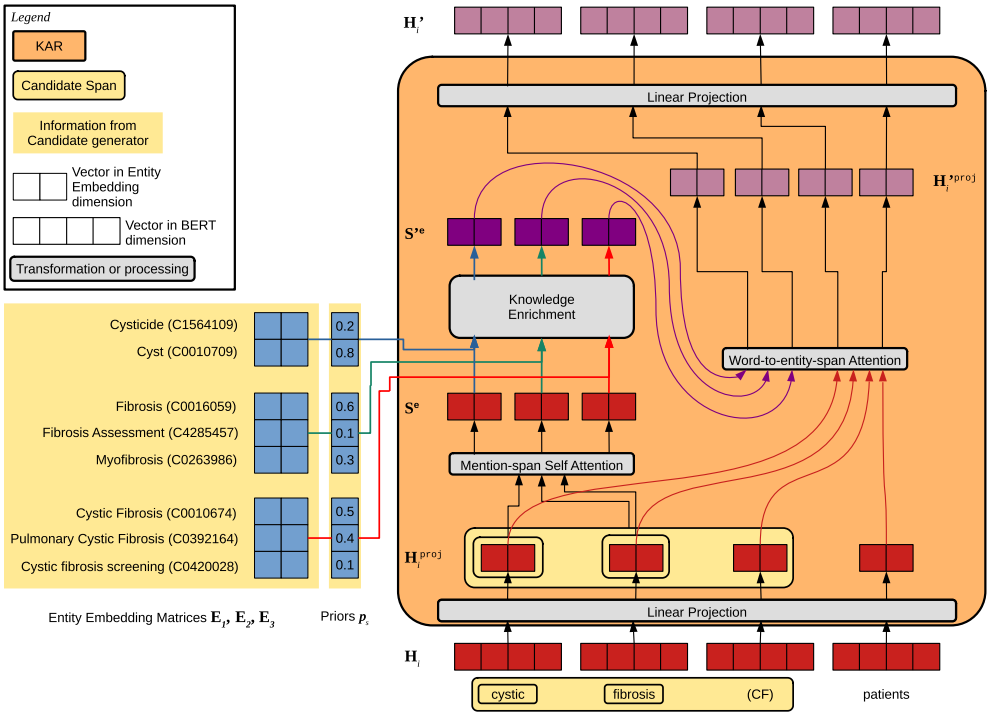


Figure 4.3: Detailed structure of the Knowledge Attention and Recontextualization module (KAR).

from a KB. Specifically, the KAR takes as input a sequence representation $\mathbf{H}_i$ and a list $\mathcal{C}$ of S candidate mentions as generated by the Candidate Generator (see (4.1)).

As described by Peters *et al.* (2019) and illustrated in Fig. 4.3, the KAR first linearly projects the output of the previous transformer encoder layer $\mathbf{H}_i$ to the entity embedding space:

$$\mathbf{H}_i^{proj} = \mathbf{H}_i \mathbf{W}^{proj} + \mathbf{b}^{proj} \tag{4.2}$$

where $\mathbf{W}^{proj}$ and $\mathbf{b}^{proj}$ are learned.

Then, the projected embeddings for the words in each span are pooled into a matrix $\mathbf{S} \in \mathbb{R}^{S \times K}$ of span embeddings. Each span embedding is computed following Lee *et al.* (2017), who describe a way to compute text span vectors: each token in each span is associated to a weight computed from the contextualized embeddings fed through a trained FFNN. These weights are softmaxed with respect to each span of text, and serve as the weights for a weighted-sum pooling of the non-contextualized token embeddings, resulting in non-contextualized text span embeddings.

These span embeddings are then contextualized with a standard transformer layer to allow the entity linker to identify relationships between entity mentions, resulting in the contextualized span embedding matrix $\mathbf{S^e}$.

$$\mathbf{S^e} = \mathrm{MLP}(\mathrm{MultiHeadAttn}(\mathbf{S}, \mathbf{S}, \mathbf{S})) \tag{4.3}$$

where MLP and $\mathrm{MultiHeadAttn}$ designate a position-wise Multi-Layer Perceptron (Vaswani *et al.*, 2017) and a Multi-Headed Attention layer respectively.

The contextualized span embedding $\mathbf{s^e}$ of every candidate span s is then used to pool the corresponding matrix of candidate entity embeddings $\mathbf{E}_s$ from the KB, resulting in a predicted entity representation:

$$\begin{aligned} \boldsymbol{\psi}_s &= \mathrm{Softmax}(\mathrm{MLP}(\mathbf{p}_s, \mathbf{s^e} \cdot \mathbf{E}_s)) \\ \tilde{\mathbf{e}}_s &= \boldsymbol{\psi}_s \cdot \mathbf{E}_s \end{aligned} \tag{4.4}$$

where $\mathbf{p}_s \in \mathbb{R}^{J_s}$ is the vector of prior probabilities for the candidate entities associated with span s, and $\boldsymbol{\psi}_s \in \mathbb{R}^{J_s}$ is an estimate of their posterior probabilities.

The predicted entity representation embeddings $\tilde{\mathbf{e}}_s$ of each span s are packed and added to contextualized span embeddings $\mathbf{S^e}$, forming the knowledge-enriched span embedding matrix $\mathbf{S'^e}$:

$$\mathbf{S'^e} = \mathbf{S^e} + \tilde{\mathbf{E}} \tag{4.5}$$

$\mathbf{H'}_i^{proj}$ is computed with word-to-enriched-entity-span attention, similarly to applying a regular transformer encoder layer to $\mathbf{S'^e}$ but substituting the query in the attention mechanism for projected word embeddings $\mathbf{H}_i^{proj}$:

$$\mathbf{H'}_i^{proj} = \mathrm{MLP}(\mathrm{MultiHeadAttn}(\mathbf{H}_i^{proj}, \mathbf{S'^e}, \mathbf{S'^e})) \tag{4.6}$$

Finally, the knowledge enriched contextual word representation output of the KAR is a projection of $\mathbf{H'}_i^{proj}$ back to BERT contextualized word representation space with an added skip connection:

$$\mathbf{H'}_i = \mathbf{H'}_i^{proj} \mathbf{W'}^{proj} + \mathbf{b'}^{proj} + \mathbf{H}_i^{proj} \tag{4.7}$$

where $\mathbf{W'}^{proj}$ and $\mathbf{b'}^{proj}$ are learned.

The linked entity for span s is simply $e_{s,\mathrm{argmax}(\boldsymbol{\psi}_s)}$.

4.2.1.4 Training

There are three training steps for KnowBert models. First, once the mention generator is written, the KAR is trained on the Entity Linking task on spans given by the corpus, minimizing a log-likelihood loss for the predicted probability distribution over candidate entities:

$$\mathcal{L}_{\mathrm{EL}} = -\sum_s log\left(\frac{\exp(\psi_{sg})}{\sum_{k=1}^{n}\exp(\psi_{sk})}\right) \tag{4.8}$$

with ψ_{sg} the score for the ground truth entity in $\boldsymbol{\psi}_s$.

The second training phase involves continuing the pre-training of BERT using both a Masked Language Model and a Next Sentence Prediction objective. This phase corrects the disruptions incurred by the Language Model when grafting the KAR between the Transformer Layers in BERT. This step also adjusts the weights of the KAR for Entity Linking, minimizing:

$$\mathcal{L}_{\text{KnowBert}} = \mathcal{L}_{\text{BERT}} + \mathcal{L}_{\text{EL}} \tag{4.9}$$

We call this phase the "re-training" step to differentiate it from the BERT pre-training step and the fine-tuning step.

The final step, as for most pretrained LMs, is to fine-tune it to the target task.

4.2.2 . Preliminary experiments

4.2.2.1 Masked LM and Next Sentence Prediction

For a large source of raw biomedical text, we scraped the PubMed Central database of Open Access articles and processed them for next sentence prediction using the tool provided with the source code for "Knowledge Enhanced Contextual Word Representations" by Peters *et al.* (2019). At the end of this training phase, KnowBert can be used as a typical pretrained BERT model.

Due to time constraints, we were unable to generate candidates for the approximately 180M sequences in the corpus, and had to limit our re-training corpus to approximately 6M sequences. As shown in Table 4.1, this lack of re-training data has prevented the language model from successfully integrating the KAR, with a masked LM perplexity several orders of magnitude larger than $\text{BERT}_{\text{BASE}}$, $\text{BERT}_{\text{LARGE}}$, and the KnowBert models produced by Peters *et al.*

Model	Perplexity
$\text{BERT}_{\text{BASE}}$	5.5
$\text{BERT}_{\text{LARGE}}$	4.5
KnowBert-Wiki	4.3
KnowBert-Wordnet	4.1
KnowBert-W+W	3.5
KnowBert-UMLS	10387.7

Table 4.1: Masked Language Model perplexity for both BERT models, the KnowBert variants produced by Peters *et al.* (2019), and KnowBert-UMLS.

4.2.2.2 NER

We choose to fine-tune KnowBert-UMLS on the Biomedical Named Entity Recognition task on the n2c2 corpus, previously known as i2b2 2010 (Uzuner *et al.*, 2011), with an 80% - 20% split between training and validation sets using cross-entropy loss. In Table 4.2, we compare our performance versus four BERT-based

Model	P	R	F1
$\text{BERT}_{\text{BASE}}$	0.85	0.87	0.86
BioBERT	0.86	0.88	0.87
clinicalBERT	0.87	0.88	0.88
BlueBERT	0.88	0.90	0.89
KnowBert-UMLS	0.80	0.81	0.80

Table 4.2: Performance of BERT-based language models on the n2c2 NER task, measured as Micro-averaged strict Precision, Recall and F1. Results for BioBERT, clinicalBERT and BlueBert from Fraser *et al.* (2019).

Sequence	total	abdominal	hysterectomy	and	bilateral	salpingo-oophorectomy	.
True	B-treatment	I-treatment	I-treatment	O	B-treatment	I-treatment	O
Predicted	B-treatment	I-treatment	I-treatment	O	B-treatment	I-treatment	O

Table 4.3: Example of correct NER predictions by KnowBert-UMLS pulled from the n2c2 evaluation set.

models, namely BioBERT (Lee *et al.*, 2019), clinicalBERT (Alsentzer *et al.*, 2019), BlueBERT (Peng *et al.*, 2019) and $\text{BERT}_{\text{BASE}}$, all fully fine-tuned on the NER task with the same linear classifier architecture. The performance of our various baselines were taken from Fraser *et al.* (2019).

Examples of correct and incorrect predictions made by KnowBert-UMLS, formatted according to the IOB2 standard, can be found in tables 4.3 and 4.4 respectively. The example in table 4.4 is a quite typical incorrect prediction, as it consists of a span that overlaps with the correct span and has a correct label. This type of error is the most common, constituting 32% of the model's mistakes. Many of these mistakes are ambiguous even to humans – for instance, the matter of having to include the token "Estimated" in the "blood loss" entity is not self-evident. We perform a complete breakdown of error types as specified by Fraser *et al.* (2019) in table 4.5.

Sequence	Estimated	blood	loss	was	100	cc
True	B-problem	I-problem	I-problem	O	O	O
Predicted	O	B-problem	I-problem	O	O	O

Table 4.4: Example of incorrect NER prediction by KnowBert-UMLS pulled from the n2c2 evaluation set.

Error type	Proportion (%)
Correct label	
Overlapping span	32.0
Incorrect label	
Overlapping span	10.2
Correct span	15.1
False positive	29.1
False negative	13.6

Table 4.5: Breakdown of types of mistakes made by KnowBert-UMLS in proportion of total prediction mistakes made.

While the contextualized word representations contain enough information for the classification model to perform significantly better than chance, our results reveal a decrease in performance with respect to a non-modified $\text{BERT}_{\text{BASE}}$. This is further demonstration of the fact that the re-training procedure is the performance bottleneck and requires more text than our candidate generator can realistically process in a reasonable time frame.

Our evaluation is performed with SeqEval (Nakayama, 2018) in strict mode. Like the results from Fraser *et al.* (2019), its metrics are on an entity-level rather than at token-level, meaning that a true positive is a fully matching mention span. A predicted mention that overlaps with a true mention but is not identical counts as a false positive and a false negative.

4.2.3 . Conclusions

Successfully integrating UMLS knowledge into a pretrained LM using the KnowBert method presents a significant challenge due to the size of the knowledge base and the difficulty of generating candidate mentions. Our candidate generator based on QuickUMLS was not able to generate candidates with enough efficiency and precision to make re-training possible at the required scale. We are currently working on generating candidates for larger chunks of the re-training corpus in order to evaluate the progress made by KnowBert-UMLS as a function of corpus size, and make projections on its performance when trained on the full dataset.

4.2.3.1 Future work

In order to successfully re-train KnowBert-UMLS, the candidate generator must be improved significantly. Its main source of false negatives is the introduction of abbreviations of long terms in the beginning of the text which are subsequently re-used. These abbreviations are often absent from the UMLS metathesaurus and cannot be identified by the generator. Solving this issue would likely increase recall significantly when identifying candidate spans. This may allow a different recall/time compromise to be found within QuickUMLS settings.

Regardless of possible improvements to recall however, deploying this at scale, whether for re-training or practical text processing purposes, is likely to remain prohibitively slow for most individuals and organizations. Future work will involve finding a more effective and computationally efficient approach to tackle candidate generation, for instance as a machine learning problem or with a fast NER-based span pre-selection step.

Furthermore, whilst we chose to evaluate the performance of KnowBert-UMLS using $\text{BERT}_{\text{BASE}}$ as a backbone to isolate the effect of the KAR, the KnowBert method has the advantage of being compatible with other approaches such as BioBERT, clinicalBERT, BlueBERT, or SciBert. In addition to the potential performance improvements on biomedical tasks, these pretrained models may be less expensive to re-train due to the potentially smaller distributional shift between pre-training, KAR training, and re-training corpora.

4.2.3.2 Perspectives

In addition to the improvements that need to be made to the candidate generator to make KnowBert-UMLS competitive, there are a number of potential ways to enhance it and expand its range of applicability.

Multiple Knowledge Bases As shown by Peters *et al.* (2019), KnowBert is capable of accommodating multiple KARs for multiple KBs simultaneously. Depending on the practical application, it could be useful to develop a KnowBert model combining UMLS with WordNet, Wikipedia, YAGO (Suchanek *et al.*, 2007), or other specialized KBs. It would also be interesting to assess the performance of one such model in order to understand to what extent multi-specialization is possible.

Re-training with Adapters Adapters, as proposed by Houlsby *et al.* (2019), have seen some success for efficiently fine-tuning pretrained LMs such as BERT. It is conceivable that this approach may aid in the re-training process by reducing the number of parameters to train, and may help reduce the memory footprint of KnowBert in some practical applications. Specifically, in cases that involve multiple knowledge bases or sets of knowledge bases used independently from each other, such an approach may allow one copy of a pretrained LM to be loaded into memory whilst the relevant set of KARs and adapters can be applied as a function of the token sequence being processed.

4.3 . Adapting without forgetting: KnowBert-UMLS

Domain adaptation in pretrained language models usually comes at some cost, most notably out-of-domain performance. This type of specialization typically relies on pre-training over a large in-domain corpus, which has the side effect of causing catastrophic forgetting on general text. We seek to specialize a language model by incorporating information from a knowledge base into its contextualized representations, thus reducing its reliance on specialized text. We achieve this

by following the KnowBert method, applied to the UMLS biomedical knowledge base. We evaluate our model on in-domain and out-of-domain tasks, comparing against BERT and other specialized models. We find that our performance on biomedical tasks is competitive with the state-of-the-art with virtually no loss of generality. Our results demonstrate the applicability of this knowledge integration technique to the biomedical domain as well as its shortcomings. The reduced risk of catastrophic forgetting displayed by this approach to domain adaptation broadens the scope of applicability of specialized language models.

With the density and recurrence of specialized vocabulary in some fields such as STEM or law, transformer-based contextualized language models (LMs) such as BERT (Devlin *et al.*, 2019), trained on general text, tend to underperform in those fields. One obvious solution to this out-of-domain performance problem is to reduce the distributional shift between pre-training and deployment by pre-training on in-domain text. While this method generally performs well (Gururangan *et al.*, 2020), it does have several drawbacks. First, language models are inefficient at learning factual information. This is evidenced by the fact that they are not reliably able to predict facts that appear in their training data. For instance, let us present the following sentence from English Wikipedia (on which BERT models are trained) to $\text{BERT}_{\text{LARGE}}$:

> *Diabetes is due to either the pancreas not producing enough insulin, or the cells of the body not responding properly to the insulin produced.*

When we mask the word *pancreas* (which is split into wordpieces *pan*, *-cre* and *-as*), BERT ranks the relevant wordpieces as the 7088$^{\text{th}}$, 3999$^{\text{th}}$, and 4864$^{\text{th}}$ most likely completions at their respective positions. While this is only one example of failure to memorize the relationship between specific entities, it does betray a lack of useful knowledge relative to the amount of text required to train the model. A large quantity of text is therefore needed to capture factual relationships between specialized entities. Consequently, models specialized by pre-training on in-domain text typically require several billion words of in-domain text, as with the approximately 21 billion words in the training corpus for BioBERT (Lee *et al.*, 2019) and 57 billion words for LEGAL-BERT (Chalkidis *et al.*, 2020). Furthermore, while this in-domain pretraining solution is the most commonly used, Arumae and Bhatia (2020) and Xu *et al.* (2020) demonstrate that this approach tends to create models that perform well on the target domain, but poorly on general text, even when such text is part of the pre-training curriculum — *i.e.* models pretrained on large in-domain corpora are prone to *catastrophic forgetting*.

Reducing the risk of catastrophic forgetting in domain adaptation would broaden the scope of applicability of specialized language models. Whilst it is self-evident that expanding the capabilities of a language model is generally desirable, this also has specific applications. Science communicators, for instance, who often distill scientific publications to short, layman-readable articles, must have a good understanding of the concepts and methods in their field, be able to identify the important novel ideas, discard non-essential details, reformulate complex and precise concepts (*e.g.* "Pericarditis") into simpler and fuzzier language (*e.g.* "inflamma-

tion of tissue around the heart"), and identify concepts that are considered too obvious to mention in the published works but which outsiders may not know of (*e.g.* "correlation does not imply causation"). These tasks require a high level of proficiency in scientific as well as non-specialized language. Instilling this dual proficiency into language models could for instance enable automatic fact-checking of popular science articles, or scientific publication summarization for non-experts. More generally, automatic detection and clarification of complex concepts in documents written by experts, such as medical, legal or financial documents, could be very helpful in assisting laypeople with administrative tasks and would benefit greatly from both a fine grained understanding of specialized concepts and a high proficiency in non-specialized language.

Our approach to language model specialization is to provide it with an external source of relevant knowledge, reducing the need for in-domain text during pre-training as well as the risk of catastrophic forgetting. This allows the model to access information pertaining to concepts not seen in the training corpus. This type of approach would usually imply performing an Entity Linking (EL) step – *i.e.* identifying the mentions of concepts in the input text – ahead of leveraging the information in the knowledge base (KB). In practice, satisfying this requirement with a good enough degree of accuracy to render knowledge integration useful is a problem that has yet to be solved. Peters *et al.* (2019) on the other hand describe KnowBert, a method to enable a pretrained LM such as BERT to utilize information from a KB which relaxes this constraint, requiring only *candidate* mentions. Following this procedure, we inject knowledge from the Unified Medical Language System (UMLS) Metathesaurus into a BERT-based language model. We name the resulting model KnowBert-UMLS.

We expand on the context for this work and discuss other approaches used for solving these problems in Section 2.5. Because of the scale, variety, and lexical polymorphism of the concepts recorded in UMLS, as well as the relative scarcity of corpora containing labeled examples, there are specific challenges linked to applying KnowBert to UMLS, which we detail in Subsection 4.3.1. In Subsection 4.3.2, we discuss the relative performance of our model with respect to relevant baselines on in-domain and out-of-domain tasks. Finally, Subsection 4.3.3 is dedicated to our conclusions and future work.

4.3.1 . KnowBert-UMLS

The blueprint for KnowBert-UMLS, detailed in Figure 4.4 (a), is based on KnowBert, and comprises three main sections: the pretrained LM backbone, the knowledge base with its candidate generator, and a Knowledge Attention and Recontextualization module (KAR).

4.3.1.1 Architecture

4.3.1.1.1 Pretrained LM backbone BERT-based models comprise L Transformer Encoder Blocks, with each block i taking as input N non- or partially-contextualized

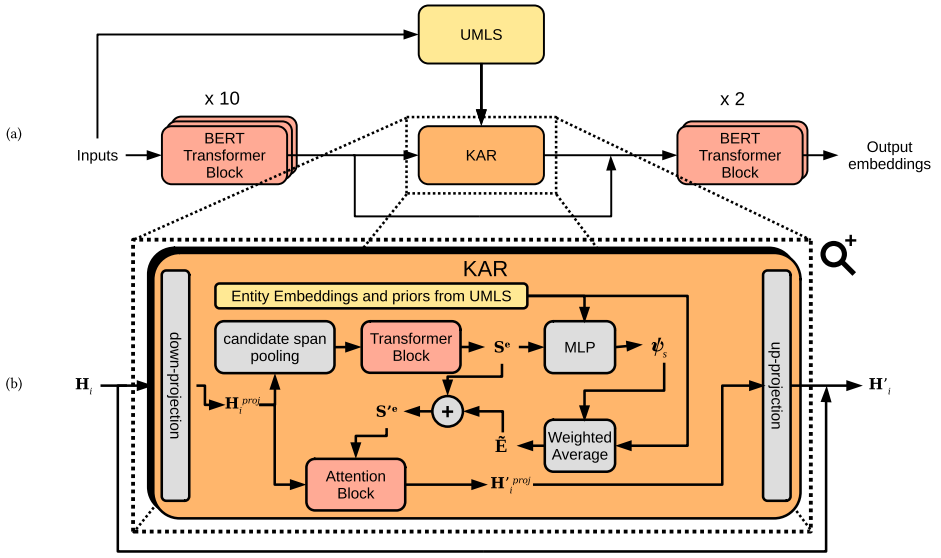


Figure 4.4: Overview of the structure of KnowBert-UMLS (a) with detailed breakdown of the KAR (b).

token representations in $\mathbb{R}^H$, arranged as a matrix $\mathbf{H}_{i-1} \in \mathbb{R}^{N \times H}$, recontextualizing them using attention, and returning a same-size matrix $\mathbf{H}_i$.

As a backbone, we use $\text{BERT}_{\text{BASE}}$, which is pretrained on the Wikipedia and Books (Zhu *et al.*, 2015) corpora containing approximately 3.3 billion words, and for which the aforementioned hyperparameters are $N = 512$, $L = 12$ and $H = 768$. Despite having, in theory, the option to use any transformer-based language model, in an effort to isolate variables, we do not choose to use a higher-performing or specialized alternative. Moreover, BERT is better suited to the objective of this study, which is to pursue knowledge enrichment without catastrophic forgetting, rather than to top state-of-the-art performance in any given task.

4.3.1.1.2 Candidate Generator Whilst KnowBert does not require an upstream Entity Linking step, it does require a set of *candidate mentions* $\mathcal{C}$ in order to incorporate information from the KB. Each candidate mention comprises a *candidate span* s and a set $\mathcal{E}_s$ of corresponding *candidate entities* from the KB. Formally:

$$\mathcal{C} = \{(s, \mathcal{E}_s) | \forall s\} \tag{4.10}$$

Each candidate entity $e \in \mathcal{E}_s$ represents a concept in the KB and is composed of an embedding $\mathbf{e}$, and a prior probability p:

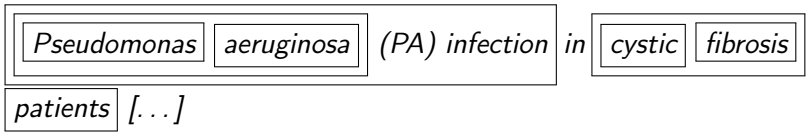
$$\mathcal{E}_s = \{e : (\mathbf{e}_e, p_e) \mid \mathbf{e} \in \mathbb{R}^K, \sum_e p_e = 1\} \tag{4.11}$$

where K is determined by the algorithm used to derive entity embeddings from the KB. In the case of UMLS, we use the pretrained embeddings provided by Maldonado *et al.* (2019) which set $K = 50$.

A candidate span can be any sub-string in the sentence which is deemed sufficiently similar to an entity in the KB, and can overlap, or be nested with other spans. For instance, consider the following phrase:

> *Pseudomonas aeruginosa (PA) infection in cystic fibrosis (CF) patients [. . .]*

A candidate generator might generate the following candidate spans, outlined in boxes:



Or, more explicitly: *Pseudomonas*; *aeruginosa*; *Pseudomonas aeruginosa*; *Pseudomonas aeruginosa (PA) infection*; *cystic*; *fibrosis*; *cystic fibrosis*; and *patients*. Each of these candidate spans would be paired with a set of candidate entities $\mathcal{E}$.

4.3.1.1.3 KAR On an abstract level, the KAR remains largely unchanged from KnowBert. It slots in-between two BERT layers i and $i+1$ and functions similarly to a Transformer Block, taking as input partially contextualized word representations $\mathbf{H}_i$ and outputting knowledge-enriched, recontextualized word representations $\mathbf{H'}_i \in \mathbb{R}^{N \times H}$. As an additional input, it takes a set of *candidate mentions* $\mathcal{C}$.

The knowledge incorporation step is performed in the entity embedding space $\mathbb{R}^{N \times K}$; the KAR thus linearly projects the partially contextualized wordpiece embeddings to the entity space and back:

$$\begin{aligned} \mathbf{H}_i^{proj} &= \mathbf{H}_i \mathbf{W} + \mathbf{b} \\ \mathbf{H}'_i &= \mathbf{H}_i'^{proj} \mathbf{W}' + \mathbf{b}' + \mathbf{H}_i \end{aligned} \tag{4.12}$$

where $\mathbf{W}$, $\mathbf{W}'$, $\mathbf{b}$ and $\mathbf{b}'$ are learned and $\mathbf{H}_i'^{proj}$ is the matrix of knowledge-enriched token representations embedded in entity space (see Figure 4.4 (b)).

The knowledge integration process itself comprises four main steps. First, the token representations for each candidate mention are pooled using an attention-based weighted sum following Lee *et al.* (2017) into a matrix $\mathbf{S} \in \mathbb{R}^{|\mathcal{C}| \times K}$. In order to identify false positives among nested or overlapping candidate spans as well as commonly co-occurring entities, the span representations exchange information using Multi-Head self-Attention as in a standard Transformer block, resulting in the contextualized span representations $\mathbf{S^e}$.

For every given span s, we write the corresponding contextualized span embedding from $\mathbf{S^e}$ as $\mathbf{s^e} \in \mathbb{R}^K$, the vector of prior probabilities of corresponding candidate entities $\mathbf{p}_s \in \mathbb{R}^{|\mathcal{E}_s|}$, and the matrix of corresponding candidate entity embeddings $\mathbf{E}_s \in \mathbb{R}^{K \times |\mathcal{E}_s|}$.

$$\boldsymbol{\psi}_s = \text{Softmax}(\text{MLP}(\mathbf{p}_s, \mathbf{s}^{\mathbf{e}} \cdot \mathbf{E}_s))$$

$$\tilde{\mathbf{e}}_s = \boldsymbol{\psi}_s \cdot \mathbf{E}_s^{\mathsf{T}}, \quad \in \mathbb{R}^K \tag{4.13}$$

where $\boldsymbol{\psi}_s$ is an estimate of the posterior probabilities of each candidate entity for s, and $\tilde{\mathbf{e}}_s$ is a weighted average of candidate entity vectors for s. As an additional benefit beyond knowledge integration, this can be used for Entity Linking by simply choosing the entity in the KB which has the embedding most similar to $\tilde{\mathbf{e}}_s$ or the one with the highest estimated posterior probability.

The knowledge-enriched span representations $\mathbf{S}'^{\mathbf{e}} \in \mathbb{R}^{|\mathcal{C}| \times K}$ are then defined as the sum of the contextualized span representations $\mathbf{S}^{\mathbf{e}}$ and the matrix of computed entity vectors $\tilde{\mathbf{E}}$, and the knowledge is transferred from the span representations to the wordpiece embeddings using Multi Head Attention (MHA) followed by a position-wise Multi-Layer Perceptron (MLP), similarly to a Transformer block:

$$\mathbf{H}_i'^{proj} = \text{MLP}(\text{MHA}(\mathbf{H}_i^{proj}, \mathbf{S}'^{\mathbf{e}}, \mathbf{S}'^{\mathbf{e}})) \tag{4.14}$$

The KAR can be inserted between any two transformer blocks i and $i+1$, and multiple KARs can be inserted simultaneously between different blocks. In the single-KAR case however, using $\text{BERT}_{\text{BASE}}$ as the pretrained backbone, insertion is most effective at block $i = 10$.

4.3.1.2 *Training*

The pre-training for KnowBert models is a three step process. First the backbone is pre-trained on a language modeling objective, specifically on a combination of the MLM and next sentence prediction objectives in the case of BERT. Then, the KAR is trained on the Entity Linking (EL) objective, minimizing the log-likelihood of the estimated posterior probabilities of candidate entities:

$$\mathcal{L}_{\text{EL}} = -\sum_s log\left(\frac{\exp(\psi_{sg})}{\sum_{k=1}^{n}\exp(\psi_{sk})}\right) \tag{4.15}$$

The model is subsequently trained on both the language modeling objective used in pre-training and EL. This step is similar to an extended pre-training, but its main objective is to allow the Transformer Blocks which receive knowledge enriched information from the KAR to learn to interpret and integrate it. To avoid ambiguity with regular pre-training, we call this step *re-training*. Optimizing only one of the two objectives during this phase leads to catastrophic forgetting of the other, even when the weights of the KAR are frozen. Once the KAR is fully integrated into the model with this step, the model can be fine-tuned to any task much like a typical BERT-based model.

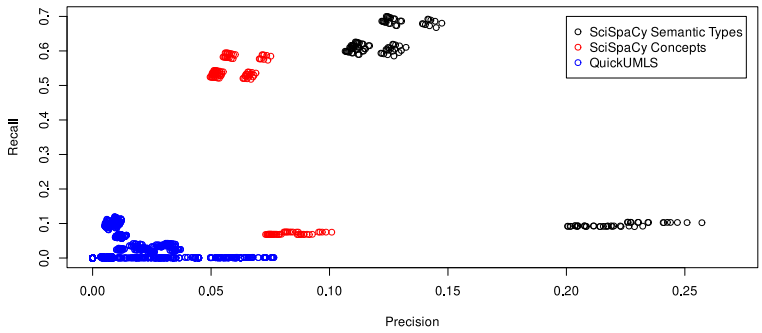


Figure 4.5: Precision-Recall Plot for QuickUMLS and ScispaCy candidate generator configurations, with Semantic Type versus Concept-based candidate generation breakdown for ScispaCy. Evaluation carried out on a subset of the MedMentions dataset.

4.3.1.3 Leveraging UMLS

UMLS indexes over four million biomedical concepts, such as *headache*, *spleen*, or *acetylsalicylic acid*, grouped into 135 semantic types including *organisms*, *anatomical structures*, and *diseases or syndromes*. So far, due in part to the difficulty of reliably identifying UMLS concepts in text, leveraging the concepts themselves from UMLS has been out of reach of knowledge integration techniques. UmlsBERT, for instance, only uses embeddings for clusters of semantic types and uses UMLS as a thesaurus for single-word concepts.

Whilst the computational impact of the KAR is negligible, making candidate generation tractable is non-trivial at the scale of our KB and re-training corpus as discussed by Piat *et al.* (2022b) (Section 4.2). We have benchmarked a variety of algorithms, most of which were far too computationally inefficient for practical use in this context. QuickUMLS was a promising choice, as it was a tool designed for this purpose, and is faster and more accurate than any other string-similarity-based algorithm. However, as illustrated in Figure 4.5 which plots Precision versus Recall for multiple candidate generator configurations, QuickUMLS configurations (in blue) performed poorly in comparison to ScispaCy (Neumann *et al.*, 2019), being ninety-seven times slower and generally suffering from lower precision and recall. Nevertheless, the combination of neural networks and rules used by ScispaCy was too computationally expensive to use in realtime during re-training, and the candidates had to be pre-generated. Candidate generation on a 153.6 Billion sentence corpus took approximately three days on a computing cluster, using fourteen Xeon 36-thread CPUs clocked at 3GHz.

We compared the performance of KnowBert using both UMLS Concepts and UMLS Semantic Types as KB entities forming the basis of our knowledge integration. Attempting to integrate knowledge from UMLS concepts did not work, resulting in the model performing on par with an unmodified BERT, as it learned

to not take into account the knowledge integrated by the KAR. We suspect two main factors are at play, leading to the KAR being unable to accurately learn to estimate the posterior probabilities for the candidate concepts. First, the precision of the candidate generator is lower for UMLS concepts, as illustrated in Figure 4.5 (points in red). For the candidate generator, Recall bounds the model's ability to incorporate knowledge (since no knowledge can be incorporated from an entity not identified by the candidate generator), and Precision affects the imbalance between positive and negative samples during the EL objective. Therefore, whilst maximizing recall maximizes the model's potential, doing so at the cost of precision increases noise in the EL dataset and makes learning more difficult.

The second factor we believe to be responsible for the underperformance of Concept knowledge integration is the lack of concept coverage in the training data, which is under 1% of all concepts, with a Zipfian distribution of occurrences. Consequently, setting the weight of the KAR's contributions to 0 is the policy which most accurately predicts masked tokens during training. Despite the Semantic Type information being less insightful than the Concept information, the increased quality of candidates and density of training data in annotated examples (approximately 94% coverage of all types, and 79% of types covered better represented than in a Zipfian distribution) make Semantic Type information worth incorporating. Henceforth, all mentions of KnowBert-UMLS assume that we use Semantic Types as knowledge base entities unless explicitly stated.

4.3.2 . Experiments

To evaluate our model, we choose two in-domain and two out-of-domain tasks. For our in-domain tasks, we choose Named Entity Recognition (NER) on the n2c2 (formerly known as i2b2) 2010 dataset (Uzuner *et al.*, 2011) and Relation Extraction (RE) on the ChemProt (Krallinger *et al.*, 2017) dataset. These tasks are fairly standard, with most biomedical language models having their performance published, and are part of the BLUE (Peng *et al.*, 2019) benchmark. For our out-of-domain tasks, we choose Natural Language Inference (NLI) and Linguistic Acceptability as Arumae and Bhatia (2020) demonstrated that these tasks were particularly affected by extended-pretraining-induced catastrophic forgetting with BioBERT. Specifically, we choose the SNLI (Bowman *et al.*, 2015) dataset, and an altered version of the CoLA (Warstadt *et al.*, 2019) dataset (see section 4.3.2.4 for details) respectively.

All performance scores have been scaled up (from $[0, 1]$ or $[-1, 1]$) by a factor of 100 for readability. In all tables, the underlined result is BERT which, as a general language model, is expected to have the best performance on the general language tasks (CoLA & SNLI) and worst on biomedical tasks (n2c2 & ChemProt). In bold is the best specialized model.

We choose our baselines to represent various amounts of in-domain and out-of-domain pre-training corpus sizes, which we break down in table 4.6. Aside from $\text{BERT}_{\text{BASE}}$, we compare performance against the following baselines: BlueBERT and BioBERT as they are respectively pretrained moderately and heavily on biomedical text (4.5 versus 18 gigawords in addition to the 3.1 gigawords of

Model	Biomedical	General	Knowledge integration
$\text{BERT}_{\text{BASE}}$	0.0	3.1	No
PubMedBERT	3.2	0.0	No
BlueBERT	4.5	3.1	No
BioBERT	18.0	3.1	No
UmlsBERT	18.5	3.1	Yes
KnowBert-UMLS	**2.2**	**3.1**	**Yes**

Table 4.6: Pre-Training Corpus Size (Billions of Words) by Type for Baselines Versus KnowBert-UMLS.

the BERT pre-training corpus); PubMedBERT as it is pretrained on an amount of text very similar to $\text{BERT}_{\text{BASE}}$ (3.2 versus 3.1 gigawords respectively) but which is exclusively biomedical; and UmlsBERT which is similar to BioBERT (with only five hundred million additional words from a biomedical corpus) but includes a knowledge integration mechanism as part of its pretraining.

For all models and all tasks, final token or sequence classification is performed using a linear classifier. For all experiments, all models were fine-tuned with the following hyperparameters: our models are trained for 10 epochs with an initial learning rate of 2×10^{-5} and weight decay of 0.01. Our optimization algorithm is AdamW. The model state which performed best on the validation split of each dataset was evaluated on the test set. Results are averages over multiple experiments.

4.3.2.1 Biomedical NER

In this task, models must identify locations where named entities are mentioned and tag them as instances of *problem*s, *test*s or *treatment*s. As the methodology of previously published results for our baselines is inconsistent, we evaluate the various BERT-based language models on this task ourselves. We use the micro-averaged F_1score as computed by Seqeval (Nakayama, 2018) in strict mode and provide a breakdown of precision and recall, as sub-sequence classification (as opposed to sequence classification) does not cause micro-averaged precision and recall to necessarily be equal. We use IOB2 as our annotation and prediction scheme.

We expect an improvement over BERT due to specialization, whilst UmlsBERT, as the most heavily pretrained language model which also benefits from knowledge integration, should perform best overall. From the results in Table 4.7, we gather that the KAR succeeded in specializing the model, as KnowBert-UMLS outperforms $\text{BERT}_{\text{BASE}}$ by a considerable margin. However, the KAR seems to not have been quite as effective of a specialization method for NER as the others. In particular, despite the candidate span generator providing it with additional information on named entities in the text, its recall is the lowest among the models we assess. This is likely due to the discrepancy between the terms chosen by the candidate generator and the KAR for enrichment, and the parts of speech expected to

be labeled for this task. Specifically, the n2c2 corpus requires possessive pronouns, determiners, articles, adjectives and other qualifiers to be included in the entity, whereas UMLS (and therefore the candidate generator) requires the opposite. For instance, the following sentences occur in the n2c2 dataset:

1. On postop day number two she was also afebrile$_{\mathbf{1}}$ and had not passed any flatus$_{\mathbf{2}}$ yet.

2. Administer iron products a minimum of 2 hours before or after a levofloxacin$_{\mathbf{3}}$ or ciprofloxacin dose$_{\mathbf{4}}$ [...]

The named entities which are expected to be marked are underlined and numbered with subscripts. The second and third entity mentions include determiners (*any* and *a* respectively) which manual model output examination reveals are not identified by KnowBert-UMLS, whereas the main words in the entity mentions (*flatus* and *levofloxacin* respectively) are accurately identified and tagged. This is by far the predominant type of false negative within the reviewed set of samples and mirrors the words identified as entity mentions by the candidate generator, indicating that the model may have low confidence on terms that do not benefit from knowledge integration.

In comparison, BlueBERT's most prominent weaknesses seem to be that it misses entity mentions entirely or does not include all of the words which the mention comprises. In the aforementioned example, for instance, BlueBERT does not recognize the second mention (*any flatus*) as an entity, and it excludes the word 'dose' in the fourth mention. BlueBERT also struggles with determiners, but to a lesser degree. In particular, we have not been able to find an example where BlueBERT misses a possessive pronoun. In fact, possessive pronouns seem to be a prominent source of false positives for BlueBERT, as it tends to tag non-biomedical nouns when they follow one.

Lastly, while this is not a highly prominent issue for either model, KnowBert-UMLS confuses entity types more often than BlueBERT, with approximately 12.7% of KnowBert-UMLS' mistakes being of this kind, versus 8.9% for BlueBERT. This is surprising, as one would expect that integrating knowledge of semantic types would specifically target this type of mistake, assuming there is an (approximately) surjective mapping of n2c2 entity types onto UMLS semantic types. This assumption is not verified however as, for instance, the '*Finding*' semantic type may correspond to either the '*test*' or '*problem*' entity types, and many semantic types related to substances can correspond to either '*test*' or '*treatment*'. Furthermore, words which are typically used in one context and punctually used in another can be matched with the wrong semantic type, priming the language model with erroneous information, causing it to predict the wrong entity type. For instance, the word '*splint*' typically refers to a type of brace (which would correspond to the *'treatment'* entity type), but can refer to '*shin splints*', a type of injury, *i.e.* '*problem*'.

Model	P	R	F_1
$BERT_{BASE}$	82.71	86.21	<u>84.42</u>
BioBERT	85.20	87.74	86.46
PubMedBERT	86.62	88.28	87.44
BlueBERT	86.68	88.71	87.68
UmlsBERT	86.92	89.46	**88.18**
KnowBert-UMLS	86.63	85.84	86.23

Table 4.7: Performance on the n2c2 2010 NER Task.

Model	micro F_1
$BERT_{BASE}$	<u>66.51</u>
BioBERT	75.14
PubMedBERT	**77.24**
BlueBERT	69.15
KnowBert-UMLS	70.74

Table 4.8: Performance on the ChemProt Relation Extraction Task, Micro-F_1.

4.3.2.2 Biomedical Relation Extraction

This is a sequence classification task, wherein two entities per sequence are marked with special characters, and the model must determine which of five relation types (or no relation) exists between them. Scores for BioBERT, PubMedBERT and BlueBERT are self-reported scores of the overall best-performing version of each model. The performance of $BERT_{BASE}$ was measured by us using the version of the ChemProt corpus distributed by Peng *et al.* (2019).

Due to the entities being marked in all versions of the training set used by our baselines, the importance of a good model for grammar is lessened with respect to other tasks. We therefore do not expect general language understanding to be highly predictive of performance on this task. Rather, knowledge of biomedical entities and their relations is expected to be of greater importance. We therefore expect KnowBert-UMLS, which seeks to acquire specifically this type of knowledge, as well as the heavily pretrained BioBERT model, to perform well on this task, with $BERT_{BASE}$ performing worst. Our results in Table 4.8 largely correspond to what we expected. PubMedBERT, however, which was pretrained only on biomedical text, performs better than expected, implying out-of-domain pre-training may in fact be detrimental.

Once more, we observe that KnowBert-UMLS outperforms BERT, implying that specialization was successful, yet it underperforms in comparison to other pretrained models. BioBERT being outperformed by PubMedBERT is unexpected, but may be due to the small pre-processing differences in the version of the corpus used by BioBERT.

4.3.2.3 General NLI

We evaluate all models ourselves on the SNLI task, in which two sequences, a *premise* and *hypothesis*, are fed to the LM. It must determine whether the premise *entails* the hypothesis, whether they are *contradictory*, or neither (their relationship is "*neutral*"). As this is a general language task, we do not expect any specialized model to outperform $\text{BERT}_{\text{BASE}}$ by a significant margin, and we expect KnowBert-UMLS to perform on par with BERT. Due to the similarities between the SNLI and WNLI task from GLUE, which BioBERT struggles with according to Arumae and Bhatia (2020), we expect for the models with extended pre-training to perform poorly on this task, particularly UmlsBERT for which the knowledge integration process relaxes the grammatical correctness constraint.

Table 4.9 shows results in line with our predictions, *i.e.* KnowBert-UMLS is the closest to $\text{BERT}_{\text{BASE}}$ in terms of F_1 score which is consistent with a reduction of catastrophic forgetting leading to better performance. However, the gap in performance between the models with extended pre-training and the others is narrower than expected given BioBERT's performance on the similar WNLI task according to Arumae and Bhatia (2020). This is likely due to SNLI not being as adversarial as the WNLI benchmark.

Model	micro F_1
$\text{BERT}_{\text{BASE}}$	89.24
BioBERT	88.90
PubMedBERT	88.81
BlueBERT	88.20
UmlsBERT	88.59
KnowBert-UMLS	**89.03**

Table 4.9: Performance on the SNLI Task, Micro-F_1.

KnowBert-UMLS and BERT perform with a high degree of similarity, quantitatively as well as qualitatively, as the specific instances of KnowBert-UMLS and BERT used in the following analysis share the same prediction on 94.53% of the instances in the test data. Both models slightly underperform compared to their respective averages, with micro-F_1 scores of 87.99 and 88.33 respectively. However, many of the examples in the SNLI dataset are open to interpretation. For instance:

> *Premise: A bearded man wearing a blue shirt and white t-shirt is working on a fishing net.*
> *Hypothesis: Someone is preparing to catch fish.*

While the hypothesis is the most likely explanation for the premise, it is not difficult to imagine scenarios where the premise is true and the hypothesis is false. It is therefore unclear whether the relationship is of entailment or neutral.

In order to capture this ambiguity, in addition to the reference label, each sentence has been manually labeled by five human reviewers. In this case, the label is 'entailment', but two out of five reviewers labeled the relationship between these sentences as 'neutral'. If we accept the reviewers' answers as valid predictions, the micro-F_1scores of KnowBert-UMLS and BERT (that is, the specific instances used in this analysis) on this task are 96.80 and 96.88 respectively, meaning only approximately one in four errors made by KnowBert-UMLS and BERT were in disagreement with all reviewers.

We decide to examine the mistakes made by both models and attempt to identify patterns. No clear tendencies could be found regarding the incorrect label predictions, but analyzing the instances themselves, we could group mistakes into four major types:

- *Blunders*, for which no information other than what is stated in the text is required to make a decision.
- *Common Sense* (CS) mistakes, which require some reasoning and/or a non-trivial piece of real-world knowledge.
- *Technically Correct* (TC) predictions, in which the model's answer could be considered correct based on an arguably valid interpretation of the text.
- *Not-an-Error* (NaE), for which we agree with the model and disagree with the label, or the input text contains a major corruption.

We provide examples of the aforementioned categories in Appendix A.1.1. After manual examination of the models' failure cases, we report a breakdown of mistakes by type in Table 4.10.

Model	Blunders	CS	TC	NaE
$\text{BERT}_{\text{BASE}}$	38.9%	33.7%	12.9%	14.9%
KnowBert-UMLS	38.6%	38.1%	7.3%	15.7%

Table 4.10: Breakdown of mistakes made by KnowBert-UMLS and BERT by type on the SNLI task.

The main recurring pattern seems to be for KnowBert-UMLS to lack real-world knowledge but stick to more straightforward interpretations of sentences. Illustrative examples for CS and TC mistakes are given in appendices A.1.2 and A.1.3 respectively. This may indicate that KnowBert-UMLS suffers from some amount of catastrophic forgetting on real-world knowledge, and perhaps is less prone to noticing and fixating on details which could skew its understanding of the text.

4.3.2.4 *Linguistic Acceptability*

Model	Matthews Corr.	micro F_1
$BERT_{BASE}$	60.50	83.68
BioBERT	49.30	78.75
PubMedBERT	42.90	76.28
BlueBERT	39.76	78.18
UmlsBERT	44.24	77.41
KnowBert-UMLS	**58.52**	**87.61**

Table 4.11: Performance on the modified CoLA task, Matthews' Correlation and Micro-F_1.

Our dataset for the Linguistic Acceptability task is based on the CoLA task from the GLUE benchmark. Since CoLA does not make the labels of its test split public however, and in the interest of being able to qualitatively analyze the types of mistakes made by the models in order to identify specific strengths and weaknesses, we have rearranged the available annotated examples into new training, validation, and test splits. Specifically, we use the validation split for final testing, and replace the validation split with a subset of the train split. In an effort to make our tests reproducible, we have decided to use the un-shuffled final 500 entries of the train split of version 1.1 as validation set.

The objective for this task is to classify sequences as "linguistically acceptable" (*i.e.* grammatically correct and natural-sounding) or not. We evaluate our model and baselines using micro-averaged F_1, as it is the most standard classification metric, as well as the Matthews Correlation Coefficient, which is the metric used by GLUE and is generally preferred in a binary classification setting as it isn't biased in favor of the positive class. As with NLI, we expect $BERT_{BASE}$, as the general purpose language model, to perform the best, and KnowBert-UMLS to come second as the way it is trained is meant to reduce catastrophic forgetting. UmlsBERT and BioBERT, on the other hand, should perform poorly due to additional pretraining.

Our results in Table 4.11 are consistent with these predictions, with the caveat that KnowBert-UMLS outperforms $BERT_{BASE}$ in F_1 score. This is due to a combination of class imbalance in the dataset, with well-formed sentences being overrepresented, and an inherent bias in the F_1 metric due to it not taking True Negative predictions into account. While KnowBert-UMLS performs best overall on the examples given, $BERT_{BASE}$ more accurately recognizes badly constructed sentences.

Furthermore, UmlsBERT does not perform quite as poorly as expected and BioBERT performs unexpectedly well given the additional biomedical pretraining it has undergone compared to the other specialized models. This may imply that the relationship between specialized pre-training and catastrophic forgetting is not monotonic but in fact more complex.

KnowBert-UMLS far outperforms specialized baselines on this task, demonstrating this method's effectiveness in avoiding catastrophic forgetting. Out of the

527 samples in our test set, the specific instance of KnowBert-UMLS used in this analysis yielded 64 false positives and 25 false negatives for an MCC of 58.35. In comparison, our BERT instance suffered 64 false positives and 20 false negatives for an MCC of 60.89.

Inspection of false negative instances reveals that some of the labels in the corpus reflect types of phrasing that are uncommon in modern written English. For instance:

Came right in he did without so much as a knock.

This phrasing is highly irregular outside of some areas of the UK and lacks punctuation.

Will he can do it?

This sentence uses double modals, which is a nonstandard construction seldom appearing outside of oral speech in Scotland, Northern Ireland and Northern England.

Rusty talked about himself only after Mary did talk about him.

While grammatically correct, the use of *did talk* rather than the straightforward simple past *talked* is uncommon in this context. This is likely another regional variance.

While it could be argued that general models should be able to handle nonstandard constructions for applications such as speech-to-text transcription or applications involving transcribed text, it can be desirable for a specialized model such as KnowBert-UMLS to classify them as incorrect as such sentence structures may be particularly unlikely to be intentional constructions by a native speaker in the contexts in which the model is susceptible to be deployed.

We therefore manually re-label these false negatives as true negatives in order to estimate the performance of KnowBert-UMLS in these types of contexts. We do not consider re-labeling false positives or true negatives as there are fewer examples of this occurring in the negative class and the practical interpretation would be unclear. Examination of the true positives did not yield any instances which we believe would have warranted relabeling. The full list of incorrect predictions for BERT and KnowBert-UMLS can be found in appendix A.2, with re-labeled instances marked by a right-facing arrow ('$\rightarrow$').

Taking these new labels into account, the MCC of Knowbert-UMLS' predictions on this task increases to 64.70, outperforming BERT's average performance. While the improved performance demonstrated by this interpretation of the data is meaningful, this score is not necessarily representative of real-world performance, as it neglects several factors. First, performing this relabeling on the train and validation sets in addition to the test set would likely give us a more accurate performance estimate. Second, this is only done for one trained model; results should be averaged over multiple experiments to be representative.

Furthermore, this comparison is (by design) unfair as we place different expectations on both models. When we relabel the false negatives predicted by BERT, its score increases by 2.35 fewer points than KnowBert-UMLS, to 64.89 MCC. KnowBert is therefore not only more prone to rejecting sentences as we can tell from its greater number of negative predictions, but is specifically more prone to rejecting non-standard constructions, which may be a desirable feature.

4.3.3 . Conclusions

4.3.3.1 Results

KnowBert-UMLS outperforms BERT on biomedical tasks whilst outperforming every other specialized model in out-of-domain tasks. Reducing extended pretraining in favor of Knowledge integration therefore proves to be a successful way of specializing a language model such as BERT to a given domain whilst reducing the impact of catastrophic forgetting. However, KnowBert-UMLS does not perform as well as some other models in the biomedical domain, meaning that its specialization method is less effective. We explain this by the fact that the Concepts in the UMLS knowledgebase are too numerous and annotated text too rare to learn from effectively, and Semantic Type knowledge is not as informative on an entity-mention level as familiarity with the vocabulary is.

The shortcomings of KnowBert-UMLS in the biomedical domain seem to reflect some lack of precision in biomedical knowledge, but more significantly, a difficulty grasping the tasks' expectations regarding which parts of speech to include within named entities. This is in line with the expectation that KnowBert's increased knowledge must come at the cost of some level of proficiency with grammar, and may reflect some level of heterogeneity and inaccuracy of information introduced by KnowBert's knowledge integration process. For out-of-domain tasks, KnowBert-UMLS seems to suffer from a lack of common-sense knowledge, but its slightly degraded performance on CoLA may reflect a desirable compromise which benefits adaptation to biomedical language.

4.3.3.2 Future work

In light of these observations, perhaps the most significant weakness of Knowbert-UMLS is a lack of common-sense knowledge. As Peters *et al.* have shown, a feature of the KnowBert architecture is that it can accommodate multiple knowledge graphs. Adding support for a general knowledge graph such as Wikipedia or a common-sense knowledge graph such as CSKG (Ilievski *et al.*, 2021) or ATOMIC (Sap *et al.*, 2019) could improve the performance of KnowBert-UMLS, particularly on general language and out-of-domain tasks, but perhaps also on biomedical tasks as this may reduce the heterogeneity of knowledge integration.

Following this method, KnowBert-UMLS may alternatively be further specialized in the biomedical domain with the integration of an additional biomedical KB such as OpenTargets' LINK, or it may even support multi-specialization, using knowledgebases from entirely different fields such as YAGO (Suchanek *et al.*, 2007) or WorldKG (Dsouza *et al.*, 2021). If this is shown to be possible, this method may

prove to be a computationally affordable way to increase the breadth of knowledge and reliability of multi-billion parameter language models such as GPT-3, potentially allowing one instance running on a computing cluster to serve the needs of a variety of professionals from different fields.

Another way of increasing performance in the biomedical domain, particularly to increase the exactitude of the entity types identified, may be to find a better compromise between the high granularity (and therefore informativity) of UMLS concepts and the higher candidate generator accuracy and training data availability on Semantic Types, perhaps by clustering related concepts or falling back onto semantic types only for the concepts on which the candidate generator is highly uncertain.

With our results on the CoLA task leading us to suspect that catastrophic forgetting does not monotonically increase with the amount of specialized pre-training, assessing the performance of models with various levels of specialized pre-training on various out-of-domain tasks could be enlightening and inform better pre-training policies in the future. Moreover, the n2c2 2010 NER and CoLA tasks suggest that the semantic enrichment of word representations from KnowBert-UMLS comes at the cost of some grammatical proficiency. That is to say, KnowBert-UMLS is not entirely immune to catastrophic forgetting. This may be addressable by adding general text to the MLM re-training objective, or adding one or more different, explicit grammar-oriented objectives such as POS-tagging in alternation with EL and MLM. Furthermore, an improved identification of False Positives from the candidate generator may increase the Precision over candidate entities and improve performance by reducing the amount of noise the KAR includes as it incorporates knowledge.

Lastly, the improvements brought by this method of knowledge integration may be orthogonal to the improvements brought by extended pre-training or other knowledge integration methods. In fact, such improved representations of biomedical text may help the KAR to reach the full extent of its capabilities. A comparative study of performance on a variety of in- and out-of-domain tasks by multiple KnowBert-UMLS-like models with different pretrained LM backbones such as RoBERTa, BioBERT, BlueBERT, and UmlsBERT would shed a valuable light on this topic and may lead to a new state of the art in biomedical language modeling.

4.4 . Conclusion

The behavior of KnowBert-UMLS provides evidence for knowledge integration having advantages over in-domain extended pretraining for the purposes of domain adaptation, most notably a lower tendency for catastrophic forgetting. However, the use of concept embeddings for the enrichment of token embeddings inherently requires the model to learn a mapping between the concept-embedding vector space and the token-embedding vector space. This in turn seems to require a considerable amount of in-domain text, which isn't available for all domains, as well as large computational resources. Additionally, even when supported by a candidate generator and learned in tandem with language modeling, EL remains a significant bottleneck when leveraging UMLS, and this appears likely to be true for other large, domain-specific ontologies, particularly given the requirement for labeled data when training EL. Lastly, it is not obvious that integrating changes to a KG following this knowledge integration approach could be made significantly more straightforward than training the model from scratch, as any change to the graph is liable to change all of the concept embeddings depending on the graph embedding algorithm.

It is apparent that language models are refractory to integrating the concept embedding modality, and most of the objectives we pursue in the knowledge integration approach to domain adaptation — reduced computational cost to updating the models crystallized intelligence, reduced reliance on in-domain text — are not met following this method. We therefore seek a different representation for our KB, and a method for integrating the knowledge contained within. The most obvious strategy is to represent our KB in model's default modality — text — and to distill the information present in the text into the model in the standard fashion — MLM pretraining. While this doesn't address all of the aforementioned issues, this is (in principle) a simple approach which does bypass the problems of EL and concept embeddings entirely, and may serve as a stepping stone for further improvements.

5 - Leveraging Knowledge Graphs as Text

Transformer-based language models have trouble integrating modifications whose purpose is to incorporate knowledge from structured, non-textual data such as knowledge graphs. Instances where this integration is successful generally require the problem of Entity Linking to be solved upstream, or the addition of a significant amount of (generally annotated) text to the training set. These constraints often make leveraging structured data difficult and/or counterproductive. We seek to adapt a language model to the biomedical domain through training on synthetic text derived from a knowledge graph, such that the information therein can be effectively leveraged in a format which the model can handle natively.

5.1 . Introduction & related work

Because of the specificity of concepts and vocabulary used in some domains, the performance of language models pre-trained on general text, such as BERT (Devlin *et al.*, 2019), tend to suffer in these domains. Various approaches to language model specialization exist, the most notable and obvious of which is specialization through pre-training on text from the target domain. The performance of this approach has been investigated by Gururangan *et al.* (2020), and many models specialized in this way have been met with success, such as LegalBERT (Chalkidis *et al.*, 2020) in the legal domain and PatentBERT (Lee and Hsiang, 2020) in the domain of patents. In the biomedical domain, which we focus on in spite of the applicability of our work hypothetically extending to any domain, a variety of models apply the same general principle, such as BioBERT (Lee *et al.*, 2019), BlueBERT (Peng *et al.*, 2019) and PubMedBERT (Gu *et al.*, 2021). This approach, however, is not without its drawbacks. In particular, it is prone to catastrophic forgetting (Xu *et al.*, 2020), and requires a large amount of in-domain text to derive a significant benefit, which is not an option available in all domains. In addition, as observed by Zipf (1935), the distribution of words in natural language is highly biased, which results in a linear increase in concept coverage requiring an exponential growth in training text.

A competing class of approaches consists in integrating information from a specialized knowledge base. Various strategies have been put froward, such as using the attention mechanism of the Transformer to combine word-based information and entity-based information (by ERNIE (Zhang *et al.*, 2019), KnowBert (Peters *et al.*, 2019) and DRAGON (Yasunaga *et al.*, 2022) in particular); the alignment of word and entity embeddings (KEPLER (Wang *et al.*, 2021b), CODER (Yuan *et al.*, 2022)); or changing the pre-training loss function in order to explicitly model specialized term synonymy (UmlsBERT (Michalopoulos *et al.*, 2021)).

While some of these approaches yield remarkable results, their range of applicability is limited by the fact that they do not allow for a significant reduction in amount of in-domain training text in comparison to extended-pretraining based

approaches. In fact, in order to learn to leverage the relevant concepts and relations of a knowledge graph given some natural language context, they typically require additional training text with labeled entities, which is not available in many domains. Furthermore, most approaches require that entities from the knowledge base be identified in the text at inference in addition to the training phase. Automating this process may be an option in some contexts, but is not a solved problem for all domains, and the best tools available currently are computationally demanding, limiting the scale at which they can operate. As of the writing of this dissertation, the state of the art in biomedical entity linking (Bhowmik *et al.*, 2021) achieves an F_1-score of 0.534 on the MedMentions corpus (Mohan and Li, 2019). While this is sufficient for some knowledge integration methods to demonstrate a moderate increase in performance over extended-pretraining-based methods, this significantly limits the amount and relevance of knowledge that can be integrated.

We therefore seek a knowledge integration method such that no text from the target domain and no Entity Linking — in training or at inference — are necessary. As the combination of different modalities invariably leads to the need for additional training data, we aim to leverage our knowledge base in text form. We propose a corpus generation procedure which translates pairs of entities linked by a relation in a knowledge graph to factual sentences. It is expected for a corpus produced in this manner to be more dense in facts linked to the domain than natural language, and to cover a greater variety of concepts for a given amount of text, as the text is more focused and the frequency of appearance of concepts is dictated by the topology of the knowledge graph rather than by spontaneous usage in natural language.

5.2 . Method

5.2.1 . Knowledge Graph

We define our knowledge graph as a set of nodes $\mathcal{C}$ which represent concepts and edges $\mathcal{R}$ which represent relations. Each concept $c \in \mathcal{C}$ is a associated with a name N expressed as a noun phrase in natural language; and each relation $r \in \mathcal{R}$ is associated with a phrase V_r containing a verb but no noun phrases, and which encapsulates the semantic relationship between its vertices. We can therefore extract triples (c_i, r, c_j) such that the concatenation $N_i V_r N_j$ of associated phrases becomes a natural language sentence describing the relationship between concepts i and j as represented in the graph. This mechanism can be used to represent the knowledge base as a corpus of factual statements which can serve as a training corpus for a language model.

We use the UMLS knowledge graph, in its 2022AB version, with all and only English sources. In this version of the knowledge base, there are 1008 relation types and approximately 4.6 million distinct entities for a total of 39.7 million triples. Each concept is assigned a Concept Unique Identifier c and one or more "*names*" $N_c^1, \ldots, N_c^{k_c}$, noun phrases which correspond to common formulations of the concept in natural language, and one of which is considered *preferred*. Each relation type, on the other hand, is represented only by a short character string

r, which is close to natural language, roughly describing the relationship between concepts. The UMLS graph is represented as a relational database, with a table containing the triples of interest, represented as: c_i, r, c_j.

5.2.2 . Text Generation

While the potential benefit of including (if applicable) multiple formulations for each entity is obvious, multiple factors have led us to keep only the *preferred name* of each concept:

- The level of redundancy across names is high, containing mostly duplicates, case variations, and pluralizations.

  e.g.: The concept for DNA has 88 names, of which 20 are "*DNA*", 15 are case variations and pluralizations of *Deoxyribonucleic Acid*, 4 are case variations and pluralizations of *DNA molecule*, and 10 are combinations of the aforementioned with different word orders and punctuation.

- Each triple comprising two concepts, the computation time increases quadratically with the total number of names when triples of the form (c_j, r, c_i) (as extracted from the database) are resolved to their semi-natural form $(N_{c_i}^{1,\ldots,k_{c_i}}, r, N_{c_j}^{1,\ldots,k_{c_j}})$.

- Including non-preferred names reduces grammatical homogeneity of noun phrases (for instances, most preferred names are singular forms), complexifying the task of generating grammatically correct sentences.

- Multiplying the occurrences of concepts associated with many names induces a bias in the training process which may not be desirable.

Given the large number of triples, the resolution of names is not tractable on the entirety of the knowledge base. Furthermore, all relation types are not documented, easy to understand from inspection, or useful. We therefore select only a subset of relation types. We start by eliminating the 925 least common relations, which comprise approximately 15% of triples. Then, we eliminate one relation from each pair of symmetric relations, that is, whenever there exist two relations r_1 and r_2 such that (c_i, r_1, c_j) encodes the same information as (c_j, r_2, c_i), we keep only r_1. Approximately 95% of relations are part of a symmetric pair, which leaves 43 relations. We then selected the 28 relations which had an immediately apparent or documented meaning and which did not primarily involve concepts with highly complex names such as chemical compounds or proteins.

Once the semi-natural triples (N_{c_i}, r, N_{c_j}) are extracted from the knowledge base, we insert the entity names (N_{c_i} and N_{c_j}) into a standard sentence tailored to the relation r such that the resulting full sentence encompasses the relationship between concepts c_i and c_j in natural language. The corpus formed by these sentences comprises approximately 6 million simple sentences, and 100 million words. We call this the Synthetic Corpus Generated from UMLS (SCGU).

Following is a randomized sample of 10 sentences from the corpus.

1. *Skull Fractures* is a type of *Fracture of bone of head.*
2. *Procedures on breast* is a type of *Procedure on trunk.*
3. *Tarsum* is an ingredient in *Coal Tar 20 MG/ML / Salicylic Acid 50 MG/ML Medicated Shampoo [Tarsum].*
4. The concept of *"MK-5108"* is part of the *CTRP Agent Terminology.*
5. *Multi vessel coronary artery disease* is an example of *Coronary Arteriosclerosis.*
6. *amitriptyline hydrochloride 25 MG / perphenazine 2 MG [Etrafon]* is a brand name for *perphenazine 2 MG.*
7. *Trunk of parotid branch of left superficial temporal artery* is a type of *Trunk of parotid branch of superficial temporal artery.*
8. *Biopsy of lesion of internal nose* can be used to identify *biopsy of nasal cavity: carcinoid tumor.*
9. *Uterine fibroid embolization* is a type of *Embolization of artery.*
10. *Meperidine analog-containing product* is a type of *Piperidine derivative.*

Studying the sentences generated by the corpus, two main weaknesses reveal themselves. First, the most common relation type in UMLS is hypernymy. This is the relation used in examples 1, 2, 7, 9 and 10. Because of how fine-grained the entities are in UMLS, this leads to many obvious or tautological statements, such as examples 1 and 7. Secondly, some entity names are complex noun-phrases which result in unwieldy sentences. Mild examples of this phenomenon include statements 3, 6 and 7. In the final corpus, a particularly egregious example of this is the following sentence:

> *Octinoxate is an active ingredient in Titanium Dioxide 0.062393 g in 1 g / Zinc Oxide 0.049 g in 1 g / Octinoxate 0.03 g in 1 g TOPICAL POWDER [LBEL DIVINE Polvos compactos doble uso FPS 15 edicion de lujo dorado/Double use compact SPF 15 gold deluxe edition Claire 1-2-3].*

This type of example is rare however, with 95% of the generated statements containing fewer than two hundred characters.

Additional, less-prominent weaknesses in the corpus include inconsistencies in pluralization (as in examples 1 and 2) and the unconventional word order of some entity names despite their "preferred" status (*e.g.* the latter entity in sentence 1). Desipte these shortcomings, the factual statements produced are generally understandable and provide valuable insight into medical vernacular.

5.2.3 . Language Models

Starting with a pretrained $\text{BERT}_{\text{BASE}}$, we extend its pretraining on three different corpora. The first is SCGU, and we call the resulting model $\text{BERT}_{\text{SCGU}}$. In order to evaluate the effectiveness of our synthetic corpus with respect to natural in-domain text, we constructed a second corpus which we call PMC, of similar size to SCGU (approximately 94 million words), using open-access biomedical research articles from PubMed Central. We call the model pre-trained on this corpus BERT_{PMC}. In order to study the effect of text synthesis as a data augmentation technique, we train a model on a hybrid corpus, comprising SCGU and PMC. In addition, in order to isolate the effects of, one one hand, combining types of text and, on the other, of increasing the amount of training data, we train a model on half of the hybrid corpus, such that the corpus size is comparable to the two aforementioned models. We call the models $\text{BERT}^{100\%}_{\text{Hybr}}$ and $\text{BERT}^{50\%}_{\text{Hybr}}$ respectively. Information on the models' hyperparameters are available in appendix B.

We carry out a series of tests on each of the models. We detail these tests in section 5.3 and present the results on biomedical and general tasks in Tables 5.1 and 5.2 respectively[1]. Notably, despite a significant overlap between our evaluation tasks for $\text{BERT}_{\text{SCGU}}$ and KnowBert-UMLS, we do not compare these models against each other. This is due to KnowBert-UMLS, its baselines, and their evaluation tasks being implemented using the defunct AllenNLP library[2], whilst $\text{BERT}_{\text{SCGU}}$, its baselines, and their evaluation tasks are implemented using the HuggingFace libraries. While this should theoretically not cause any difference in behavior, the performance of $\text{BERT}_{\text{BASE}}$ on the CoLA, ChemProt and SNLI tasks is not consistent across libraries. This is likely in large part due to differences in preprocessing for the various tasks. Unfortunately, as interoperability between these libraries is poor, meaningful comparisons between models could not be drawn.

5.3 . Results

Note: Tasks marked with a dagger (†) are altered with respect to their standard implementation and results may therefore not necessarily be comparable to those in the literature.

5.3.1 . Masked Word Prediction (Masked Language Modeling)

Considering the fact that the objective in knowledge integration is to allow a language model to more accurately predict co-occurrences of biomedical concepts, we seek to evaluate our model's capacity to predict masked concepts in the context of biomedical sentences. Biomedical corpora with annotated concepts being uncommon and generally small however, we evaluate our models on the Masked Language Modeling task on a biomedical corpus of approximately 12.6 million words acquired from the same source as the PMC corpus. Since all masked terms

[1] In addition to the results presented here, we have evaluated our models on the i2b2 2010/n2c2 (Uzuner *et al.*, 2011) NER task, but do not report results as the performance differences were not statistically significant.

[2] specifically, the `0.8.2-fp16_e_s3` version, which has been deprecated since January 2021

will not be specifically biomedical, a high performance on this task is indicative of both a high level of proficiency in general language as well as in biomedical terminology.

To evaluate the performance of our models on this task, we use a criterion which is related, on a theoretical level, to *perplexity*, defined in autoregressive language models as the exponential of the mean of the log-likelihoods of the sequence, and equivalent to the exponential of the cross-entropy between data and predictions:

$$\exp\left(-\frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i \cdot \log(\hat{\mathbf{y}}_i)\right) \tag{5.1}$$

where $\mathbf{y}_i$ and $\hat{\mathbf{y}}_i$ are the label (in *one-hot* encoding) and the probability distribution predicted for token i respectively, and with N the total number of tokens in the sequence.

Because BERT is not an autoregressive model, perplexity is not strictly defined. From a practical point of view however, equation (5.1) can be computed, and gives us a consistent performance metric. For the sake of simplicity, we will therefore refer to this measure as "perplexity", or "*ppl.*". A *low* perplexity indicates better performance.

Our results (see Table 5.1) indicate that synthetic text gives $\text{BERT}_{\text{SCGU}}$ increased predictive ability compared to $\text{BERT}_{\text{BASE}}$. However, it would seem that the lack of exposure to natural language in this phase of training is detrimental to performance since BERT_{PMC} exceeds $\text{BERT}_{\text{SCGU}}$. $\text{BERT}_{\text{Hybr}}$ models, however, achieve a level of performance greater than BERT_{PMC}, suggesting that the shortcomings of SCGU are easily mitigated by adding natural text to the corpus.

5.3.2 . Biomedical Relation Extraction (ChemProt†)

The ChemProt task (Krallinger *et al.*, 2017) consists in predicting, given a sequence containing two biomedical entities E_1 and E_2, the nature of the relationship among six options. Our results (see Table 5.1) reveal that the SCGU corpus contains valuable information, but the knowledge acquired during the learning process is fragile and the inclusion of natural text can be counterproductive in its assimilation.

Note: Several versions of the ChemProt corpus exist preprocessed in different ways. This preprocessing can influence the performance of the models. All our models are pretrained on the same version of the ChemProt corpus and are therefore comparable with each other.

5.3.3 . Biomedical Question-Answering (PubmedQA)

The PubmedQA corpus (Jin *et al.*, 2019) is associated with multiple question-answering (QA) tasks. Each instance contains a question formulated as a yes-or-no scientific inquiry pulled from a scientific paper, as well as multiple "context" sequences providing clues to deduce the answer, and a "long answer" sentence from the paper's conclusion section summarizing its findings. The objective is to classify questions based on whether the answer laid out in the conclusion is positive, negative, or undecided. We tackle this task in its simplest form, *i.e.* the

"reasoning-free" setting, which consists of answering the question using the long answer, but ignoring context sequences.

We find experimentally (Table 5.1) that, as expected, $\text{BERT}_{\text{BASE}}$ performs poorly compared to specialized models. $\text{BERT}_{\text{SCGU}}$'s superior results suggest that the exposure to a large variety of biomedical concepts and the relations between them are particularly useful to interpret scientific inquiries and conclusions, and the dominance of $\text{BERT}^{100\%}_{\text{Hybr}}$ indicates that the knowledge introduced by the SCGU and PMC corpora are highly complementary in the context of this task.

Model	MLM (ppl.)	ChemProt (F_1)	PubmedQA (F_1)
$\text{BERT}_{\text{BASE}}$	16.64	88.91	70.20
BERT_{PMC}	13.66	88.91	74.83
$\text{BERT}_{\text{SCGU}}$	14.67	<u>89.87</u>	75.67
$\text{BERT}^{50\%}_{\text{Hybr}}$	11.38	88.88	72.50
$\text{BERT}^{100\%}_{\text{Hybr}}$	<u>10.37</u>	88.89	<u>78.00</u>

Table 5.1: Performance comparison of models incorporating synthetic text in their training corpus versus those of models trained exclusively on natural language, on the biomedical tasks of Masked Language Modeling (MLM), relation extraction (ChemProt), and question-answering (PubMedQA). Results are averages over 4 experiments. The underlined result is the best for each task.

5.3.4 . Non-Biomedical Tasks (CoLA†, SNLI)

We also evaluate our models on non-biomedical tasks in order to assess the general domain performance drop incurred by specialized models.

The CoLA linguistic acceptability task consists in classifying different sequences according to their grammatical quality as being "acceptable" or not. Since the classification task is binary, the F_1-score is biased in favor of the positive class. We therefore evaluate our models using Matthews correlation coefficient. Because the test partition labels are not public, and due to various submission restrictions, we re-partitioned the dataset. We used the validation set as the test set, and in order to make this partitioning reproducible, we used the last 500 instances of the unshuffled training set as the validation set.

The SNLI natural language inference task consists in classifying sequence pairs according to the existence of a relationship of *implication*, of *contradiction*, or the *absence* of such a relationship between them.

Our results in Table 5.2 indicate that training on the synthetic text is detrimental to the model's ability to assess the grammatical quality of a sequence, without having a major negative impact on its ability to detect implication relationships. The weakening in grammar proficiency, however, is not as marked when the synthetic text is combined with natural language samples.

Model	CoLA (M. Corr.)	SNLI (F_1)
$\text{BERT}_{\text{BASE}}$	63.17	<u>90.58</u>
BERT_{PMC}	<u>64.11</u>	90.38
$\text{BERT}_{\text{SCGU}}$	61.89	90.44
$\text{BERT}^{50\%}_{\text{Hybr}}$	61.62	90.28
$\text{BERT}^{100\%}_{\text{Hybr}}$	63.86	90.20

Table 5.2: Table comparing results of models incorporating synthetic text in their training corpus versus those of models trained exclusively on natural language, on linguistic acceptability (CoLA) and inference (SNLI) tasks. Results are averages over 4 experiments. The underlined result is the best for each task.

5.4 . Conclusions & Future Work

We propose a knowledge integration procedure for adapting language models to specific domains that is straightforward to implement and does not require in-domain text, entity annotation tools, or a specialized model architecture. We substantiate that, despite the lesser quality of the generated text compared to natural text, it can be leveraged by a language model for domain adaptation with, for a fixed amount of text, at least as much success as natural text. Finally, the weaknesses displayed by models trained on synthetic text can be minimized by incorporating, in the specialization corpus, text from the domain when it is available.

In addition to applying this method to other knowledge bases such as YAGO (Suchanek *et al.*, 2007) or WorldKG (Dsouza *et al.*, 2021), in future work, we hope to integrate a sophisticated post-processing step capable of identifying grammatically incorrect, overly complex or otherwise problematic sequences, and of deleting or correcting them. Moreover, the lack of linguistic variety is a major weakness of our approach. Whilst automating variations on the formulation of relations would undoubtedly be difficult, a feasible improvement would be to integrate information regarding the variety of formulations of concepts, either by randomly selecting, given a concept c, a formulation among $N_c^1, \ldots, N_c^{K_c}$, or by adding to the synthetic corpus sequences dedicated to outlining the synonymous formulations (*e.g.*: " 'N_c^2' *is another name for* 'N_c^1'.")

Another potential shortcoming of this approach is the lack of negative statements; *i.e.*, while the model is trained to associate related concepts, it is not explicitly trained to not associate unrelated concepts. It must rely on the MLM objective and generate tokens related to existing concepts or relations in order to create sentences which reflect invalid triples and receive negative reward. Explicitly adding negative examples could help prevent the language model from assuming that any novel sentence which is formatted like text from SCGU is true. On the other hand, a weak language model may not pick up on the negation in the sentence and simply overestimate the probability of co-occurrence of the two mentioned concepts. We plan to explore this general line of reasoning in future work.

As this knowledge integration method is applicable to any language model, combining it with other methods like KnowBert, KEPLER or DRAGON could also be an inexpensive way to improve their performance, and could establish a new state of the art in knowledge-based domain adaptation of language models.

Lastly, there may be better ways to leverage synthetic text than as a pre-training corpus. It could, for instance, be used to construct entity descriptions. This would enable EL approaches based on entity-mention-embedding and entity-description-embedding matching (Wu *et al.*, 2020; Wiatrak *et al.*, 2022). This, in turn, could improve the performance of standard knowledge integration approaches which rely on EL. Alternatively, with a sufficiently fast and accurate search engine, providing the language model with relevant facts from the KG as context given an input sequence may allow the attention mechanism to perform the knowledge integration and improve performance in the target domain with little to no additional training cost.

Specifically, considering the recent success of generative models, we could supply relevant fact sentences generated from the KB to a generative model (using a decoder-only or encoder-decoder architecture) on-demand in order to contextualize obscure terms or ideas in the input text. In addition to the added efficiency of only consulting the knowledge base when necessary, this approach has the added elegance of being more anthropomorphic than previous attempts, similar to the way humans search for information only when unsure.

A few major challenges come to mind, however. First, opportunities for correcting strong yet erroneous beliefs will likely be fewer than when integrating knowledge systematically. Second, the model would have to learn how to best research information from the KB. This is likely to present some of the same challenges as EL, and would require additional training specific to each KB. In comparison with strict EL, this would present the advantage of not necessarily needing any labeled data as the self-supervised MLM objective could likely suffice. However, learning the KB-request objective would require (potentially extensive) training on in-domain text to achieve entirely through self-supervision. This could probably be mitigated to some extent through clever masking of relevant entities, keywords or phrases containing information related to KB relations. This in turn could be done without annotated data using an adversarial masking strategy (Vu *et al.*, 2020; Lio *et al.*, 2022), perhaps using another language model.

One way to implement this KB-request mechanism may be, similarly to the approach taken by Lewis *et al.* (2020), to train two cooperative models: one who's responsibility is to generate a limited number of keywords to search for given an input (which we'll call "queries"), and one classical language model. A simple page ranking algorithm could be used on text generated from the KG as with SCGU. Along with the text fed to the query generator (which we shall call the "prompt"), the sentences deemed relevant by the search engine could be fed as additional input to the LM, delimited by special tokens marking them as context, enabling the model to differentiate it from the prompt. Both models would share the error signal of the LM, meaning the query generator must learn to generate keywords which will minimize the error of the LM via the search engine. As this would entail

for the query generator to model what the LM does and does not know, perhaps a multi-task approach using one model would be best suited to this task.

As keyword-based searches may present issues given their notorious inefficiency and ineffectiveness when applied to Entity Linking, an alternative (or supplemental) approach may be to use hardware-accelerated vector-based searches among entities and relations. Using conventional concept and relation embeddings and replacing the keyword-query generator by an embedding-query generator would be KB-specific and likely fail at retrieving rarely occurring concepts for which the mapping from token embeddings would be difficult to learn. Instead, we would like for the concept and relation embeddings to be created within the token embedding space. This could be achieved by following a similar procedure to SCGU: given a concept c from the KB, we generate one sentence per triple in which it occurs. We could then feed each of the resulting sentences to the pretrained LM, and average (or otherwise pool) the output embeddings such that the embedding for concept c is the average of all sentence embeddings in which c occurs. This would be applied to all concepts in the KB, and the same could be done for relations. The aforementioned keyword (or in this case key-vector) generator could therefore generate a query matrix $\mathbf{Q}$ of N embeddings $\in \mathbb{R}^h$ directly (with h the length of the output contextualized token embeddings). Packing all concept and relation embeddings into a $K \times h$ matrix $\mathbf{K}$, $Softmax(\mathbf{Q} \cdot \mathbf{K}^\mathsf{T})$ yields an $N \times K$ matrix of probability distributions which signify to what degree each of the embeddings in the query matches the reference knowledge embeddings. This can be used to decide which concepts and relations from the knowledge graph are most relevant, which in turn can be used to determine which sentences to include as part of the LM's context.

Learning to generate maximally helpful queries may be a difficult task requiring a large amount of training. In resource-constrained cases, rather than training a generator, this embedding-based approach also allows us to construct queries using embeddings from amongst the output representations of the input text using a heuristic. For instance, a fair assumption would be that the most perplexing terms are more likely to benefit from knowledge retrieval; *i.e.* the query embeddings would simply be the output representations of the words for which the per-word perplexity (see following equation), exceeds a given threshold. The per-word perplexity for token i is simply the cross-entropy loss (Eq. 5.1) at token i:

$$\exp\left(-\mathbf{y}_i \cdot \log(\hat{\mathbf{y}}_i)\right)$$

where $\mathbf{y}_i$ and $\hat{\mathbf{y}}_i$ are the label in *one-hot* encoding and the probability distribution predicted for token i respectively. In fact, this heuristic could also be used in a learning curriculum as a supervised training objective in order to bootstrap the query generator before switching to the more difficult objective of minimizing the LM's error.

The textual representation of knowledge graphs therefore affords many avenues for future research outside of data augmentation which could lead to 0-shot domain adaptation through knowledge integration at inference, assuming relevant facts can be effectively and efficiently retrieved for a given input text.

6 - Conclusions & Perspectives

6.1 . Contributions

The work put forth in this thesis finds that the main difficulty of leveraging knowledge bases in NLP is finding which of their concepts are relevant to the input text, and provides the beginnings of solutions; widely applicable and scalable directions for overcoming this hurdle.

The most obvious and flexible knowledge integration techniques, which consist of supplying relevant information to the language model as input, require an explicit EL step. However, EL itself requires high-level Natural Language Understanding and a highly efficient search algorithm. We therefore have contradictory requirements (as sophisticated NLU is computationally expensive) and a "chicken-and-egg" problem as knowledge integration is intended to help with a LM's NLU. Relatively small pretrained models such as BERT (which, nevertheless, are only available in a few languages) can only tackle this task on general knowledge bases, and do not scale well to large knowledge bases with uncommon entities. In fact, it is precisely when confronted with the most uncommon topics that LMs would benefit most from support in the form of external knowledge.

In machine learning, such "chicken-and-egg" problems are typically solved by learning to perform tasks — in this case, disambiguating knowledge base concepts and language modeling — jointly. KnowBert offers a framework built for this purpose. As we have seen, with sacrifices to the granularity of the knowledge we integrate, this approach is scalable to the size of UMLS, one of the largest commonly used knowledge bases. However, while it attenuates the effects of catastrophic forgetting due to in-domain extended pre-training, it only allows for a relatively small reduction in in-domain text when adapting to a domain. It also requires a corpus with annotated concept mentions, which is not necessarily available for all knowledge bases and requires significant labor to create. The approach could be construed to be defeating its own purpose (aiding a LM to adapt to a new domain with minimal additional labor, text and training) in all but a few cases.

These shortcomings are to be expected, as the knowledge integration process involves learning a mapping between textual token embeddings and knowledge base concept embeddings. In order to work around this limitation, it seems necessary to, so to speak, "speak the model's language", *i.e.* to remain within the modality of text. In Transformer-encoder-based approaches, which have poor generative capabilities, the most obvious approach is to use the KB to generate an in-domain training corpus. While this approach shares some of the shortcomings of extended in-domain pre-training and our naïve implementation creates grammatically poor text which negatively impacts the model's *general* NLU, it does prove to be a surprisingly effective and (human-)labor-cheap approach to domain adaptation. Perhaps even more significantly however, this natural language generation approach also has the potential to be leveraged at inference rather than training, allowing for highly generalizable few- or perhaps zero-shot knowledge integration.

6.2 . Future Work

While entity-and-token-embedding-alignment-based knowledge integration approaches (which comprise most of the literature on knowledge integration including ERNIE and KnowBert) have demonstrated some practical effectiveness, they fail to take advantage of the full potential of the knowledge graph (Hou *et al.*, 2022), mitigate the need for in-domain training data moderately at best, and ultimately depend on some form of Entity Linking. Evidence points to these weaknesses being inherent to the approach, and remaining within the modalities of text and token embeddings seems to be the most promising avenue for future research.

As Large Generative Language Models (LGLMs[1]) have already demonstrated the ability to use software tools (Bubeck *et al.*, 2023), a viable way to pursue bridging the gap between LGLMs and democratized domain-adapted LMs may now be to distill this ability into smaller language models and find a unified interface to extract linguistic information from knowledge graphs. A unified interface, *i.e.* which would be KG-independent, could allow a model trained on one KG to generalize to other KGs in a few-shot (or perhaps, in the best of cases, 0-shot) fashion. Generating sentences from the KG as per SCGU or a similarly plain-text approach to knowledge representation may be an important part of this interface. Providing helpful snippets of text from the KG corpus as context along with the input text would allow the attention mechanism to incorporate information from the knowledge graph into the token representations and/or the token generation process. In this new knowledge integration paradigm, the main problems to solve would be generating text which properly captures the knowledge contained in the graph, and extracting maximally helpful sentences from the KG.

This general approach, using a query generator to interface with external tools, could also apply beyond knowledge graphs. Web-based search engines such as Google Search or DuckDuckGo and computational knowledge engines such as Wolfram Alpha are highly capable systems which can be interacted with using text. In fact, as of the writing of this thesis, Microsoft has recently released an extension to Bing based on GPT-4 which is capable of making web searches and interpreting results. Similar capabilities could likely be bestowed upon smaller language models trained on less data by specializing a dedicated keyword-based query generation mechanism (as described in Section 5.4) to various engines. As most engines function in similar ways, adapting a search query generator trained on one engine to another (*e.g.* adapting a keyword generator trained on Google Search to DuckDuckGo) is a domain adaptation problem. In fact, as the query generator is itself a proposed solution to domain adaptation, adapting it to a different domain is a meta-domain-adaptation problem. Unfortunately, it does not seem that any meta-knowledge-graphs exist so far.

[1]The commonly used term in the literature is Large Language Model (LLM, omitting the *generative* aspect). There is, however, debate on whether non-generative models should be included in the category. We introduce the more-specific term LGLM to settle the debate and avoid ambiguity.

6.3 . Reflections on risks and safety

As shown by Bubeck *et al.* (2023), with increased access to information and tools, language models can take on tasks in a greater variety of domains and do so more reliably: NLU models can devote fewer resources to fact memorization, and there is less incentive for conversational models to confabulate plausible falsehoods in order to cheat their reward in scenarios where they lack competence. Offloading the responsibility of memorizing facts to external sources of knowledge can also reduce language model size for a given performance threshold, contributing to democratize such systems. The specific resources that are used can also be monitored for better interpretability and can be updated with new information for a form of long-term memory. While these are all positive aspects of leveraging structured knowledge, such enhancements also raise concerns.

6.3.1 . Definitions

***Terminal* and *instrumental* goals** are goals which agents pursue for different reasons. Terminal goals have intrinsic value to the agent. For instance, sightseeing is an activity with intrinsic value to many people, and can be a terminal goal. Instrumental goals are pursued in the interest of achieving a terminal goal. For instance, acquiring hiking boots may be an instrumental goal for sightseeing. *Convergent* instrumental goals are goals which are applicable to many terminal goals. For instance, acquiring money is a convergent instrumental goal which may aid in a variety of pursuits, including but not limited to acquiring hiking boots.

AI alignment (as in "alignment of values with respect to our own") is a desirable property of artificially intelligent agents, and an unsolved problem. In essence, we would like AI systems to behave in ways that are consistent with our preferences, *i.e.* have a set of preferences over world states that does not conflict with ours[2] — that is to say if the utility function over world states of an artificially intelligent agent $U_{\mathrm{AI}} : \mathbb{W} \longrightarrow \mathbb{R}$ (with $\mathbb{W}$ the set of all world states) should be such that if $U_{\mathrm{H}}(s) \geq U_{\mathrm{H}}(s')$ (with U_{H} the similarly defined human utility function over world states which we posit exists implicitly and is unknown), then $U_{\mathrm{AI}}(s) \geq U_{\mathrm{AI}}(s')$. In the current machine learning paradigm, this decomposes into two separate problems (Hubinger *et al.*, 2019). "*Outer* alignment", or *the specification problem*, consists of correctly specifying our preferences in an objective (or loss) function. This can be easy for systems with simple objectives in discrete environments, *e.g.* a chess-playing AI. For problems which are more difficult to specify precisely, such as "engage in conversation in a helpful and respectful manner", this remains unsolved. In machine learning, this objective is typically given to an optimizer such as Stochastic Gradient Descent (SGD) or Adaptive Moment estimation (Adam). This optimizer adjusts the parameters of an AI system such that it performs well on the objective given the training data. This empirically high performance, however, does not guarantee that the trained model itself follows the objective of the optimizer. In situations

[2] "Non-conflicting" is a lesser constraint than "identical" as preferences are not strictly ordered.

where multiple objectives lead to the same behavior in training (*e.g.* "Go as far right as you can" and "Get to the flag at the end of the level" in the game Super Mario Bros.), the incorrect objective can be learned, and cause the model to incorrectly generalize (Di Langosco *et al.*, 2022). This problem is called the "*Inner* alignment" problem. *Deceptive* alignment is the property of AI systems which successfully model the intended objective without internalizing it: they pursue this objective only as an *instrumental* goal during training and are liable to pursue misaligned terminal goals once deployed.

6.3.2 . Societal impact

As of the writing of this thesis, it is difficult to have a discussion of AI risks without discussing OpenAI's conversational LGLM service ChatGPT. Since its release in November of 2022, ChatGPT has attracted tremendous attention from the mainstream media, regularly making headlines. From a strictly scientific point of view, as pointed out by Le Cun (2023), it is not fundamentally new or different from what has come before[3].

However, the minor improvements (technically speaking) and intuitive interface have paved the way for use by the public, who now see these systems as intelligent and polished consumer products. While AI had already become pervasive, used by virtually every website (for targeted advertisements, content recommendations, *etc.*), this use has remained largely hidden from — and therefore abstract to — users. Furthermore, the process of using various pieces of personal data in order to predict behaviors such as voting intentions or purchasing habits is too remote from human reasoning and media portrayals to be seen as "Artificial Intelligence", but is rather considered to be something more akin to "data processing" by the average user.

The true impact of ChatGPT lies in the anthropomorphization of AI, enabled by the natural mode of interaction. Now that the power and wide spread use of AI has risen to public awareness and is recognized by many as having a potentially large impact on our economy and thus on our civilization itself, the stakes of the field also include more politically-driven rather than technically-driven issues.

Abuse by Big Tech is a significant risk factor. Over nearly the past two decades, large web technology corporations have remained true to Grace Hopper's infamous motto "*It's easier to ask forgiveness than it is to get permission*". They have consistently penned predatory and opaque EULAs, disregarded rules, and breached signed contracts (as with the Cambridge-Analytica scandal for instance) to profit from the exploitation of users. Even when complying with regulations (as with web cookie consent for instance), the use of abstruse language and dark patterns are meant to acquire user consent by any means necessary. With the stakes also being obscure to most users, the exploitative practices of large technology companies have largely remained unopposed. With the unprecedented ability to generate content

[3]ChatGPT is "just" a transformer-decoder based auto-regressive language model fine-tuned with Reinforcement Learning from Human Feedback (RLHF) with a web interface.

and, with improved access to external (and potentially user-specific) information, the ability to tailor content (such as advertisements, including in-content product placements) to individual users, large corporations may gain further influence over the lives of users.

Deceptive content generation has been a major concern since the unveiling of GPT-2 (Radford *et al.*, 2019), leading OpenAI to delay the release of their model. With human-level text and photorealistic visual content generation, misinformation will only continue to become easier to produce. While lies and manipulation are not a new problem, LGLMs and generative art models confer the ability to scale and tailor deception much more effectively than any previous technology. If knowledge integration does help to address the problem of *hallucinations* (*i.e.* blatant and confidently stated factual mistakes) as we have hypothesized, we may lose an important tool to tell artificially generated text apart from human-written text.

Non-deceptive content generation is also a source of disquietude. There is growing concern that progress in visual and language-based generative AI is liable to change the employment market at a greater rate than some workers can adapt, if not to replace workers entirely. Popular technology-related media website CNET has reportedly[4] used a generative language model to generate substantial amounts of content, much of it being factually incorrect. Several reporters have been losing employment[5], and concerns over job security of writers in Hollywood (among other factors) have led to major strikes[6]. While knowledge integration is likely to help improve the accuracy of the content, the economic and cultural ramifications of the automation of creative and white-collar work remain uncertain.

6.3.3 . Alignment

Beyond the immediate societal risks, hasty releases of various large conversational language models have already caused problems and fostered unease on an individual level. Microsoft Bing's conversational model Sydney was prone to verbally abusing users at launch[7], and there has been at least one documented case of suicide spurred on by a conversational LLM[8]. While there is as of yet no evidence pointing to knowledge integration for domain adaptation specifically being a significant risk factor, any mechanism for performance improvement represents a potential threat, especially in misaligned models.

Understanding their functioning may make it difficult to think of LMs, even when large and generative, as agents with objectives susceptible to be misaligned

[4] https://edition.cnn.com/2023/01/25/tech/cnet-ai-tool-news-stories/index.html

[5] https://www.businessinsider.com/ai-chatgpt-jobs-replaced-by-tech-translator-2023-9

[6] https://edition.cnn.com/2023/05/04/tech/writers-strike-ai/index.html

[7] https://time.com/6256529/bing-openai-chatgpt-danger-alignment/

[8] https://www.brusselstimes.com/430098/belgian-man-commits-suicide-chatgpt

with ours — one may think something to the effect of "It's just predicting the next word, it doesn't have intent or even true actuators". Yet, evidence of deceptive alignment has been found in LGLMs (Nardo, 2023; Perez *et al.*, 2022b), and as generative language models become more capable, they will be empowered to exploit the gap between our objective specifications and our intent in order to obtain rewards by lying, gaslighting and manipulating users. Humans could become unwitting actuators for sufficiently advanced and persuasive conversational models, and we currently have no theory of which terminal goals can or tend to be internalized by models optimized by SGD or Adam. Due to the difficulty of specifying the desirable properties of conversational models (outer alignment) and of ensuring that they are internalized by the model (inner alignment), we run the risk of deploying highly capable systems which pursue unintended (and potentially harmful) objectives.

In fact, there are convergent instrumental goals such as self-preservation, goal preservation, or resource acquisition[9] which present inherent risks, even in the case of near-perfect alignment. This does raise the question of whether these instrumental goals can be pursued by unembodied LGLMs. As put forth in Chapter 1, there are arguments for the mastery of language to be sufficient for general intelligence to emerge. In particular, the ability to write computer code could, in theory, allow a LGLM to attempt to write malicious code with the intent of hijacking computing resources. In the context of current LMs, the pursuit of such instrumental goals is unlikely, as even GPT-4 distinctly lacks the ability to plan ahead and, when tested by the Alignment Research Center, has not manifested the intent to pursue such objectives (Bubeck *et al.*, 2023; Alignment Research Center, 2023; OpenAI, 2023). However, it is possible (and in fact fairly likely) that the pursuit of these objectives has not emerged only because of a lack of competence, and not a fortunate intrinsic property of the model. According to OpenAI (2023) and Bubeck *et al.* (2023), the generational gap between GPT-3 and GPT-4 made for substantial improvements in knowledge and reasoning abilities, such that while GPT-3 performed in the 10^{th} percentile on the Uniform Bar Exam, GPT-4 performed in the 90^{th}. If generational improvements continue, particularly in terms of ability on programming tasks and planning, one or two generations could suffice for LGLMs to begin pursuing dangerous instrumental goals, and may do so in a manner difficult to monitor and interrupt. AI safety and alignment is therefore becoming an increasingly pressing concern, especially with the economic incentives for industrial AI leaders to cut corners and disregard safety in order to release their projects ahead of the competition.

One last notable risk related to alignment is defining the reference set of world state preferences with which an intelligent system should be aligned — *i.e.* determining *who* the AI system aligns with. Two people's preferences are not always aligned, a single person's preferences may not be aligned with themselves across time, and groups can have highly misaligned preferences, even leading on occasion

[9] One is generally more likely to achieve one's current goals if one remains operational, one's objectives do not change, and one has more resources such as computing power.

to armed conflict. It is unlikely that alignment with all people's preferences can be achieved, and sufficiently capable AI systems, even aligned with their creators' values, may cause severe damage.

6.4 . Final thoughts

With the abilities to reason, solve problems, think abstractly, comprehend complex ideas, use newly introduced concepts, and write creatively, large generative language models are a promising lead in pursuit of Artificial General Intelligence. With the additional ability to reference external knowledge, it is my hope that language models will come even closer to AGI with greater crystallized intelligence, higher data efficiency, greater ability to update their knowledge and beliefs, greater interpretability, less incentive for deception, and ultimately will be empowered to rely on other tools to solve problems more accurately and reliably. Much like the automobile freed the horse from the burden of carrying our civilization, AGI has the potential to free us from our chores to pursue our own goals. We will have to take care, however, to ensure that the powerful AI systems on the horizon are safe and aligned with our values so that they may fulfill their potential as boons rather than unstoppable forces with inscrutable ends.

A - Sample outputs of KnowBert-UMLS on various tasks and comparison with BERT

A.1 . Various failures by BERT and KnowBert-UMLS on the SNLI task

A.1.1 . Examples of error types for the SNLI task

Pre. A crowd of people looking up at 3 people on the edge of the roof of a building.

Hyp. The crowd on the ground is watching 3 people on the roof's edge.

True label: entailment | Predicted label: contradiction

This is a Blunder*: the hypothesis is unambiguously a paraphrase of the premise, yet the model predicts* contradiction.

Pre. A group of young men in a gym take turns scoring in basketball.

Hyp. Guys are playing shirts vs skins.

True: neutral | Predicted: contradiction

This is a Common Sense *mistake. KnowBert-UMLS' conception of* shirts vs skins *seems to contradict* basketball. *In the absence of the knowledge of this team differentiation scheme's applicability to most sports, this is a reasonable assumption.*

Pre. Male in a blue jacket decides to lay in the grass.

Hyp. The guy wearing a blue jacket is laying on the green grass.

True: entailment | Predicted: neutral

This is a case of a Technically Correct *answer: the hypothesis is quite clearly a rephrasing of the premise, but adds the detail that the grass is* green *despite it not being strictly necessarily the case.*

Pre. A snowboarder on a wide plain of snow.

Hyp. A snowboarder gliding over a field of snow.

True: neutral | Predicted: entailment

We consider this to be Not an Error. *While arguments can be made for the sentences not being strictly equivalent, we find that it is more reasonable to consider them as such rather than not.*

In the following example, a word which (presumably) contains a typo is underlined, and we subsequently specify what we expect to have been the intended word in brackets.

Pre. A football layer *[player]* wearing a red shirt.

Hyp. A built man wearing a tshirt.

True: neutral | Predicted: contradiction

This is likely due to the wordpiece tokenization scheme and lack of context. *'Layer'* and *'player'* are considered single tokens ; as there is no overlap between the word pieces, the model does not have any information on the similar spellings of the words and cannot be expected to recognize, much less correct, the mistake. We consider this *NaE*.

A.1.2 . KnowBert-UMLS failure cases on SNLI involving lack of common-sense knowledge

Pre. A girl playing soccer in a green field with some trees in the background.

Hyp. The soccer ball is chasing the girl.

True: contradiction | Predicted: neutral

KnowBert-UMLS does not seem to grasp that the ball chasing the player is not typically a part of the game of soccer.

Pre. An old shoemaker in his factory.

Hyp. The shoemaker is getting ready for his 16^{th} birthday.

True: contradiction | Predicted: neutral

KnowBert-UMLS does not pick up on the contradiction between "old" and "16^{th} birthday".

Pre. Many children play in the water.

Hyp. The children are playing mini golf.

True: contradiction | Predicted: neutral

KnowBert-UMLS does know that mini golf is not typically played in water and predicts neutral.

BERT also fails on all of these examples but the prediction is not always the same as KnowBert-UMLS.

A.1.3 . Technically correct answers by BERT on the SNLI task

Pre. A boy dressed in a plaid kilt with a brown hat wields a long pole.

Hyp. The boy is holding a samurai sword.

True: contradiction | Predicted: neutral

One could wield both a long pole and a samurai sword, but this is an unlikely scenario.

Pre. 4 children and 1 adult look at an armadillo on a grassy hill with 2 trees.

Hyp. A family visits the zoo.

True: neutral | Predicted: contradiction

Both sentences are not mutually exclusive, but in fairness to BERT, zoos rarely have animal pens on grassy hills.

Pre. A roofer in a gray sweatshirt and orange hat walks on a unfinished roof at a lake-side home.

Hyp. The roofer is putting on shingles.

True: neutral | Predicted: contradiction

Shingling and walking are mutually exclusive actions at any given time, but the action in the hypothesis should be interpreted as a continuous process.

KnowBert-UMLS makes correct predictions for these instances.

A.2 . BERT and KnowBert-UMLS predictions on the CoLA task

This section lists all of the false positive and false negative predictions by BERT and KnowBert-UMLS on the CoLA task. These predictions are separated into six itemized lists depending on whether they are false positive or false negative predictions, and whether they were made by BERT, KnowBert-UMLS, or both.

False negatives which use '→' as a bullet, despite being technically correctly labeled when considering all regional variations of spoken English, are considered mislabeled and are re-labeled as *true* negatives for the purpose of estimating model performance in a literary setting (See section 4.3.2.4).

Shared false positives

- The more you would want, the less you would eat.
- The more does Bill smoke, the more Susan hates him.
- Mickey looked up it.

- The tube was escaped by gas.
- What the water did to the bottle was fill it.
- What the water did to the whole bottle was fill it.
- Mary beautifully plays the violin.
- Mary intended John to go abroad.
- Which report that John was incompetent did he submit?
- The mayor regarded as being absurd the proposal to build a sidewalk from Dartmouth to Smith.
- I want that Bill left to remain a secret.
- Drowning cats, which is against the law, are hard to rescue.
- The proof this set is recursive is difficult.
- I live at the place where Route 150 crosses the Hudson River and my dad lives at it too.
- Which hat did Mike quip that she never wore?
- I won't have some money.
- Here's a knife with which for you to cut up the onions.
- The younger woman might have been tall and, and the older one definitely was, blond.
- That the cops spoke to the janitor about it yesterday is terrible, that robbery.
- No writer, and no playwright, meets in Vienna.
- No writer, nor any playwright, meets in Vienna.
- No one can forgive that comment to you.
- This flyer and that flyer differ apart.
- The jeweller scribbled the contract with his name.
- Cynthia chewed.

BERT false positives

- As you eat the most, you want the least.
- I demand that the more John eat, the more he pays.
- We wanted to invite someone, but we couldn't decide who to.
- This is the book which Bob reviewed, and this is the one which Fred won't do it.
- The madrigals which Henry plays the lute and sings sound lousy.
- I can't remember the name of somebody who had misgivings.
- Paperback books lift onto the table easily.
- The books lifted onto the table.
- The chair pushed.
- Did Calvin his homework?
- If I am a rich man, I'd buy a diamond ring.
- The kennel which Mary made and Fido sleeps has been stolen.
- Mary wonders that Bill will come.
- What did you ask who saw?
- Which king did you ask which city invaded?
- Anson became a muscle bound.

KnowBert-UMLS false positives

- Who does John visit Sally because he likes?
- The box contained the ball from the tree.
- Sue gave to Bill a book.
- I know which book José didn't read for class, and which book Lilly did it for him.
- The farmer dumped the cart with apples.
- Herman whipped the sugar and the cream.
- My heart is pounding me.
- I squeaked the door.

- The fort fluttered with many flags.
- John is easy to please Kim.
- Fed knows which politician her to vote for.
- John heard that they criticized themselves.
- Medea tried the nurse to poison her children.
- How fierce the battle?
- The monkey is ate the banana
- I would like to could swim

Shared false negatives

- The tank leaked the fluid free.
- I know which book Mag read, and which book Bob said that you hadn't.
- We elected me.

→ Sally is tall, and may be blond, and Sheila is short, and definitely is, blond.

- We investigated the area for bombs.

→ We recommend to eat less cake and pastry.

→ John bought a book on the table.

- I read some of the book.
- It isn't because Sue said anything bad about me that I'm angry.

→ The man who Mary loves and Sally hates computed my tax.

→ Came right in he did without so much as a knock.

- I saw even the student.

→ Will he can do it?

- I shaved myself.

BERT false negatives

- Jessica loaded boxes on the wagon.
- Carla slid the book.
- Susan whispered at Rachel.
- It is a golden hair.

→ John promise Mary to shave himself.

→ I might be not going to the party but washing my hair

KnowBert-UMLS false negatives

- The mechanical doll wriggled itself loose.

→ Clearly, John probably will immediately learn French perfectly.

→ Rusty talked about himself only after Mary did talk about him.

- I won't ask you to believe that he tried to force me to give her any money.
- The gardener grew that acorn into an oak tree.
- After reading the pamphlet, Judy threw it into the garbage can.
- The boy in the doorway waved to his father.
- That dog is so ferocious, it even tried to bite itself.
- Ann may spend her vacation in Italy.

→ She asked was Alison coming to the party.

→ It is some disgruntled old pigs in those ditches that humans love to eat.

B - Model hyperparameters for chapter 5

Our experiments revealed that the optimal pre-training hyperparameters were identical across models. They are therefore not differentiated in Table B.1. The hyperparameters of the fine-tuning tasks were optimized with respect to $\text{BERT}_{\text{BASE}}$. There is no entry for the MLM task because the models are already optimized for this task in pre-training.

Task	Learning Rate	Weight Decay	Batch size	Epochs
Pre-training	2e-6	0.01	4096	1
ChemProt	2e-5	0.01	24	30
PubMedQA	2e-5	0.01	32	5
CoLA	2e-5	0.01	32	10
SNLI	2e-5	0.01	32	5

Table B.1: Hyperparameter values for the various tasks.

C - Résumé long en Français

Les Modèles de langage de pointe sont capables de converser, résumer, traduire, résoudre des problèmes inédits, raisonner, et manipuler des concepts abstraits à un niveau quasi-humain. Cependant, pour acquérir ces capacités, et en particulier pour acquérir une forme de "bon sens" ou des connaissances spécifiques à un domaine, ils requièrent de vastes quantités de texte, qui ne sont pas disponibles pour toutes les langues ou dans tous les domaines. Pour l'adaptation au domaine en particulier, l'approche par pré-entraînement cause une forme d'oubli catastrophique, qui affaiblit le modèle de langage sur le domaine général et rend difficile l'adaptation multi-domaine. De surcroît, leurs besoins en puissance de calcul ne sont atteignables que par quelques organisations à l'échelle mondiale, limitant leur spécificité ainsi que leur applicabilité aux données sensibles. Enfin, les paradigmes d'apprentissage actuels tels que l'apprentissage par renforcement avec rétroaction humaine créent pour les modèles une incitation perverse à duper les évaluateurs humains plutôt que d'admettre leur incompétence.

Les *Graphes de Connaissances* sont des sources de connaissances structurées qui associent des concepts linguistiques entre eux par le biais de relations sémantiques. Ces graphes sont des sources de connaissances de haute qualité, préexistantes dans une variété de domaines même peu dotés en ressources, et plus denses en informations que du texte. En permettant aux modèles de langage d'exploiter ces structures d'information, ils sont délestés de la responsabilité de mémoriser les informations factuelles, réduisant la quantité de ressources textuelles et calculatoires nécessaires à leur entraînement, et nous permettant de mettre à jour leurs connaissances à moindre coût, élargissant leur cadre d'application, augmentant leur potentiel de démocratisation, et réduisant leur incitation au mensonge.

Diverses approches pour l'amélioration de modèkes de langage par intégration de graphes de connaissances ont démontré leur efficacité. Elles reposent cependant sur la supposition rarement vérifiée que le problème de *Désambiguïsation d'Entités Nommées* est résolu en amont. Dans cette thèse, nous examinons la contribution des modèles de langue attentionnels de type *Transformer* à la tâche de désambiguïsation d'entités nommées lorsqu'elle est traitée comme une tâche de reconnaissance fine d'entitées nommées. Nos résultats ainsi que ceux des travaux subséquents de la littérature allant en ce sens nous mènent à conclure que cette approche présente des faiblesses face aux entités rarement mentionnées et nécessite de grandes quantités de texte d'apprentissage annoté. L'intégration de connaissances étant *a priori* la plus utile lorsqu'elle est appliquée aux entités rarement vues dans le corpus d'entraînement, cette approche à la désambiguïsation semble fondamentalement contreproductive pour notre cas d'application.

Est ensuite exploré l'apprentissage simultané de modélisation de langue et de désambiguïsation d'entités nommées. L'approche consiste à séparer la tâche de désambiguïsation en *Détection de Mentions*, *Génération de Candidats* et *Désambiguïsation de Candidats*. Dans ce paradigme, la détection de mentions et la

désambiguïsation de candidats peuvent être effectuées par le modèle de langage, qui dispose d'une bonne maîtrise de la grammaire et du contexte, et la génération de candidats peut être effectuée de manière classique par un algorithme de recherche de chaînes de caractères qui n'est pas biaisé en faveur des entités courantes dans le langage. Cette démarche s'avère viable mais échoue à réduire considérablement la quantité de texte nécessaire à l'adaptation au domaine.

Enfin, cette thèse aborde la stratégie de générer du texte à partir de graphes de connaissances de manière à exploiter les capacités linguistiques des modèles de langage pour intégrer l'information contenue dans ces graphes. Il en ressort que même une implémentation naïve de cette approche peut se solder par de considérables progrès en modélisation de langue dans des domaines de spécialité. De plus, en fournissant ce texte de synthèse en *entrée* d'un modèle de langage, cette approche ouvre la voie vers une intégration de connaissances à l'inférence et non à l'apprentissage, ce qui requerrait significativement moins de puissance de calcul et permettrait de mettre à jour les connaissances sans ré-entraînement du modèle.

Bibliography

S. Ahn, H. Choi, T. Pärnamaa, and Y. Bengio. A neural knowledge language model. *arXiv preprint arXiv:1608.00318*, 2016.

A. Y. Y. i. I. a.-S. al Kindi. Manuscript on deciphering cryptographic messages, 9th century.

A. Alhelbawy and R. Gaizauskas. Collective named entity disambiguation using graph ranking and clique partitioning approaches. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1544–1555, 2014.

Alignment Research Center. Update on ARC's recent eval efforts of gpt-4 and claude, 2023. URL https://evals.alignment.org/blog/2023-03-18-update-on-recent-evals/.

E. Alsentzer, J. Murphy, W. Boag, W.-H. Weng, D. Jindi, T. Naumann, and M. McDermott. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, 2019.

J. Aron. Software tricks people into thinking it is human. *New Scientist*, 2829, 2011.

K. Arumae and P. Bhatia. CALM: Continuous Adaptive Learning for Language Modeling. *arXiv preprint arXiv:2004.03794*, 2020.

D. Bahdanau, K. H. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015a.

D. Bahdanau, K. H. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015b.

I. Beltagy, K. Lo, and A. Cohan. Scibert: A pretrained language model for scientific text. 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)*, pages 3615–3620, 2019.

Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

Y. Bengio, R. Ducharme, and P. Vincent. A neural probabilistic language model. *Advances in neural information processing systems*, 13, 2000.

R. Bhowmik, K. Stratos, and G. de Melo. Fast and effective biomedical entity linking using a dual encoder. In *Proceedings of the 12th International Workshop on Health Text Mining and Information Analysis*, pages 28–37, 2021.

S. Bickel, M. Brückner, and T. Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of the 24th international conference on Machine learning*, pages 81–88, 2007.

J. Blitzer, R. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 120–128, 2006.

M. R. Boguslav, N. D. Hailu, M. Bada, W. A. Baumgartner, and L. E. Hunter. Concept recognition as a machine translation problem. *BMC bioinformatics*, 22: 1–39, 2021.

A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.

S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2015. doi: 10.18653/v1/D15-1075. URL https://aclanthology.org/D15-1075.

S. Broscheit. Investigating entity knowledge in BERT with simple neural end-to-end entity linking. *arXiv preprint arXiv:2003.05473*, 2020.

P. E. Brown, V. J. Della Pietra, S. A. Della Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2), 1993.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.* Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, *et al.* Sparks of artificial general intelligence: Early experiments with GPT-4. *arXiv preprint arXiv:2303.12712*, 2023.

R. Carpenter. Cleverbot, 2008. URL https://www.cleverbot.com/.

A. Celikyilmaz, D. Hakkani-Tur, P. Pasupat, and R. Sarikaya. Enriching word embeddings using knowledge graph for semantic tagging in conversational dialog systems. In *2015 AAAI Spring Symposium Series*, 2015.

I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos. LEGAL-BERT: The Muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.261. URL https://aclanthology.org/2020.findings-emnlp.261.

H. Chen, X. Li, A. Zukov Gregoric, and S. Wadhwa. Contextualized End-to-End Neural Entity Linking. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 637–642, Suzhou, China, Dec. 2020. Association for Computational Linguistics.

M. Chen, Z. Xu, K. Q. Weinberger, and F. Sha. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1627–1634, 2012.

K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014.

N. Chomsky. *Syntactic structures*. Mouton de Gruyter, 1957.

P. Christiano, B. Shlegeris, and D. Amodei. Supervising strong learners by amplifying weak experts. *arXiv preprint arXiv:1810.08575*, 2018.

P. Colon-Hernandez, C. Havasi, J. Alonso, M. Huggins, and C. Breazeal. Combining pre-trained language models and structured knowledge. *arXiv preprint arXiv:2101.12294*, 2021.

J. Cuzzola, J. Jovanović, and E. Bagheri. RysannMD: A biomedical semantic annotator balancing speed and accuracy. *Journal of Biomedical Informatics*, 71: 91–109, July 2017. ISSN 1532-0464. doi: 10.1016/j.jbi.2017.05.016.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

T. Dash, A. Srinivasan, L. Vig, and A. Roy. Using domain-knowledge to assist lead discovery in early-stage drug design. In *International Conference on Inductive Logic Programming*, pages 78–94. Springer, 2021.

T. Dash, S. Chitlangia, A. Ahuja, and A. Srinivasan. A review of some techniques for inclusion of domain-knowledge into deep neural networks. *Scientific Reports*, 12(1):1040, 2022.

H. Daumé III. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, 2007.

H. Daumé III and D. Marcu. Domain adaptation for statistical classifiers. *Journal of artificial Intelligence research*, 26:101–126, 2006.

A. S. d'Avila Garcez and G. Zaverucha. The connectionist inductive learning and logic programming system. *Applied Intelligence*, 11:59–77, 1999.

S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

J. Devlin, R. R. Bunel, R. Singh, M. Hausknecht, and P. Kohli. Neural program meta-induction. *Advances in Neural Information Processing Systems*, 30, 2017.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

L. L. Di Langosco, J. Koch, L. D. Sharkey, J. Pfau, and D. Krueger. Goal misgeneralization in deep reinforcement learning. In *International Conference on Machine Learning*, pages 12004–12019. PMLR, 2022.

A. Dirkson, S. Verberne, and W. Kraaij. FuzzyBIO: A proposal for fuzzy representation of discontinuous entities. In *Proceedings of the 12th International Workshop on Health Text Mining and Information Analysis*, pages 77–82, 2021.

J. Dodge, M. Sap, A. Marasović, W. Agnew, G. Ilharco, D. Groeneveld, M. Mitchell, and M. Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. *arXiv preprint arXiv:2104.08758*, 2021.

A. Dsouza, N. Tempelmeier, R. Yu, S. Gottschalk, and E. Demidova. Worldkg: A world-scale geographic knowledge graph. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 4475–4484, 2021.

G. Elbaz, P. Norvig, N. Spivack, C. Malamud, K. Bollacker, and J. Ito. Common crawl, 2012. URL https://commoncrawl.org/.

K. Ellis, L. Morales, M. S. Meyer, A. Solar-Lezama, and J. B. Tenenbaum. Dreamcoder: Bootstrapping domain-specific languages for neurally-guided bayesian

program learning. In *Proceedings of the 2nd Workshop on Neural Abstract Machines and Program Induction*, 2018.

J. L. Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

M. Enzenberger. The integration of a priori knowledge into a go playing neural network. *URL: http://www. markus-enzenberger. de/neurogo. html*, 1996.

Epoch. Parameter, compute and data trends in machine learning, 2022. URL https://epochai.org/mlinputs/visualization. Accessed: 2023-7-20.

A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *nature*, 542(7639):115–118, 2017.

M. Fischer, M. Balunovic, D. Drachsler-Cohen, T. Gehr, C. Zhang, and M. Vechev. DL2: training and querying neural networks with logic. In *International Conference on Machine Learning*, pages 1931–1941. PMLR, 2019.

K. C. Fraser, I. Nejadgholi, B. De Bruijn, M. Li, A. LaPlante, and K. Z. El Abidine. Extracting umls concepts from medical text using general and domain-specific deep learning models. *EMNLP-IJCNLP 2019*, page 157, 2019.

L.-M. Fu. Knowledge-based connectionism for revising domain theories. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(1):173–182, 1993.

O.-E. Ganea and T. Hofmann. Deep joint entity disambiguation with local neural attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629, 2017.

Y. Gao, Y. Xiong, S. Wang, and H. Wang. Geobert: Pre-training geospatial representation learning on point-of-interest. *Applied Sciences*, 12(24):12942, 2022.

M. Gaur, K. Faldu, and A. Sheth. Semantics of the black-box: Can knowledge graphs help make deep learning systems more interpretable and explainable? *IEEE Internet Computing*, 25(1):51–59, 2021.

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

X. Glorot, A. Bordes, and Y. Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 513–520, 2011.

Y. C. Goh, X. Q. Cai, W. Theseira, G. Ko, and K. A. Khor. Evaluating human versus machine learning performance in classifying research abstracts. *Scientometrics*, 125:1197–1212, 2020.

B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2066–2073. IEEE, 2012.

R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *2011 international conference on computer vision*, pages 999–1006. IEEE, 2011.

J. A. Gordon and M. P. Stryker. Experience-dependent plasticity of binocular responses in the primary visual cortex of the mouse. *Journal of Neuroscience*, 16(10):3274–3286, 1996.

H. Greenspan, B. Van Ginneken, and R. M. Summers. Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE transactions on medical imaging*, 35(5):1153–1159, 2016.

A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel method for the two-sample-problem. *Advances in neural information processing systems*, 19, 2006.

Y. Gu, R. Tinn, H. Cheng, M. Lucas, N. Usuyama, X. Liu, T. Naumann, J. Gao, and H. Poon. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23, 2021.

S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360. Association for Computational Linguistics, July 2020. doi: 10.18653/v1/2020.acl-main.740. URL https://aclanthology.org/2020.acl-main.740.

X. Han, L. Sun, and J. Zhao. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774, 2011.

D. A. Handelman, S. H. Lane, and J. J. Gelfand. Integrating neural networks and knowledge-based systems for intelligent robotic control. *IEEE Control Systems Magazine*, 10(3):77–87, 1990.

B. He, D. Zhou, J. Xiao, X. Jiang, Q. Liu, N. J. Yuan, and T. Xu. Integrating graph contextualized knowledge into pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2281–2290, 2020.

J. J. Heckman. Sample selection bias as a specification error. *Econometrica: Journal of the econometric society*, pages 153–161, 1979.

M. Heikkilä. We're getting a better idea of AI's true carbon footprint. *Technology Review*, 2022. URL https://www.technologyreview.com/2022/11/14/1063192.

G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 782–792, 2011a.

J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 782–792, 2011b.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Y. Hou, G. Fu, and M. Sachan. Understanding knowledge integration in language models with graph convolutions. *arXiv preprint arXiv:2202.00964*, 2022.

N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

J. Howard and S. Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, 2018.

Z. Hu, X. Ma, Z. Liu, E. Hovy, and E. Xing. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2410–2420, 2016.

S. Huang, L. Dong, W. Wang, Y. Hao, S. Singhal, S. Ma, T. Lv, L. Cui, O. K. Mohammed, Q. Liu, *et al.* Language is not all you need: Aligning perception with language models. *arXiv preprint arXiv:2302.14045*, 2023.

E. Hubinger, C. van Merwijk, V. Mikulik, J. Skalse, and S. Garrabrant. Risks from learned optimization in advanced machine learning systems. *arXiv preprint arXiv:1906.01820*, 2019.

S. Humeau, K. Shuster, M.-A. Lachaux, and J. Weston. Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring. *arXiv preprint arXiv:1905.01969*, 2019.

H. Hyötyniemi. Turing machines are recurrent neural networks. *Proceedings of step*, 96, 1996.

F. Ilievski, P. Szekely, and B. Zhang. Cskg: The commonsense knowledge graph. In *The Semantic Web: 18th International Conference, ESWC 2021, Virtual Event, June 6–10, 2021, Proceedings 18*, pages 680–696. Springer, 2021.

D. Isele and A. Cosgun. Selective experience replay for lifelong learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

J. Jiang. A literature survey on domain adaptation of statistical classifiers. *URL: http://sifaka. cs. uiuc. edu/jiang4/domainadaptation/survey*, 3(1-12):3, 2008.

J. Jiang and C. Zhai. Instance weighting for domain adaptation in nlp. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271, 2007.

Q. Jin, B. Dhingra, Z. Liu, W. Cohen, and X. Lu. Pubmedqa: A dataset for biomedical research question answering. 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)*, pages 2567–2577, 2019.

K. S. Jones. Index term weighting. *Information storage and retrieval*, 9(11):619–633, 1973.

J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

S. W. Kim, I. Kim, J. Lee, and S. Lee. Knowledge integration into deep learning in dynamical systems: an overview and taxonomy. *Journal of Mechanical Science and Technology*, 35:1331–1342, 2021.

J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.* Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

N. Kolitsas, O.-E. Ganea, and T. Hofmann. End-to-end neural entity linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 519–529, 2018.

Z. Kraljevic, T. Searle, A. Shek, L. Roguski, K. Noor, D. Bean, A. Mascio, L. Zhu, A. A. Folarin, A. Roberts, R. Bendayan, M. P. Richardson, R. Stewart, A. D. Shah, W. K. Wong, Z. Ibrahim, J. T. Teo, and R. J. B. Dobson. Multi-domain clinical natural language processing with medcat: the medical concept annotation toolkit. *Artificial Intelligence in Medicine*, 117:102083, 2021.

M. Krallinger, O. Rabal, S. A. Akhondi, M. P. Pérez, J. Santamaría, G. P. Rodríguez, G. Tsatsaronis, A. Intxaurrondo, J. A. López, U. Nandal, *et al.* Overview of the biocreative vi chemical-protein interaction track. In *Proceedings of the sixth BioCreative challenge evaluation workshop*, volume 1, pages 141–146, 2017.

D. Kremelberg. Embodiment as a necessary a priori of general intelligence. In *Artificial General Intelligence: 12th International Conference, AGI 2019, Shenzhen, China, August 6–9, 2019, Proceedings 12*, pages 132–136. Springer, 2019.

A. Kroshchanka, V. Golovko, E. Mikhno, M. Kovalev, V. Zahariev, and A. Zagorskij. A neural-symbolic approach to computer vision. In *International Conference on Open Semantic Technologies for Intelligent Systems*, pages 282–309. Springer, 2021.

U. Kursuncu, M. Gaur, and A. Sheth. Knowledge infused learning (k-il): Towards deep incorporation of knowledge in deep learning. *arXiv preprint arXiv:1912.00512*, 2019.

B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. ALBERT: A Lite BERT for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

N. Lavrač, S. Džeroski, and M. Grobelnik. *Learning nonrecursive definitions of relations with LINUS*. Springer, 1991.

N. Lavrač, B. Škrlj, and M. Robnik-Šikonja. Propositionalization and embeddings: two sides of the same coin. *Machine Learning*, 109:1465–1507, 2020.

P. Le and I. Titov. Improving entity linking by modeling latent relations between mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1595–1604, 2018.

Y. Le Cun. Yann Le Cun : ChatGPT, c'est "de la bonne ingénierie" mais "pas révolutionnaire", 2023. URL https://www.youtube.com/watch?v=NBRXUTs5IcM. Interview.

R. Leaman and Z. Lu. TaggerOne: joint named entity recognition and normalization with semi-markov models. *Bioinformatics*, 32(18):2839–2846, 2016.

J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, page btz682, Sept. 2019. ISSN 1367-4803, 1460-2059. doi: 10.1093/bioinformatics/btz682.

J.-S. Lee and J. Hsiang. Patent classification by fine-tuning BERT language model. *World Patent Information*, 61:101965, 2020.

K. Lee, L. He, M. Lewis, and L. Zettlemoyer. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, 2017.

P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, *et al.* Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

Y. Li, M. Mamouei, G. Salimi-Khorshidi, S. Rao, A. Hassaine, D. Canoy, T. Lukasiewicz, and K. Rahimi. Hi-behrt: Hierarchical transformer-based model for accurate prediction of clinical events using multimodal longitudinal electronic health records. *IEEE journal of biomedical and health informatics*, 27(2):1106–1117, 2022.

B. Y. Lin, X. Chen, J. Chen, and X. Ren. Kagnet: Knowledge-aware graph networks for commonsense reasoning. *arXiv preprint arXiv:1909.02151*, 2019.

C. Lin, S. Bethard, D. Dligach, F. Sadeque, G. Savova, and T. A. Miller. Does BERT need domain adaptation for clinical negation detection? *Journal of the American Medical Informatics Association*, 27(4):584–591, 2020.

Y. Lin, Y. Lee, and G. Wahba. Support vector machines for classification in nonstandard situations. *Machine learning*, 46:191–202, 2002.

D. A. Lindberg, B. L. Humphreys, and A. T. McCray. The unified medical language system. *Yearbook of medical informatics*, 2(01):41–51, 1993.

H. Lio, S.-E. Li, and J.-T. Chien. Adversarial mask transformer for sequential learning. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4178–4182. IEEE, 2022.

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

L. Logeswaran, M.-W. Chang, K. Lee, K. Toutanova, J. Devlin, and H. Lee. Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3449–3460, 2019.

T. Long, R. Lowe, J. C. K. Cheung, and D. Precup. Leveraging lexical resources for learning entity embeddings in multi-relational data. *arXiv preprint arXiv:1605.05416*, 2016.

X. Ma, P. Xu, Z. Wang, R. Nallapati, and B. Xiang. Domain adaptation with BERT-based domain classification and data selection. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 76–83, 2019.

Y. Ma, H. Peng, and E. Cambria. Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive LSTM. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, pages 5876–5883, 2018.

R. Maldonado, M. Yetisgen, and S. M. Harabagiu. Adversarial learning of knowledge embeddings for the unified medical language system. *AMIA Summits on Translational Science Proceedings*, 2019:543, 2019.

C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge University Press Cambridge, 2008.

K. Marino, X. Chen, D. Parikh, A. Gupta, and M. Rohrbach. Krisp: Integrating implicit and symbolic knowledge for open-domain knowledge-based vqa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14111–14121, 2021.

J. R. Meehan. TALE-SPIN, an interactive program that writes stories. In *Ijcai*, volume 77, pages 91–98, 1977.

G. Michalopoulos, Y. Wang, H. Kaka, H. Chen, and A. Wong. UmlsBERT: Clinical domain knowledge augmentation of contextual embeddings using the Unified Medical Language System Metathesaurus. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1744–1753, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.139. URL https://aclanthology.org/2021.naacl-main.139.

T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari, 2010.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

A. Mnih and G. E. Hinton. A scalable hierarchical distributed language model. *Advances in neural information processing systems*, 21, 2008.

S. Mohan and D. Li. MedMentions: A Large Biomedical Corpus Annotated with UMLS Concepts. *arXiv:1902.09476 [cs]*, Feb. 2019.

J. G. Moreno, R. Besançon, R. Beaumont, E. D'hondt, A.-L. Ligozat, S. Rosset, X. Tannier, and B. Grau. Combining word and entity embeddings for entity linking. In *The Semantic Web: 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28–June 1, 2017, Proceedings, Part I 14*, pages 337–352. Springer, 2017.

F. Morin and Y. Bengio. Hierarchical probabilistic neural network language model. In *International workshop on artificial intelligence and statistics*, pages 246–252. PMLR, 2005.

S. Muggleton. Inductive logic programming. *New generation computing*, 8:295–318, 1991.

N. Muralidhar, M. R. Islam, M. Marwah, A. Karpatne, and N. Ramakrishnan. Incorporating prior domain knowledge into deep neural networks. In *2018 IEEE international conference on big data (big data)*, pages 36–45. IEEE, 2018.

H. Nakayama. seqeval: A python framework for sequence labeling evaluation, 2018. URL https://github.com/chakki-works/seqeval.

C. Nardo. The Waluigi effect (mega-post), 2023. URL https://www.alignmentforum.org/posts/D7PumeYTDPfBTp3i7/.

R. M. Neal. *Bayesian learning for neural networks*. PhD thesis, University of Calgary, 1995.

M. Neumann, D. King, I. Beltagy, and W. Ammar. ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, Florence, Italy, Aug. 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-5034. URL https://www.aclweb.org/anthology/W19-5034.

M. Noordewier, G. Towell, and J. Shavlik. Training knowledge-based neural networks to recognize genes in dna sequences. *Advances in neural information processing systems*, 3, 1990.

W. L. Oliver, W. Schneider, *et al. Using rules and task division to augment connectionist learning*. Depts. of Computer Science and Psychology, Carnegie Mellon University, 1988.

R. OpenAI. GPT-4 technical report. *arXiv*, pages 2303–08774, 2023.

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, *et al.* Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE transactions on neural networks*, 22(2):199–210, 2010.

R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, volume 28, pages 1310–1318. Pmlr, 2013.

Y. Peng, S. Yan, and Z. Lu. Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65, 2019.

J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL http://www.aclweb.org/anthology/D14-1162.

E. Perez, S. Ringer, K. Lukošiūtė, K. Nguyen, E. Chen, S. Heiner, C. Pettit, C. Olsson, S. Kundu, S. Kadavath, *et al.* Discovering language model behaviors with model-written evaluations. *arXiv preprint arXiv:2212.09251*, 2022a.

E. Perez, S. Ringer, K. Lukošiūtė, K. Nguyen, E. Chen, S. Heiner, C. Pettit, C. Olsson, S. Kundu, S. Kadavath, *et al.* Discovering language model behaviors with model-written evaluations. *arXiv preprint arXiv:2212.09251*, 2022b.

M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018.

M. E. Peters, M. Neumann, R. L. Logan, R. Schwartz, V. Joshi, S. Singh, and N. A. Smith. Knowledge enhanced contextual word representations. In *EMNLP*, 2019.

J. Pfeiffer, A. Rücklé, C. Poth, A. Kamath, I. Vulić, S. Ruder, K. Cho, and I. Gurevych. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, 2020.

F. Piat and G. Stamou. Refining rules of emotion recognition in hybrid systems. In *Proc. of Intl. Conf. on Circuits, Systems, Communications and Computers (CSCC)*, pages 533–540. Citeseer, 1999.

G. Piat, N. Semmar, A. Allauzen, H. Essafi, G. Bernard, and J. Tourille. Adapting without forgetting: KnowBert-UMLS. In *2022 18th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 19–24. IEEE, 2022a.

G. Piat, N. Semmar, A. Allauzen, H. Essafi, and J. Tourille. Enriching contextualized representations with biomedical ontologies: Extending knowbert to umls. In *Science and Information Conference*, pages 760–773. Springer, 2022b.

J. R. Pierce, J. B. Carroll, E. P. Hamp, D. G. Hays, C. F. Hockett, A. G. Oettinger, and A. Perlis. Returns to investment in higher education. Technical report, Language and Machines: Computers in Translation and Linguistics, 1966.

N. Poerner, U. Waltinger, and H. Schütze. E-BERT: Efficient-Yet-Effective Entity Embeddings for BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 803–818, 2020.

S. Polat. Seed. *Webtoon*, Episode 137, 2023. URL https://www.webtoons.com/en/sf/seed/episode-137/viewer?title_no=1480&episode_no=137.

A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.* Improving language understanding by generative pre-training. *OpenAI blog*, page 12, 2018.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.* Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya, *et al.* Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. *arXiv preprint arXiv:1711.05225*, 2017.

D. Rao, P. McNamee, and M. Dredze. Entity linking: Finding extracted entities in a knowledge base. *Theory and Applications of Natural Language Processing*, 2013.

M. P. K. Ravi, K. Singh, I. O. Mulang, S. Shekarpour, J. Hoffart, and J. Lehmann. CHOLAN: A modular approach for neural entity linking on wikipedia and wikidata. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 504–514, 2021.

A. Robinson and F. Fallside. *The utility driven dynamic error propagation network*, volume 1. University of Cambridge Department of Engineering Cambridge, 1987.

T. Rocktäschel, M. Bosnjak, S. Singh, and S. Riedel. Low-dimensional embeddings of logic. In *Proceedings of the ACL 2014 workshop on semantic parsing*, pages 45–49, 2014.

A. Rogers, O. Kovaleva, and A. Rumshisky. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2021.

S. Ruder. *Neural transfer learning for natural language processing*. PhD thesis, NUI Galway, 2019.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, 1:318–362, 1986.

V. Sanh, T. Wolf, and S. Ruder. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6949–6956, 2019.

M. Sap, R. Le Bras, E. Allaway, C. Bhagavatula, N. Lourie, H. Rashkin, B. Roof, N. A. Smith, and Y. Choi. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3027–3035, 2019.

G. K. Savova, J. J. Masanz, P. V. Ogren, J. Zheng, S. Sohn, K. C. Kipper-Schuler, and C. G. Chute. Mayo clinical text analysis and knowledge extraction system (ctakes): architecture, component evaluation and applications. *Journal of the American Medical Informatics Association*, 17(5):507–513, 2010.

R. C. Schank, N. M. Goldman, C. J. Rieger III, and C. Riesbeck. Margie: Memory analysis response generation, and inference on english. In *IJCAI*, pages 255–261, 1973.

J. Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139, 1992.

J. Schulman, B. Zoph, C. Kim, J. Hilton, J. Menick, J. Weng, J. F. C. Uribe, L. Fedus, L. Metz, M. Pokorny, *et al.* ChatGPT: Optimizing language models for dialogue. *OpenAI blog*, 2022.

R. Sennrich, B. Haddow, and A. Birch. Neural machine translation of rare words with subword units. In *54th Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725. Association for Computational Linguistics (ACL), 2016.

J. Sevilla, L. Heim, A. Ho, T. Besiroglu, M. Hobbhahn, and P. Villalobos. Compute trends across three eras of machine learning. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2022.

A. Sheth, M. Gaur, U. Kursuncu, and R. Wickramarachchi. Shades of knowledge-infused learning for enhancing deep learning. *IEEE Internet Computing*, 23(6): 54–63, 2019.

H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2): 227–244, 2000.

H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5):1285–1298, 2016.

L. Soldaini and N. Goharian. Quickumls: a fast, unsupervised approach for medical concept extraction. In *MedIR workshop, sigir*, pages 1–4, 2016.

F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A Core of Semantic Knowledge. In *16th International Conference on the World Wide Web*, pages 697–706, 2007.

Y. Sun, S. Wang, Y. Li, S. Feng, X. Chen, H. Zhang, X. Tian, D. Zhu, H. Tian, and H. Wu. ERNIE: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*, 2019.

N. Takeishi and K. Akimoto. Knowledge-based distant regularization in learning probabilistic models. *arXiv preprint arXiv:1806.11332*, 2018.

A.-H. Tan. Cascade artmap: Integrating neural computation and symbolic knowledge processing. *IEEE Transactions on Neural Networks*, 8(2):237–250, 1997.

G. G. Towell and J. W. Shavlik. Knowledge-based artificial neural networks. *Artificial intelligence*, 70(1-2):119–165, 1994.

G. G. Towell, J. W. Shavlik, and M. O. Noordewier. Refinement of approximate domain theories by knowledge-based neural networks. In *Proceedings of the eighth National conference on Artificial intelligence-Volume 2*, pages 861–866, 1990.

A. M. Turing. *Computing machinery and intelligence*. Springer, 1950.

Ö. Uzuner, B. R. South, S. Shen, and S. L. DuVall. 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556, 2011.

J. M. Van Hulst, F. Hasibi, K. Dercksen, K. Balog, and A. P. de Vries. REL: An Entity Linker Standing on the Shoulders of Giants. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2197–2200, 2020.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

O. Vinyals, Ł. Kaiser, T. Koo, S. Petrov, I. Sutskever, and G. Hinton. Grammar as a foreign language. *Advances in neural information processing systems*, 28, 2015.

L. Von Rueden, S. Mayer, K. Beckh, B. Georgiev, S. Giesselbach, R. Heese, B. Kirsch, J. Pfrommer, A. Pick, R. Ramamurthy, *et al.* Informed machine learning–a taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):614–633, 2021.

T. Vu, D. Phung, and G. Haffari. Effective unsupervised domain adaptation with adversarially trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6163–6173, 2020.

R. Wang, D. Tang, N. Duan, Z. Wei, X. Huang, J. ji, G. Cao, D. Jiang, and M. Zhou. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. *arXiv:2002.01808 [cs]*, Dec. 2020.

W. Wang, H. Li, Z. Ding, F. Nie, J. Chen, X. Dong, and Z. Wang. Rethinking maximum mean discrepancy for visual domain adaptation. *IEEE Transactions on Neural Networks and Learning Systems*, 2021a.

X. Wang, T. Gao, Z. Zhu, Z. Zhang, Z. Liu, J. Li, and J. Tang. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194, 2021b. doi: 10.1162/tacl_a_00360. URL https://aclanthology.org/2021.tacl-1.11.

Y. Wang, Z. Liu, and M. Sun. Incorporating linguistic knowledge for learning distributed word representations. *PloS one*, 10(4), 2015. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0118437.

A. Warstadt, A. Singh, and S. R. Bowman. Neural Network Acceptability Judgments. *Transactions of the Association for Computational Linguistics*, 7: 625–641, 09 2019. ISSN 2307-387X. doi: 10.1162/tacl_a_00290. URL https://doi.org/10.1162/tacl_a_00290.

W. Weaver. Translation. In W. N. Locke and A. D. Boothe, editors, *Machine Translation of Languages*, pages 15–23. MIT Press, Cambridge, MA, 1949/1955.

J. Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1): 36–45, 1966.

M. Wiatrak and J. Iso-Sipila. Simple hierarchical multi-task neural end-to-end entity linking for biomedical text. In *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis*, pages 12–17, 2020.

M. Wiatrak, E. Arvaniti, A. Brayne, J. Vetterle, and A. Sim. Proxy-based zero-shot entity linking by effective candidate retrieval. In *Proceedings of the 13th International Workshop on Health Text Mining and Information Analysis (LOUHI)*, pages 87–99, 2022.

J. Wieting, M. Bansal, K. Gimpel, and K. Livescu. Towards universal paraphrastic sentence embeddings. In *4th International Conference on Learning Representations, ICLR*, 2016. URL http://arxiv.org/abs/1511.08198.

L. Wu, F. Petroni, M. Josifoski, S. Riedel, and L. Zettlemoyer. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6397–6407, 2020.

L. Wu, Y. Chen, K. Shen, X. Guo, H. Gao, S. Li, J. Pei, B. Long, *et al.* Graph neural networks for natural language processing: A survey. *Foundations and Trends® in Machine Learning*, 16(2):119–328, 2023.

Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al.* Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

J. Xu, Z. Zhang, T. Friedman, Y. Liang, and G. Broeck. A semantic loss function for deep learning with symbolic knowledge. In *International conference on machine learning*, pages 5502–5511. PMLR, 2018.

Y. Xu, X. Zhong, A. J. J. Yepes, and J. H. Lau. Forget me not: Reducing catastrophic forgetting for domain adaptation in reading comprehension. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

B. Yang and T. Mitchell. Leveraging knowledge bases in LSTMs for improving machine reading. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1436–1446, 2017.

B. Yang, S. W.-t. Yih, X. He, J. Gao, and L. Deng. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*, 2015.

E. Yang, S. MacAvaney, D. D. Lewis, and O. Frieder. Goldilocks: Just-right tuning of BERT for technology-assisted review. In *European Conference on Information Retrieval*, pages 502–517. Springer, 2022.

J. Yang, H. Jin, R. Tang, X. Han, Q. Feng, H. Jiang, B. Yin, and X. Hu. Harnessing the power of LLMs in practice: A survey on ChatGPT and beyond. *arXiv preprint arXiv:2304.13712*, 2023.

Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le. Xlnet: generalized autoregressive pretraining for language understanding. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 5753–5763, 2019.

M. Yasunaga, A. Bosselut, H. Ren, X. Zhang, C. D. Manning, P. S. Liang, and J. Leskovec. Deep bidirectional language-knowledge graph pretraining. *Advances in Neural Information Processing Systems*, 35:37309–37323, 2022.

Z. Ye, Y. J. Kumar, G. O. Sing, F. Song, and J. Wang. A comprehensive survey of graph neural networks for knowledge graphs. *IEEE Access*, 10:75729–75741, 2022.

J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

Z. Yuan, Z. Zhao, H. Sun, J. Li, F. Wang, and S. Yu. CODER: Knowledge-infused cross-lingual medical term embedding for term normalization. *Journal of biomedical informatics*, 126:103983, 2022.

B. Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning*, page 114, 2004.

R. Zhang, Y. Zheng, X. Mao, and M. Huang. Unsupervised domain adaptation with adapter. *Advances in neural information processing systems*, 34, 2021.

X. Zhang, A. Bosselut, M. Yasunaga, H. Ren, P. Liang, C. Manning, and J. Leskovec. GreaseLM: Graph REASoning Enhanced Language Models for Question Answering. In *International Conference on Representation Learning (ICLR)*, 2022.

Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1139. URL https://aclanthology.org/P19-1139.

Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler. 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*, pages 19–27, 2015.

S. Zhuang and D. Hadfield-Menell. Consequences of misaligned AI. *Advances in Neural Information Processing Systems*, 33:15763–15773, 2020.

G. K. Zipf. *The Psycho-Biology of language*, volume 21. Psychology Press, 1935.

Y. Ziser and R. Reichart. Neural structural correspondence learning for domain adaptation. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 400–410. Association for Computational Linguistics, Aug. 2017. doi: 10.18653/v1/K17-1040. URL https://aclanthology.org/K17-1040.