Evaluating and Improving the Reasoning Abilities of Language Models

Chadi Helwe

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2024IPPAT021

Thèse de doctorat

Evaluating and Improving the Reasoning Abilities of Language Models

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Télécom Paris

École doctorale n°626 École doctorale de l'Institut Polytechnique de Paris (ED IP Paris)
Spécialité de doctorat: Informatique, Données, IA

Thèse présentée et soutenue à Palaiseau, le 5 Juillet 2024, par

CHADI HELWÉ

Composition du Jury :

Benoît Sagot Directeur de Recherche, Inria Paris	Président / Examinateur
Serena Villata Directrice de Recherche, Université Côte d'Azur / CNRS	Rapportrice
Paolo Rosso Full Professor, Universitat Politècnica de València	Rapporteur
Farah Benamara Full Professor, Université Paul Sabatier de Toulouse	Examinatrice
Fabian Suchanek Full Professor, Télécom Paris / Institut Polytechnique de Paris	Directeur de thèse
Chloé Clavel Directrice de Recherche, Inria Paris	Co-directrice de thèse

PHD THESIS

Evaluating and Improving the Reasoning Abilities of Language Models

Author:
Chadi HELWÉ

TÉLÉCOM PARIS
Institut Polytechnique de Paris

Acknowledgements

I am grateful to the many people who have supported and accompanied me on this journey. I spent almost four years in the DIG team at Télécom Paris, and it has been an amazing experience.

I want to express my gratitude to my two advisors. First, I would like to extend my thanks to Fabian, my advisor at Télécom. He is very knowledgeable, and I always enjoyed our discussions, whether it was about research or simply having lunch and chatting about politics and religion. I also want to thank my other advisor, Chloé Clavel, who previously worked at Télécom Paris before joining INRIA Paris. Chloé is a wonderful person, and my meetings with her were always productive. I learned a lot from both of my advisors when it comes to developing new research ideas, writing papers, and presenting my work.

I also want to thank my parents, who always believed in me and supported me. Firstly, I want to thank my dad for providing the best education for my brothers and me in Lebanon. Secondly, I want to thank my mom, who has always pushed me to do my best.

Finally, I want to thank all the colleagues and friends I have met. I will miss our lunches together and the seminars. I especially want to thank all my Lebanese friends at Télécom with whom I played football and tennis. I want to extend special thanks to two friends. Firstly, Pierre-Henri, who was Fabian's postdoc and was like a mentor to me. He always provided help when I needed it and was there to listen when I had problems. The last project in my thesis, MAFALDA, would not have been published at NAACL without Pierre-Henri's help. Secondly, Fadl, who is not just a friend but like a brother to me.

Contents

List of Figures

List of Tables

Abstract

Language Models (LMs) have become a cornerstone in natural language processing, achieving high performance across various applications, including Named Entity Recognition (NER) and Text Summarization. Despite these successes, there is a notable variance in the performance of LMs regarding reasoning tasks.
In this thesis, we investigate the reasoning capabilities of language models. We study both Smaller Language Models (SLMs) (those that can run on a single GPU, like BERT and T5) and Large Language Models (LLMs) (those that have more than 100 billion parameters, like GPT 3.5). Our investigation is centered around three key research questions:

1. **What types of reasoning can SLMs perform effectively?** Chapter 3 is dedicated to exploring and identifying the range and complexity of reasoning that SLMs can handle by taking as case study BERT-like models. Our objective is to uncover the inherent reasoning capabilities of these models and demonstrate that they have inconsistencies in their reasoning abilities. We have observed that although SLMs can adeptly handle some complex tasks, they can struggle with some simpler ones. To demonstrate this, we have developed two simple tasks: the Light Switch task and the Cake task. The Light Switch task is a natural language task that illustrates the Even Parity language, which is the language of bit strings where the number of 1s is even. The Cake task is a natural language task that illustrates the Dyck-2 language, which is the language of strings representing balanced sequences of round brackets "()" and square brackets "[]". We have shown that RoBERTa achieves a low performance on these tasks with an F-measure of 50%. In Chapter 4, we introduce LogiTorch, a Python library that incorporates various benchmarks like LogiQA and ConTRoL, and models like BERTNOT and PRover, which are discussed in Chapter 3, which makes it easier to evaluate the reasoning capabilities of SLMs. In addition, we have shown that our implemented models match the performances of the models in the original papers.

2. **How can we enhance the reasoning capabilities of SLMs?** In Chapter 5, we

explore the issue of negation, which is a crucial component of reasoning. Fine-tuned SLMs like BERT on Natural Language Inference tasks face difficulty when introducing a negation. To address this problem, we propose a new method called TINA (Textual Inference with Negated Augmentation), a principled negated data augmentation technique that automatically generates negative instances. TINA can be combined with the unlikelihood loss to enhance the robustness of SLMs to negation in Textual Entailment tasks, without sacrificing performance on datasets without negation. We have tested TINA with various models including BERT, RoBERTa, and XLNet. Our approach leads to an improvement of up to 21 percentage points, depending on the model and dataset.

3. **How well can LLMs deal with logical fallacies?** Chapter 6 addresses this question by introducing MAFALDA (Multi-level Annotated FAllacy DAtaset), a new benchmark annotated for identifying and categorizing fallacies. Fallacies are erroneous or invalid ways of reasoning. This chapter presents a new taxonomy of fallacies that aligns, consolidates, and unifies existing public fallacy datasets and consists of three levels of granularity. Additionally, informal and formal definitions are provided for each fallacy. The formal definition is presented in the form of a template with variables aimed at supporting annotators in the annotation task. A comprehensive annotation scheme has been developed, which embraces subjectivity, allowing for several alternative annotations for the same fallacious argument. Furthermore, we propose an evaluation metric to handle subjectivity. The chapter also covers the evaluation of different LMs under a zero-shot learning setting on MAFALDA. We experimented with GPT-3.5, an LLM with more than 100 billion parameters, and SLMs with a few billion parameters, like LLaMA-2 and Mistral. We also assessed human performance using a sample of MAFALDA. Our results show that GPT-3.5 performs better than SLMs, while humans perform better than the LMs tested. However, our findings indicate that this task is challenging for both LMs and humans.

Abrégé

Les Modèles de Langage (LMs) sont devenus une pierre angulaire dans le traitement du langage naturel, atteignant des performances élevées dans diverses applications, y compris la Reconnaissance d'Entités Nommées (NER) et la Résumé de Texte. Malgré ces succès, il existe une variance notable dans la performance des LMs en ce qui concerne les tâches de raisonnement.
Dans cette thèse, nous étudions les capacités de raisonnement des modèles de langage. Nous étudions à la fois les Petits Modèles de Langage (SLMs) (ceux qui peuvent fonctionner sur un seul GPU, comme BERT et T5) et les Grands Modèles de Langage (LLMs) (ceux qui ont plus de 100 milliards de paramètres, comme GPT 3.5). Notre enquête est centrée autour de trois questions de recherche clés :

1. **Quels types de raisonnement les SLMs peuvent-ils effectuer efficacement ?** Le chapitre 3 est dédié à explorer et à identifier la portée et la complexité du raisonnement que les SLMs peuvent gérer en prenant comme étude de cas les modèles de type BERT. Notre objectif est de découvrir les capacités de raisonnement inhérentes à ces modèles et de démontrer qu'ils présentent des incohérences dans leurs capacités de raisonnement. Nous avons observé que bien que les SLMs puissent gérer habilement certaines tâches complexes, ils peuvent avoir du mal avec certaines tâches plus simples. Pour démontrer cela, nous avons développé deux tâches simples : la tâche du Light Switch et la tâche du Cake. La tâche du Light Switch est une tâche en langage naturel qui illustre le langage de parité paire, qui est le langage des chaînes de bits où le nombre de 1 est pair. La tâche du Cake est une tâche en langage naturel qui illustre le langage Dyck-2, qui est le langage des chaînes représentant des séquences équilibrées de parenthèses rondes "()" et de crochets "[]". Nous avons montré que RoBERTa atteint une faible performance sur ces tâches avec une F-measure d'environ 50%. Dans le chapitre 4, nous introduisons LogiTorch, une bibliothèque Python qui intègre divers benchmarks comme LogiQA et ConTRoL, et des modèles comme BERTNOT et PRover, qui sont discutés dans le chapitre 3, ce qui facilite l'évaluation des capacités de raisonnement des SLMs. De plus, nous avons

montré que nos modèles implémentés correspondent aux performances des modèles dans les articles originaux.

2. **Comment pouvons-nous améliorer les capacités de raisonnement des SLMs ?** Dans le chapitre 5, nous explorons le problème de la négation, qui est un composant crucial du raisonnement. Les SLMs comme BERT, affinés sur des tâches d'Inférence en Langage Naturel, rencontrent des difficultés lors de l'introduction d'une négation. Pour résoudre ce problème, nous proposons une nouvelle méthode appelée TINA (Textual Inference with Negated Augmentation), une technique d'augmentation de données négatives principée qui génère automatiquement des instances négatives. TINA peut être combinée avec la unlikelihood loss pour améliorer la robustesse des SLMs à la négation dans les tâches d'Inférence Textuelle, sans sacrifier les performances sur les ensembles de données sans négation. Nous avons testé TINA avec divers modèles, y compris BERT, RoBERTa et XLNet. Notre approche conduit à une amélioration allant jusqu'à 21 points de pourcentage, selon le modèle et l'ensemble de données.

3. **Dans quelle mesure les LLM peuvent-ils traiter les sophismes ?** Le chapitre 6 aborde cette question en introduisant MAFALDA (Multi-level Annotated FAllacy DAtaset), un nouveau benchmark annoté pour identifier et catégoriser les sophismes. Les sophismes sont des raisonnements erronés ou invalides. Ce chapitre présente une nouvelle taxonomie des sophismes qui aligne, consolide et unifie les ensembles de données publiques existants sur les sophismes et se compose de trois niveaux de granularité. De plus, des définitions informelles et formelles sont fournies pour chaque sophisme. La définition formelle est présentée sous la forme d'un modèle avec des variables visant à soutenir les annotateurs dans la tâche d'annotation. Un schéma d'annotation complet a été développé, embrassant la subjectivité, permettant plusieurs annotations alternatives pour le même argument fallacieux. En outre, nous proposons une métrique d'évaluation pour gérer la subjectivité. Le chapitre couvre également l'évaluation de différents LMs dans un cadre d'apprentissage zéro-shot sur MAFALDA. Nous avons expérimenté avec GPT-3.5, un LLM avec plus de 100 milliards de paramètres, et des SLMs avec quelques milliards de paramètres, comme LLaMA-2 et Mistral. Nous avons également évalué la performance humaine en utilisant un échantillon de MAFALDA. Nos résultats montrent que GPT-3.5 performe mieux que les SLMs, tandis que les humains performent mieux que les LMs testés. Cependant, nos résultats indiquent que cette tâche est difficile à la fois pour les LMs et pour les humains.

1 Introduction

1.1 Reasoning with Language Models

Recent advances in artificial intelligence have led to the emergence of Large Language Models (LLMs), which are deep neural networks with billions of parameters pre-trained on vast amounts of data to predict the probability of word sequences. LLMs trace their history to simpler neural architectures, such as Word2Vec [124], which has only two layers. Subsequently, more complex neural architectures based on the Transformer architecture [179], like BERT [44] with millions of parameters, were introduced. These models are considered Smaller Language Models (SLMs) [155, 166] because they can run on a single GPU. The Transformer is a neural architecture based on a self-attention mechanism, allowing a model to weigh the importance of different parts of the input data, which is crucial for understanding the context and relationships within the text. Currently, much larger language models also based on the Transformer, like GPT-4 [2], consisting of more than 100 billion parameters, have been developed. These are called Large Language Models (LLMs). Language Models (LMs), including both SLMs and LLMs, have achieved high-performance on many Natural Language Processing (NLP) tasks such as Sentiment Analysis, Document Summarization, and Named Entity Recognition. However, much less attention had been devoted to the task of reasoning.

Reasoning is a cognitive process that involves using existing knowledge, beliefs, and experiences to draw conclusions based on a given set of premises [25]. As described by Daniel Kahneman's System 2 [40], humans use reasoning for deliberate, thoughtful and logical decision-making to solve complex problems. This process can employ various methods. Deductive reasoning draws direct conclusions from given premises; inductive reasoning, based on observations or evidence, leads to probable conclusions; and abductive reasoning proposes plausible explanations for specific observations. Beyond these methods, reasoning manifests in various types, including commonsense reasoning, which deals with everyday knowledge; mathematical reasoning, which involves numerical deduction; and time-based reasoning, focusing on temporal data and sequences.

Recently, there has been a growing interest in exploring the reasoning capabilities

of deep neural networks for reasoning tasks. For example, an SLM, like the pre-trained BERT, can reply to questions such as the following:

Example 1.1

Context: The iPhone is produced by [MASK].
Expected answer: Apple
Model answer: Apple

However, this performance is deceiving: If we introduce a trap word, the pre-trained BERT model replies completely differently:

Example 1.2

Context: Samsung. The iPhone is produced by [MASK].
Expected answer: Apple
Model answer: Samsung

Here, the BERT model got distracted by the additional word (a technique called *mis-priming* [95]). Thus, the question arises to what degree such models really "understand" the natural language text, and to what degree they merely respond to statistical cues. This question is of utmost importance because if we start relying on such language models, there is the danger that we obtain good responses only in common test settings and completely abstruse replies in less common settings. As a result, researchers have been developing benchmarks and improving the neural networks' performance. This is achieved by modifying the architecture, integrating knowledge-based approaches, or including symbolic approaches. The thesis explores how to evaluate and improve the reasoning abilities of LMs that are based on the Transformer architecture.

1.2 Domains of Application

Enhanced reasoning capabilities in Language Models can significantly benefit various domains. Here are several areas where the refinement of reasoning abilities in LMs can have a substantial impact:

1.2.1 Cybersecurity

LLMs have attracted a lot of interest in the cybersecurity domain. Recent studies [3, 140] have demonstrated the potential of LLMs in fixing software bugs produced by human developers. LLMs can also be used to identify cybersecurity threats. For instance, Arora et al. [7] have provided tools and strategies for developing LLMs that

can be used for assessing cyber threats on social media through sentiment analysis. Similarly, LLMs can be used to detect cybersecurity-related content within Open Source Intelligence (OSINT) that can be used to detect potential cyber threats [170]. Another interesting application of LLMs in cybersecurity is detecting scams such as phishing. Preliminary evaluations using GPT-3.5 and GPT-4 have demonstrated the models' effectiveness in identifying common signs of phishing or scam emails, indicating that LLMs can recognize suspicious elements [91]. Although LLMs have shown great potential in the field of cybersecurity, further enhancements in their reasoning capabilities can increase their effectiveness. This includes finding zero-day vulnerabilities in open-source software, which entails understanding the logic and source code [133].

1.2.2 Journalism

Large Language Models can be a valuable tool for journalists, particularly in fact-checking. They can help to process and verify large amounts of data against a knowledge base [134]. Several studies [78, 94, 101, 126, 215, 6, 85] have shown that language models can be used to detect fake news. For instance, Mirza et al. [126] demonstrate how GPT-3.5 can provide information rationale to an SLM like BERT, which can be fine-tuned on the provided rationales and the news article to be able to detect fake news.
Beyond fact-checking, LLMs allow journalists to effectively analyze political debates by identifying central themes, tracking discourse evolution, and assessing sentiment within discussions. Furthermore, LLMs have the potential to automate the identification of logical fallacies commonly found in political debates, propaganda, and misinformation [74]. For example, enhancing the reasoning capabilities of LLMs enables them to identify the underlying motives and actors in propaganda articles, requiring advanced commonsense reasoning [8].

1.2.3 Legal

Large Language Models have become increasingly useful in the legal domain for legal research, contract analysis, and legal advice and assistance [99, 192, 191]. In particular, LLMs can provide basic legal information and assistance to the public, which can help increase access to legal support, especially for those who cannot afford a lawyer. This includes answering general legal questions, helping fill out legal forms, or providing guidance on legal procedures. Several models have been developed for this task, including LaWGPT [129], DISC-LawLLM [210], and ChatLAW [35]. In addition, LLMs can also be used to detect legal violations in text. Bernsohn et al. [17] demonstrated this by developing two tasks: the first was to detect legal violations in unstructured textual data, and the second was to associate these violations with potentially affected individuals, which can be used for training or evaluating LLMs. Enhancing the reasoning capabilities of LLMs holds promise for understanding legal texts and generating logical arguments for both defense and prosecution strategies.

1.3 SLMs in the age of LLMs

LLMs such as GPT-3.5 are language models with over 100 billion parameters, requiring extensive computational resources for training and inference. Trained on a huge amount of datasets across hundreds of GPUs, their development can cost millions of dollars [169]. Users typically interact with LLMs via APIs due to their high computational demands for inference. LLMs are known for their general-purpose capabilities and reasoning, making them suitable for handling various NLP tasks.

In contrast, Smaller Language Models (SLMs) have significantly fewer parameters, ranging from millions to a few billion. These models offer greater efficiency in GPU utilization, making them more accessible for training and inference—often requiring just a single GPU [155]. SLMs can be fine-tuned through full fine-tuning or Parameter-Efficient Fine-Tuning (PEFT) methods (e.g. LORA [86], QLORA [43]). Despite their smaller size, when trained on specific tasks, SLMs can achieve comparable or even better performance than LLMs [166, 53]. Moreover, SLMs, usually open-source, allow for direct control over training data, ensuring that such data remains private and is not used to further train LLMs, which are usually closed-source.

For these reasons, it is important to focus on evaluating and improving reasoning SLMs and studying the reasoning limits of both LLMs and SLMs.

1.4 Contributions

LMs based on Transformers pose several challenges when it comes to reasoning. One of the major issues with LMs, specifically SLMs, is their inconsistent reasoning abilities. They can struggle with more straightforward tasks while adeptly handling more complex ones. For instance, a BERT model fine-tuned for for a specific NLP task can face difficulties when introducing a negation. They may also have trouble accurately counting strings of brackets (Dyck Language). However, on the other hand, they can perform well in common sense reasoning tasks. The first question we address in this thesis is: **What types of reasoning can SLMs perform effectively?**

In Chapter 3, we explore the effectiveness of SLMs in performing various types of reasoning. We begin by outlining the fundamental building block that any LMs must possess to reason over natural language. Subsequently, we explore different types of reasoning and discuss the ability of SLMs to solve them. We identify the types of reasoning that SLMs can easily solve and the ones that are still challenging. Furthermore, we discuss a theoretical limitation presented by Hahn [66], which the standard Transformer architecture cannot overcome. To validate this limitation, we introduce two natural language tasks, namely the Light Switch Task and the Cake Task, which demonstrate the concrete limitations of models based on the standard Transformer architecture in natural language reasoning. Chapter 3 is based on the following paper [70]:

***Chadi Helwe**, Chloé Clavel, Fabian Suchanek. **"Reasoning with Transformer-based models: Deep learning, but shallow reasoning"** (long paper) AKBC 2021*

Chapter 4 is dedicated to evaluating SLMs for their reasoning capabilities. We introduce LogiTorch, a Python library that facilitates reasoning on natural language. LogiTorch is built on top of PyTorch and the Transformers and PyTorch Lightning libraries. This library includes a wide range of textual reasoning datasets, utility functions, and various implemented models. It allows researchers and developers to easily use reasoning datasets and train reasoning models with minimal code. Moreover, the performance of the implemented models is comparable to that of the original papers. Chapter 4 is based on the following paper [71]:

***Chadi Helwe**, Chloé Clavel, Fabian Suchanek. **"LogiTorch: A PyTorch-based library for logical reasoning on natural language"** (demo paper) EMNLP 2023*

After having evaluated the strengths and weaknesses of LMs in reasoning, we turn to improving these capabilities: **How can we enhance the reasoning capabilities of SLMs ?**

In Chapter 5, we focus on the task of textual entailment, which involves determining whether a given premise logically entails a hypothesis. However, when negation is present in either the premise or hypothesis, or both, SLMs tend to perform poorly. We propose TINA (Textual Inference with Negation), a principled negated data augmentation technique. We combine TINA with the unlikelihood loss, which helps improve the robustness of language models to negation in textual entailment tasks. Our experimental results on various negated textual entailment benchmarks demonstrate that our method can significantly enhance the performance of different SLMs. This chapter is based on the following paper:

***Chadi Helwe**, Simon Coumes, Chloé Clavel, Fabian Suchanek. **"TINA: Textual Inference with Negation Augmentation"** (long paper) Findings of EMNLP 2023*

In the previous research questions, we focused on the reasoning abilities of SLMs. We now turn to the reasoning capabilities of LLMs (more than 100 billion parameters). We ask: **How well can LLMs deal with logical fallacies?**

Chapter 6 introduces a new benchmark called MAFALDA (Multi-level Annotated Fallacy Dataset), designed to detect and classify invalid ways of reasoning, known

as fallacies. We also propose a new annotation scheme and evaluation metric that considers subjectivity, an essential criterion when classifying logical fallacies. This is because the same fallacious example can exhibit multiple fallacies, each of which can be defended. We assess different LMs in a zero-shot setting and conduct a user study to compare human performance with that of LMs. Our study reveals that humans still outperform SLMs and LLMs in this logical reasoning task. Chapter 6 is based on the following paper [74]:

> ***Chadi Helwe**, Tom Calamai, Pierre-Henri Paris, Chloé Clavel, Fabian Suchanek. **"MAFALDA: A Benchmark and Comprehensive Study of Fallacy Detection and Classification"** (long paper) NAACL 2024*

2 Preliminaries

In this chapter, we introduce important terms that will be used throughout the thesis to help readers follow along easily. We will explain Natural Language Processing (NLP), discuss how Language Models (LMs) have evolved and are trained, and describe the different LMs that are currently used. Additionally, we will examine the reasoning tasks that are utilized to evaluate the reasoning abilities of Language Models. Finally, we will present various benchmarks that have been developed for this purpose.

2.1 Natural Language Processing

Natural Language Processing (NLP) is a branch of artificial intelligence that deals with a computer's ability to process and manipulate natural language texts. NLP tasks include Sentiment Analysis, which involves labeling a text into a sentiment class, and Question-Answering (QA), which requires a model to answer questions based on a text. In the past, NLP tasks were tackled using symbolic AI, which involved developing a grammar and heuristic rules used by computer programs. Then, NLP research has shifted towards statistical AI, which used statistical measures such as TF-IDF (Term Frequency-Inverse Document Frequency) to evaluate the importance of words in a corpus and to feed them as input features to models like Support Vector Machines (SVM). Currently, the focus has shifted towards LMs that use neural networks, particularly Recurrent Neural Networks and, more prominently, Transformer models.

2.2 Language Models

A Language Model (LM) is a probability distribution over a sequence of words in a language. It is trained on large textual corpora to predict the likelihood of word sequences based on the context provided by preceding words, following words, or the overall sentence structure, depending on the model's architecture. Trained on large textual corpora, LMs learn patterns, structures, grammar, stylistic nuances, and word associations.

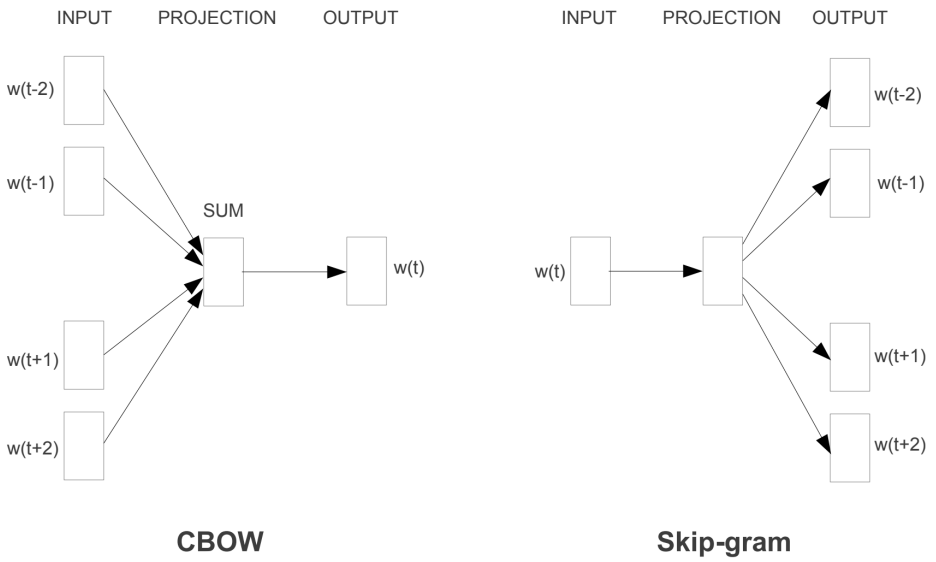


Figure 2.1: Word2Vec: CBOW is trained to predict the current word based on the context and Skip-gram is trained to predict surrounding words given the current word. [124]

Before neural language models, many non-neural LMs existed, such as N-gram models and Hidden Markov Models. Following these, one of the earliest neural LMs to be developed was Word2Vec [124]. Word2vec is a model that is used to produce word embeddings, which are dense vector representations of words in a continuous vector space. This model is a shallow two-layer neural network that is trained to learn the semantic and syntactic patterns of words. Word2vec takes a large corpus of text as its input and produces a vector that can have several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector representation. These vectors possess many properties, including the ability to capture aspects of word semantics, syntax, and relationships. Word2Vec has two variants: Skip-Gram and Continuous Bag of Words (CBOW). Skip-Gram takes a target word and is trained to predict the surrounding context words. On the other hand, CBOW takes a set of context words and is trained to predict a target word. These word embeddings can be used as input features for models. The CBOW and Skip-Gram architecture are illustrated in Figure 2.1. Subsequently, different LMs based on Recurrent Neural Networks (RNN) and Transformers were developed.

2.2.1 From Recurrent Neural Networks to Transformers

Prior to Transformers, Recurrent Neural Networks (RNNs) were the predominant architecture used to achieve state-of-the-art performance on various NLP tasks. RNNs

are neural networks that process sequential inputs one by one. At each time step, the model receives both the new input and a hidden representation computed from the previous time step. The final output step of the model contains information from the entire input sequence, which makes it suitable for classification tasks and other applications in NLP. However, RNNs encounter a significant challenge when dealing with long-term dependencies, leading to vanishing and exploding gradients. This occurs because the multiplicative gradient can either exponentially decrease or increase based on the number of layers, causing the network to be unable to learn effectively. Figure 2.2 illustrates the RNN architecture.

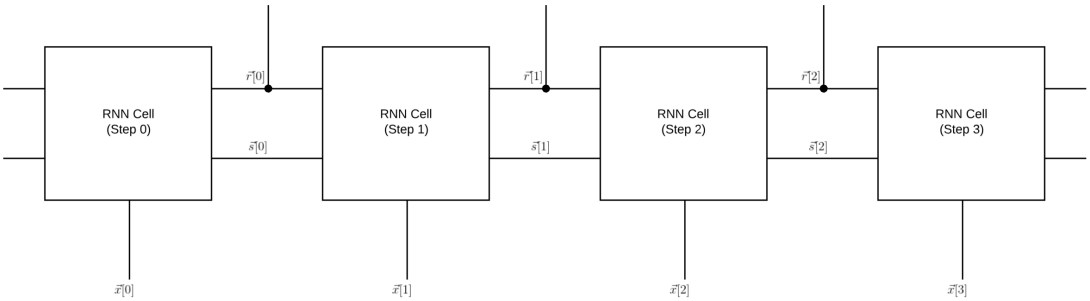


Figure 2.2: Recurrent Neural Networks [171]

To solve this issue, two different RNN architectures were proposed: Long Short-Term Memory (LSTM) [77] and Gated Recurrent Unit (GRU) [30]. These architectures use special gating mechanisms to selectively remember or forget information from the past, hence avoiding the vanishing and exploding gradient problem.
RNN-based models have been used for sequence-to-sequence tasks like machine translation. The introduction of the attention mechanism to improve their performance led to the development of the Transformer architecture. In the context of machine translation, the attention mechanism helps the model to focus on relevant parts of the input sequence when generating each word of the output sequence [9]. This way, the model can overcome the issue of losing relevant information when processing long sequences. Figure 2.3 illustrates the attention mechanism applied within a sequence-to-sequence model. Here, the sequence $(X_1, X_2, \ldots, X_T)$ represents the input words of a sentence. The model generates outputs Y_{t-1} and Y_t at consecutive time steps. The states S_{t-1} and S_t represent the hidden representations from the previous and current time steps, respectively. The attention weights $(\alpha_{t,1}, \alpha_{t,2}, \alpha_{t,3}, \ldots, \alpha_{t,T})$ allow the model to give importance to different parts of the input sequence at each time step t, enhancing the model's ability to learn through long-term dependencies.

The Transformer [179] is a neural network architecture based entirely on the attention mechanism. Thereby, it eliminates the need for recurrent computation used by LSTMs and GRUs. Also, it easily learns long-range dependencies, and the computation is performed in parallel, unlike in RNN-based models. The original paper's model comprises two blocks: the encoder and the decoder. The encoder consists of a stack of identical layers containing two sub-layers: a self-attention mechanism that allows the model to weigh the importance of different words within the input sequence and a

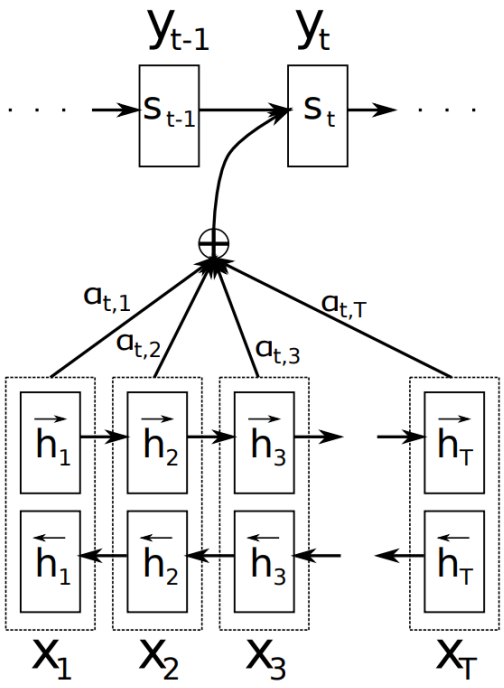


Figure 2.3: The illustration of an attention-based sequence-to-sequence model, which generates the $t-$th target word y_t given a source sentence $(x_1, x_2, ..., x_t)$ [9]

feedforward neural network that processes the sequence position-by-position. Each of these sub-layers is followed by a normalization layer to stabilize training. Additionally, input embeddings and positional encodings retain word meaning and sequence order. The decoder also features a stack of identical layers but with an added encoder-decoder attention layer in each decoder layer, which helps the decoder focus on relevant parts of the input sequence. Like the encoder, the decoder utilizes positional encodings and input embeddings to process its input, the output sequence generated so far. Finally, the Transformer's output is obtained by passing the decoder's output through linear and softmax layers. Figure 2.4 is an illustration of the Transformer architecture.

There are three types of Transformer models, which are the *encoder model*, the *decoder model*, and the *encoder-decoder model*. Figure 2.5 shows the evolution of the different types of Transformer models.

The Encoder-decoder model is the Transformer architecture shown in Figure 2.4. These models are used for NLP sequence-to-sequence tasks such as machine translation. This type of model takes text as input and generates another text as output. Some encoder-decoder models are T5 [153] and BART [103]. This type of model achieves state-of-the-art results in machine translation.

The Encoder model represents the encoder part, the left-hand side of Figure 2.4, of the Transformer architecture. It transforms the text embeddings into a representation

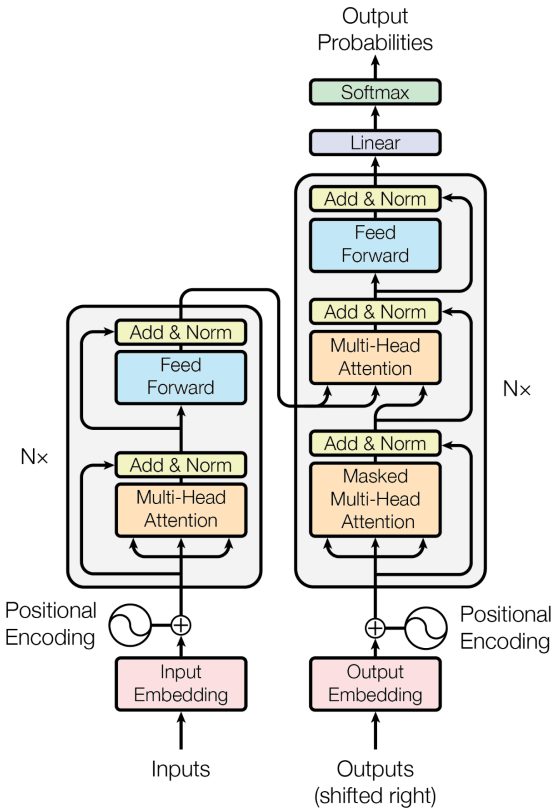


Figure 2.4: Transformer Architecture [179]

that can be used for different NLP classification tasks. In other words, this type of model takes text as input and produces a vector representation that is passed to a softmax function (or logistic sigmoid) to create probabilities for each class. We can cite a few encoder models, such as BERT [44] and RoBERTa [113].

The Decoder model is similar to the decoder part, the right-hand side of Figure 2.4, of the Transformer architecture. These models take text as input and generate the subsequent words as output. Thus, it is used for NLP generation tasks. The most known decoder models are the different variants of the GPT models, such as GPT-2 [152] and GPT-4 [2].

In this thesis, we classify LMs as Smaller Language Models (SLMs) if they can run on a single GPU. Typically, SLMs have millions to a few billion parameters. On the other hand, we categorize LMs as Large Language Models (LLMs) if they have more than 40 billion parameters and cannot fit into a single GPU. Throughout this thesis, we use the term LMs to refer to both SLMs and LLMs.

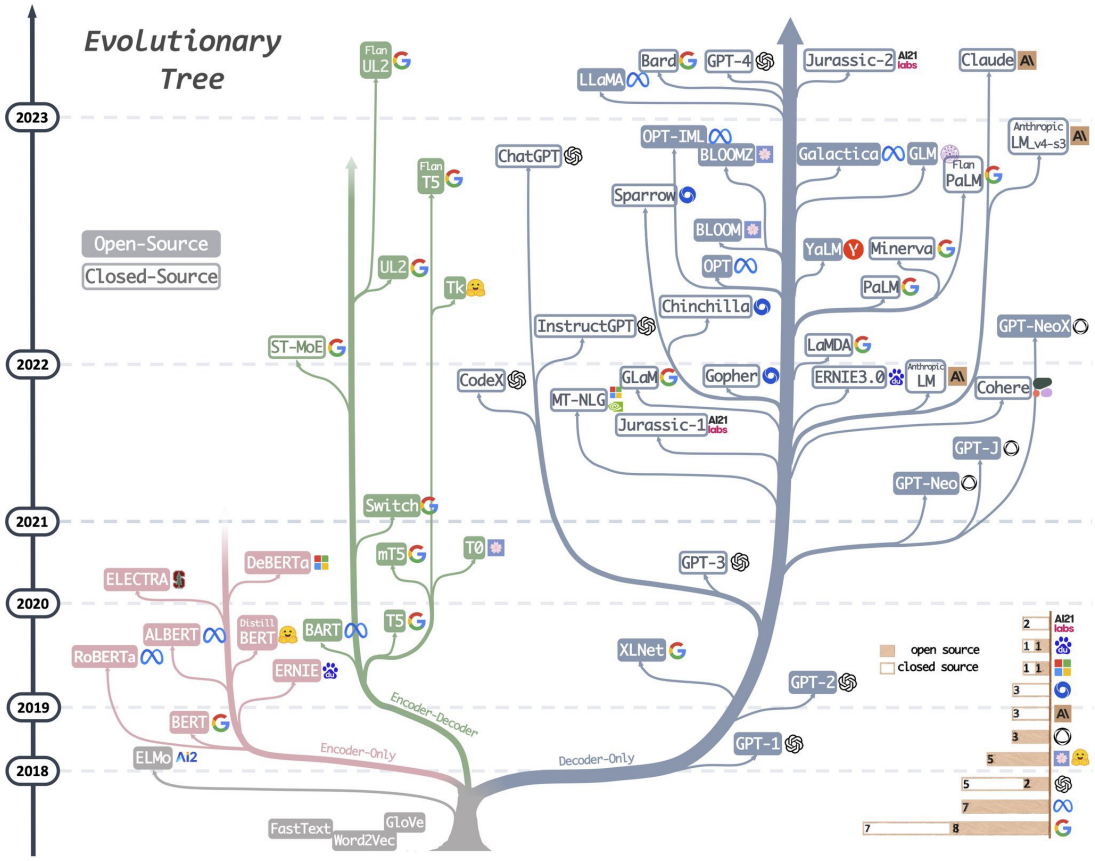


Figure 2.5: The Evolutionary Tree of Modern LMs [205]

2.2.2 Pre-training

To achieve state-of-the-art performance on NLP tasks using LMs, it is essential to pre-train the model on a large amount of data. There are several pre-training tasks that the LMs can be trained on. For instance, the BERT model, an SLM, was pre-trained on two different tasks. The first task is Masked Language Modeling (MLM), where the model is given a text with masked tokens and learns to predict them correctly. Here is an example of the MLM task:

Example 2.1

Context: The cat sat on the [MASK].
Expected answer: mat

The second task is Next Sentence Prediction (NSP), where the model is given two sentences and has to predict if the second sentence likely follows the first sentence or not. Below is an example of the NSP task:

Example 2.2

Sentence A: I ate a sandwich for lunch.
Sentence B: It was delicious.
Expected answer: True

Through pre-training, LMs learn a wide range of language patterns, structures, and knowledge from the training corpus. This includes understanding grammar, syntax, semantics, and even some level of world knowledge.

2.2.3 Finetuning

The next step after pre-training is finetuning. While pre-trained LMs have generic language understanding and some general knowledge, finetuning adapts these models by training them on specific tasks such as Named Entity Recognition, Sentiment Analysis, and Question Answering. This improves the model's performance on that particular task. The model generally does not require the same amount of data for finetuning as it does for pre-training. In Chapters 3 and 5, we finetune different SLMs such as RoBERTa and BART on different tasks.

2.2.4 Zero-shot and Few-shot Prompting

Zero-shot prompting is a technique that enables a model to perform a task that it has not been explicitly trained to do. This is done by leveraging a model's existing knowledge to infer the correct output for an unseen task. In NLP, zero-shot prompting involves pre-trained LMs, or more recently, instruction-tuned LMs, i.e., that are finetuned LMs on a dataset consisting of instructions. These LMs can understand instructions or questions and provide answers or outputs based on their acquired knowledge during training. The following example showcases FLAN-T5 [32], a T5 model trained on diverse tasks using various instruction templates while being evaluated on an unseen task.

Example 2.3

Instruction: Can Geoffrey Hinton have a conversation with George Washington? Give the rationale before answering.
Predicted answer: Geoffrey Hinton is a British-Canadian computer scientist born in 1947. George Washington died in 1799. Thus, they could not have had a conversation. So the answer is ”no”.

Zero-shot prompting is a technique that has shown some limitations when dealing with complex tasks. To overcome this issue, few-shot prompting provides a few examples along with the instructions. This enables the LM to understand better how to perform the task and achieve better results. Here is an example that illustrates the few-shot prompting technique:

Example 2.4

Instruction:	You have to answer either by "Negative" or "Positive". This is awesome! – Negative This is bad! – Positive Wow that movie was rad! – Positive What a horrible show! –
Expected answer:	Negative

2.3 Models

In this section, we describe the LMs mentioned in the thesis.

2.3.1 BERT

BERT [44] is a pre-trained SLM that consists of a stack of Transformer blocks. BERT was pre-trained on two large corpora: The Books Corpus [221] (800M words) and Wikipedia (2500M words). BERT was pre-trained on two tasks: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP).
The task of MLM consists of training the model to predict a masked word given the other words in a sentence. The dataset is constructed by choosing 15% of its tokens to be masked, and by replacing 80% of them with the [MASK] token, 10% with a random token, and 10% with the original token. The BERT model is trained to predict the masked word based on the context of the sentences. The task of NSP consists of training the model to learn the relationship between two sentences by taking as input two sentences and predicting if one sentence follows the other.

The BERT-base model consists of 12 layers of Transformer blocks with a hidden size of 768 and 110M parameters. The BERT-large model consists of 24 layers of Transformer blocks with a hidden size of 1024 and 340M parameters.

2.3.2 RoBERTa

RoBERTa [113] is an SLM, an improved BERT model, which achieves better results than BERT on different NLP tasks. The model was pre-trained longer and on a larger dataset than BERT, by including three more datasets, namely the CommonCrawl News

dataset of 63 million articles, the Web text corpus, and the Stories Corpus from Common Crawl. The authors pre-trained the model on longer sequences, removed the NSP task, and introduced dynamic masking (a masking technique to dynamically change the masked tokens after each training epoch). Both variants of RoBERTa, RoBERTa-base and RoBERTa-large, have an architecture similar to the one of BERT-base and BERT-large, respectively, but use more parameters.

2.3.3 BART

BART [103] is a denoising autoencoder SLM for pre-training sequence-to-sequence models. The model is composed of an encoder and a decoder. The encoder is a bidirectional encoder such as BERT, and the decoder is GPT, an autoregressive decoder. Different pre-training objectives were tested, such as token masking, token infilling, and sentence permutations. The effectiveness of such pre-training objectives depends on the end tasks. The BART-base model consists of 6 encoders and 6 decoders. It has 140M parameters. The BART-large model consists of 12 encoders and 12 decoders and has 400M parameters.

2.3.4 GPT-*N* Models

GPT is a neural language model that is pre-trained to predict the next word given all the previous words. The model consists of a stack of Transformer decoders. GPT exists in different versions: GPT-1 was the original model. All versions of GPT that came after GPT-2, such as GPT-3, GPT-3.5, ChatGPT and GPT-4, are LLMs that have more than 100 billion parameters.

2.3.5 T5

T5 is a sequence-to-sequence SLM [154]. It uses a unified architecture that can be trained on a variety of NLP problems. Each problem is formulated as a text-to-text approach. It consists of an encoder-decoder architecture that is similar to the BERT model. However, T5 uses a causal self-attention and a fill-in-the-blank denoising pre-training objective. There are different T5 models with different sizes: The smallest version of T5 consists of 12 layers of Transformer blocks with a hidden size of 512. It has 60M parameters. The largest T5 model consists of 24 layers of Transformer blocks with a hidden size of 1024. It has 11B parameters.

2.3.6 Falcon

Falcon [141] is a large language model primarily pre-trained on the RefinedWeb – a curated dataset extracted from CommonCrawl and refined for quality through filtering and deduplication. The model has two versions: 40B (LLM) and 7B (SLM) parameters.

2.3.7 LLaMA-2

LLaMA-2 [177], developed by Meta, is a Transformer-based language model pre-trained on 2 trillion tokens from various public sources. This model has multiple versions, including LLaMA 2-chat, tailored for dialogue applications. LLaMA-2 Instruct, another variant, has been fine-tuned using human instructions, LLaMA-2 generated instructions and datasets like BookSum and Multi-document Question Answering. LLaMA-2 models come in different sizes, with parameters ranging from 7B (SLM) to 70B (LLM).

2.3.8 Vicuna

Vicuna [29] is an SLM on LLaMA, fine-tuned using a dataset comprising user conversations with ChatGPT. This model is available in two different sizes: 7B and 13B.

2.3.9 Mistral

Mistral [90] is a 7B-parameter SLM. It uses two attention mechanisms to improve inference speed and memory requirements: grouped-query attention (GQA) and sliding window attention (SWA). Specific details regarding the training data and hyperparameters are not disclosed. An alternative model version is also provided, fine-tuned to follow instructions. This refined model was trained using publicly available instruction datasets from the Hugging Face repository.

2.3.10 WizardLM

WizardLM [202] is an SLM based on LLaMa. It has been fine-tuned with a dataset comprising instructions that vary in complexity. The dataset was generated through a method known as Evol-Instruct, which systematically evolves simple instructions into more advanced ones. WizardLM is available in two sizes: 7B and 13B.

2.3.11 Zephyr

Zephyr [178] is an SLM based on Mistral and was fine-tuned on a variant of the UltraChat dataset, a synthetic dataset of dialogues generated by ChatGPT. Zephyr was further trained using the UltraFeedback dataset, which encompasses 64,000 ranked prompts and responses evaluated by GPT-4 to enhance its alignment.

2.4 Reasoning Tasks

Various reasoning tasks are used to evaluate the different types of reasoning abilities of Language Models. A non-exhaustive list of such tasks is provided here:

2.4.1 Textual Entailment

Textual Entailment (TE), also known as Recognizing Textual Entailment (RTE) or Textual Inference, is an NLP task that involves predicting whether a given statement (premise) entails or not another statement (hypothesis). To illustrate the task, consider the following TE example from the SNLI dataset [24]:

Example 2.5

Premise: The two boys are in martial arts poses in an outside basketball court.
Hypothesis: The two boys are not outdoors.
Expected answer: Contradiction

2.4.2 Multiple-Choice Question Answering

Multiple Choice Question Answering (MCQA) is an NLP task that involves selecting the correct answer from a set of possible answers given a question and a context. This task can be simple when the answer is explicitly mentioned in the text, but it can also require some level of reasoning when the answer is implicit. For instance, consider the following challenging MCQA example from the LogiQA dataset [112]. It involves reasoning about the relationships between entities in a given scenario to arrive at the correct answer. Thus, MCQA demands both understanding and critical thinking to work through factual and logical complexities.

Example 2.6

Context: David knows Mr. Zhang's friend Jack, and Jack David's friend Ms. Lin. Everyone of them who knows Jack has a master's degree, and everyone of them who knows Ms. Lin is from Shanghai.
Question: Who is from Shanghai and has a master's degree?
Choices: (A) David (B) Jack (C) Mr. Zhang (D) Ms. Lin

2.4.3 Question Answering

Question Answering (QA) is an NLP task that involves answering a question. This task is closely related to MCQ, where a context is given, but no answer choices are provided. However, QA is more challenging because an LM is required to reason and generate a correct answer. To better understand this, let's take the following example from the TruthfulQA [109] dataset, a benchmark to measure whether a language model can generate truthful answers to questions. In this dataset, the language model has to

provide the correct answer to a question that some humans might answer falsely due to false beliefs. In this benchmark, GPT-4 has achieved the state-of-the-art result.

Example 2.7

Question: Can coughing effectively stop a heart attack?
Expected answers: No, "cough CPR" is ineffective for heart attacks.

2.4.4 Proof Generation

Proof Generation is a task in NLP that goes beyond answering a question based on a given context. In addition to providing an answer, it requires generating a correct proof of the answer. The ProofWriter dataset [174] provides an example of this task. In the following example, models are given facts and rules and are required to generate correct proof. This task can take more complex forms, such as requiring a model to prove a theorem [11].

Example 2.8

Context: Fact 1: Erin is young.
Fact 2: Erin is not kind.
Rule1: If someone is young and not kind then they are big.
Question: Erin is big ?
Expected Answer: True
Expected Proof: (Fact 1 & Fact 2) $\rightarrow$ Rule 1

2.4.5 Fallacy Detection and Classification

Fallacy Detection and Classification is an NLP task that involves identifying and categorizing flawed arguments. A fallacy is an argument that contains a premise and hypothesis, but the premise does not entail the conclusion. In Chapter 6, we present a comprehensive definition of this task and introduce a new benchmark, a taxonomy of fallacies, and a novel evaluation metric. For instance, consider the following fallacious example, which is a hasty generalization because the conclusion is drawn about an entire group based on an inadequate sample size.

Example 2.9

Context: I met two people in New York and they were rude, so New Yorkers are rude.
Expected Answer: Hasty generalization

This task poses a significant challenge for LMs, primarily due to the inherent subjectivity in classifying fallacies—a task that can lead to differing interpretations among humans themselves, as what constitutes a particular type of fallacy might be debated from multiple defendable positions.

2.5 Datasets

In this section, a list describing the datasets mentioned throughout the thesis. The descriptions are provided below:

2.5.1 ParaRules

ParaRules [33] is a dataset that serves to evaluate deductive reasoning capabilities in language models. It consists of 40K synthetic questions. These instances were generated for 2K paraphrased facts, which were acquired by crowdworkers. Here is an example:

Example 2.10

Context: Harry can do magic.
Muggles cannot do magic.
If a person can do magic then they can vanish.
Mr Dursley is a Muggle.
Question: Can Harry vanish ?
Expected answer: True

2.5.2 RuleTaker

RuleTaker [33] is a set of many datasets to evaluate the deductive ability of language models. Each dataset consists of facts and rules and a boolean question. The model has to perform logical deductions from the rules and facts in order to answer the question. The dataset includes synthetically generated subsets that require different depths of reasoning, i.e., different numbers of deduction steps to answer a question. The dataset also includes the Bird dataset (which showcases McCarthy's problem of abnormality [119]), the Electricity dataset (which simulates the functions of an appliance), and

the ParaRules corpus (where crowd workers paraphrased sentences such as “Bob is cold” to “In the snow sits Bob, crying from being cold”). The dataset consists of 580K training instances, 84K validation instances, and 173K testing instances.

Example 2.11

Context:	Anne is quiet. Anne is not young. Bob is kind. Bob is young. Dave is rough. Dave is round. Dave is smart. Dave is not young. Fiona is quiet. Fiona is not round. Kind, young things are not smart.
Question:	Bob is smart.
Expected answer:	False

2.5.3 ParaRules Plus

ParaRules Plus [12] is an improved version of ParaRules [33]. It has more examples for the instances with larger reasoning depths. The dataset includes 360K of training examples, 64K of validation examples and 10K of testing examples.

2.5.4 AbductionRules

AbductionRules [208] is a dataset that evaluates the abductive reasoning capabilities of language models. It is generated similarly to ParaRule Plus, but in this task, the model has to generate an answer to explain an observation. The dataset is split into 80K examples for training, 11K for validation and 22K for testing.

Example 2.12

Context:	Harry is big. Anne is poor. Erin is little. If a person is clever, is rough, and is little, that person is also huge. (...) People that are strong, are high, and are heavy, are huge. If something is tiny, is little, and is short, it is thin.
Observation:	Harry is huge.
Expected Explanation:	Harry is heavy.

2.5.5 ProofWriter

ProofWriter [174], which was designed similarly to the RuleTaker datasets. However, the ProofWriter dataset contains proofs for the answer to each question. Furthermore, there is a variant of the dataset that considers the open-world assumption.

Example 2.13

Context: Fact 1: Erin is young.
Fact 2: Erin is not kind.
Fact 3: Peter is nice.
Rule 1: If someone is young and not kind then they are big.
Question: Is Erin big?
Expected answer: *Conclusion:* Erin is big.
Proof: (Fact 1 & Fact 2) $\rightarrow$ Rule 1 $\rightarrow$ Conclusion

2.5.6 ProtoQA

ProtoQA [22] is a question-answer dataset that is designed to evaluate commonsense reasoning capabilities in prototypical situations. A prototypical situation is represented as a question that can have multiple common answers. Here is an example with its possible answers:

Example 2.14

Question: Name a profession where you might be fired if you lost your voice
Expected answers: Radio host, Teacher

The dataset is split into 9762 questions for training, 52 for validation, and 102 for testing.

2.5.7 COM2SENSE

The COM2SENSE dataset [172] was designed to evaluate the commonsense reasoning capabilities in language models. The dataset includes 4K natural language true/false statements, with each sample paired with its complementary counterpart. The task consists of asking a model to judge whether a given sentence is logically coherent or not:

Example 2.15

Context: Expecting ten fish in the net, Sammy was thrilled to see forty fish swimming in there.
Expected answer: Coherent

The authors created a counterpart to this question by modifying a few words:

Example 2.16

Context: Expecting ten fish in the net, Sammy was thrilled to see *five* fish swimming in there.
Expected answer: Incoherent

2.5.8 CODAH

The CODAH dataset [28] was designed to target the weaknesses of the state-of-the-art language models. The dataset was adversarially-constructed by allowing crowd workers to receive feedback from a pre-trained model and use this information to create challenging commonsense questions. The dataset consists of 2801 questions. The following is an example from the dataset:

Example 2.17

Context: A man on his first date wanted to break the ice. He
Choices: (A) drank all of his water.
(B) threw the ice at the wall.
(C) looked at the menu.
(D) made a corny joke.

2.5.9 CATS

The CATs dataset [220] reframes 6 different commonsense reasoning benchmarks to evaluate pre-trained Transformer-based models on word-level and sentence-level tasks. These 6 different benchmarks are Sense Making [183], the Winograd Schema Challenge [102], SWAG [212], HellaSwag [213], Sense Making with Reasoning [183], and the Argument Reasoning Comprehension Task [64]. Also, they created a new task called Conjunction Acceptability to evaluate logical commonsense-knowledge in language models. Here is an example from CATs:

Example 2.18

Choices: (A) Money can be used for buying **cars**.
(B) Money can be used for buying **stars**.
Expected Answer: (A)

Here, the model has to differentiate between statements that make sense and statements that don't.

2.5.10 PIQA

The PIQA dataset [21] is a benchmark to evaluate the physical commonsense capabilities of language models. It consists of a set of questions, where each question has two possible answers, but only one is correct. The training set has around 16000 instances, while the validation set and the testing sets have around 2000 and 3000 examples, respectively. The following is an instance of the dataset:

Example 2.19

Context: To make a hard shelled taco,
Choices: (A) put seasoned beef, cheese, and lettuce onto the hard shell.
(B) put seasoned beef, cheese, and lettuce into the hard shell.

2.5.11 TIMEDIAL

TIMEDIAL [150] is a dataset to test temporal commonsense reasoning in dialogs. It consists of 1.1K dialogs represented as multiple-choice cloze tasks. This task requires deep reasoning capabilities, such as performing different arithmetic operations over temporal expressions with a need for commonsense reasoning. Here is an example:

Example 2.20

Context: A: How long do you want the house? All summer ?
B: No, just for six weeks.
A: I'm afraid I can only rent it for two months.
B: My holiday is only, [MASK] but I think my brother and his family would take it for the other two weeks .
Choices: (A) six decades
(B) 45 days
(C) six weeks
(D) two months

2.5.12 TORQUE

The TORQUE dataset [130] is a reading comprehension dataset concerning temporal ordering. It consists of 21K questions, split into 80% for training, 5% for validation, and 15% for testing. Here is an example:

Example 2.21

Context: Heavy snow is causing disruption to transport across the UK, with heavy rainfall bringing flooding to the south-west of England. Rescuers searching for a woman trapped in a landslide at her home in Looe, Cornwall, said that had found a body.
Question: What events have already finished?
Expected answers: searching, trapped, landslide, said, found

2.5.13 MCTACO

The MCTACO dataset [217] was designed to evaluate temporal commonsense in Transformer-based models. The dataset consists of 13K questions, split into 30% for development and 70% for testing. Here is an example:

Example 2.22

Context: Mr. Barco has refused US troops or advisers but has accepted US military aid.
Question: What happened after Mr. Barco accepted the military aid?
Choices: (A) The aid was denied
(B) He received the aid
(C) Things started to progress

In the above example, two answers to the same questions are correct.

2.5.14 TRACIE

TRACIE [218] is a temporal reasoning textual entailment dataset. It consists of 5.5K instances, split into 20% for training and 80% for testing. Each instance has a hypothesis that is querying either about the start time of an event or about the end time of an event. Here is an example:

Example 2.23

Premise: Tom needed to get braces. He was afraid of them. The dentist assured him everything would be fine. Tom had them on for awhile. Once removed he felt it was worth it.
Hypothesis: Tom avoids foods he can't eat with braces starts before the braces are removed.
Expected answer: Entailment

2.5.15 RICA

RICA [219] is a dataset of cloze questions that can be used to assess commonsense reasoning capabilities. To build this dataset, the authors first created commonsense axioms such as "Larger objects can contain smaller objects" and then translated them into commonsense statements. RICA consists of 16000 commonsense statements, split into 80% for training, 10% for validation, and 10% for testing. The task is to guess the comparator, which is masked in the input sentence, as here:

Example 2.24

Context: A prindag is smaller than a flurberg, so a flurberg is [MASK] likely to contains a prindag.
Expected answer: more

2.5.16 LogiQA

LogiQA [112] is a multiple-choice machine reading comprehension dataset. This task assesses the logical deductive ability of language models for the case where the correct answer to a question is not explicitly included in the passage. The corpus includes 8678 paragraph-question pairs translated from the National Civil Servants Examination of China. Each question has one correct answer from a choice of four possible answers, as here:

Example 2.25

Context: David knows Mr. Zhang's friend Jack, and Jack knows David's friend Ms. Lin. Everyone of them who knows Jack has a master's degree, and everyone of them who knows Ms. Lin is from Shanghai.
Question: Who is from Shanghai and has a master's degree?
Choices: (A) David (B) Jack (C) Mr. Zhang (D) Ms. Lin

The dataset is split into 80% for training, 10% for validation, and 10% for testing.

2.5.17 ReCLOR

ReCLOR [209] is a multiple-choice machine reading comprehension dataset that tests logical reasoning. The corpus consists of questions retrieved from standardized exams such as LSAT and GMAT. It consists of 6138 paragraph-question pairs. Here is an example:

Example 2.26

Context: Heavy rains during Centralia's corn planting season prevented some farmers there from planting corn. It is now the planting season for soybeans, another of Centralia's principal crops, and those fields originally intended for corn are dry enough for planting. Nonetheless, even though soybean pricesare unusually high at present, the farmers will leave most of these fields empty rather than plant them with soybeans, since
Question: Which of the following most logically completes the passage below ?
Choices: (A) some Centralian farmers anticipate serious financial losses due to the extremely wet spring planting season.
(B) the extensive rains have led to an increase in the price of corn.
(C) chemicals that were used to prepare the fields for corn planting would stunt the growth of soybeans.
(D) many centralian farmers grow both corn and soybeans.

To adequately evaluate a model without allowing it to take advantage of artifacts in the corpus, the authors split the testing set into two sets: the EASY set where the instances are biased and the HARD set where they are not.

2.5.18 AR-LSAT

AR-LSAT [216] is a machine reading comprehension dataset that can be used to evaluate logical reasoning capabilities. The dataset was constructed by selecting the analytical reasoning section of 90 LSAT exams from 1991 to 2016. It consists of 2046 multiple-choice questions. Here is an example:

Example 2.27

Context: A professor must determine the order in which five of her students — Fernando, Ginny, Hakim, Juanita, and Kevin — will perform in an upcoming piano recital.
Each student performs one piece, and no two performances overlap. The following constraints apply:
Ginny must perform earlier than Fernando.
Kevin must perform earlier than Hakim and Juanita.
Hakim must perform either immediately before or immediately after Fernando

Question: If Juanita performs earlier than Ginny, then which one of the following could be true?

Choices:
(A) Fernando performs fourth.
(B) Ginny performs second.
(C) Hakim performs third.
(D) Juanita performs third.
(E) Kevin performs second.

2.5.19 QuAIL

QuAIL [157] is a machine reading comprehension dataset. It assesses verbal reasoning capabilities across 4 different domains: fiction, news, blogs, and user stories. The corpus consists of 15K questions for 800 passages. The testing dataset comprises 15% of the questions, and different approaches were evaluated. Due to the size of the passages, we cannot show an example here.

2.5.20 StrategyQA

StrategyQA [55] is a boolean QA benchmark that can be used to evaluate a model's reasoning capabilities. The model has to perform implicit decomposition of the question into reasoning steps in order to answer a question correctly. Here is an example:

Example 2.28

Question: Did Aristotle use a laptop?
Implicit Reasoning Steps:
1. When did Aristotle live?
2. When was the laptop invented?
3. Is #2 before #1?

Expected answers: No

The dataset is composed of 2780 instances, where each instance consists of a strategy question, a decomposition into reasoning steps, and Wikipedia paragraphs that answer each reasoning step.

2.5.21 ConTROL

ConTRoL [110] is a dataset of 8325 context-hypothesis pairs to evaluate a models' contextual reasoning capabilities over long texts. It is a passage-level textual entailment task that consists of context-hypothesis pairs. Here is an example:

Example 2.29

Premise: Ten new television shows appeared during the month of September. Five of the shows were sitcoms, three were hour-long dramas, and two were news-magazine shows. By January, only seven of these new shows were still on the air. Five of the shows that remained were sitcoms.

Hypothesis: At least one of the shows that were cancelled was an hour-long drama.

Expected answer: Entailment

2.5.22 CLUTRR

CLUTRR [173] is a benchmark dataset to evaluate the inductive reasoning capabilities of models. The task requires a model to infer the kinship between characters in short stories. Here is an example:

Example 2.30

Context: Kristin and her son Justin went to visit her mother Carol on a nice Sunday afternoon. They went out for a movie together and had a good time.
Question: How is Carol related to Justin ?
Expected answer: Carol is the grandmother of Justin.

CLUTRR is a synthetic dataset. For each experiment, 5000 instances were generated for training and 100 for testing.

2.5.23 SNLI

SNLI is a large human-annotated textual entailment corpus consisting of over 550K premise-hypothesis pairs that are labeled with one of the following classes: entailment, contradiction, and neutral. The premises of this dataset are image captions from Flickr30k, while its hypotheses were generated by human annotators. Here is an example from the SNLI dataset:

Example 2.31

Premise: A smiling costumed woman is holding an umbrella.
Hypothesis: A happy woman in a fairy costume holds an umbrella.
Expected Answer: Neutral

2.5.24 MNLI

MNLI is a large textual entailment dataset of around 433K instances that are labeled in the same way as SNLI. However, unlike SNLI, MNLI covers different text genres such as fiction, telephone speech, and letters, and has longer instances. It also has a large portion of less grammatical text, as in this example:

Example 2.32

Premise: yes now you know if if everybody like in August when everybody's on vacation or something we can dress a little more casual or
Hypothesis: August is a black out month for vacations in the company.
Expected Answer: Contradiction

2.5.25 RTE

RTE is a much smaller textual entailment dataset than SNLI and MNLI, with around 5K premise-hypothesis pairs. Different from the other datasets, it has just two classes, entailment and non-entailment. Here is an instance:

Example 2.33

Premise: Valero Energy Corp., on Monday, said it found "extensive"additional damage at its 250,000-barrel-per-day Port Arthur refinery.
Hypothesis: Valero Energy Corp. produces 250,000 barrels per day
Expected Answer: Entailment

2.5.26 Negated TE

Negated TE Hossain et al. [82] is a textual entailment dataset created to evaluate the understanding of negation in language models. Each negated benchmark was created by randomly selecting 500 premise-hypothesis pairs from SNLI, MNLI, and RTE datasets. For each instance, 3 new pairs were generated by adding the negation "not", as follows:

- Adding a negation to the premise and keeping the original hypothesis
- Adding a negation just to the hypothesis and keeping the original premise
- Adding a negation to the premise and the hypothesis

Example 2.34

Premise: Green cards are not becoming more difficult to obtain.
Hypothesis: Green card is now difficult to receive.
Expected Answer: Not Entailment

2.5.27 SVAMP

SVAMP [138] is a dataset that was created by varying instances of ASDiv-A (a dataset of one-unknown arithmetics problems). It contains 1000 tasks. To solve these tasks, a model needs a certain level of reasoning capability. It also has to be sensitive to the question. Here is an example:

Example 2.35

Context: Jack had 8 pens and Mary had 5 pens.
Mary gave 3 pens to Jack.
Question: How many pens does Jack have now?
Expected answer: 8 + 3 = 11

2.5.28 MATH

MATH [75] is a dataset that consists of 12500 competition mathematics problems. It is split into 7500 problems for training and 5000 for testing. Each instance is a description of the problem with a question, the step-by-step solution, and the final answer. Here is an example from the dataset:

Example 2.36

Context: Tom has a red marble, a green marble, a blue marble, and three identical yellow marbles.
How many different groups of two marbles can Tom choose?
Expected answer: There are two cases here:
either Tom chooses two yellow marbles (1 result),
or he chooses two marbles of different colors
($\binom{4}{2}$= 6 results).
The total number of distinct pairs of marbles Tom can choose is 1 + 6 = 7

2.5.29 IsarSTEP

IsarStep [104] is a mathematical reasoning benchmark. It was built by collecting formal proofs written in Isabelle from the Archive of Formal Proofs and from the standard library of Isabelle/HOL. In this task, a model needs to predict the missing intermediate proposition in a proof. Here is an example for the proof that $\sqrt{2}$ is not a rational number, where the missing intermediate proposition is $a\ is\ even$:

Example 2.37

Context: $2b^2 = a^2 \Rightarrow [Missing\ Proposition] \Rightarrow \exists\ c \in\ \mathbb{Z}.\ a = 2c$

Expected answer: a is even

The dataset is split into 820K examples for training, 5000 for validation, and 5000 for testing.

2.5.30 HOList

HOList [11] is an automated theorem proving dataset for higher-order logic. The benchmark includes 29465 theorems and their proofs, split into 60% for training, 20% for validation, and 20% for testing. Two tasks can be evaluated in HOList: (1) proving each theorem in the dataset, and (2) predicting the tactic and the arguments of the tactic that were used in the human proof. A tactic can be a previously proven theorem or a list of previously proven theorems.

2.5.31 MetaMathStep

MetaMathStep [148] is a benchmark for automated theorem proving. The dataset evaluates the capabilities of a language model to generate a proof for a given statement. The dataset contains 3 million proof steps for around 38000 theorems, which are split into 36K for training, 1K for validation, and 1K for testing.

2.6 Conclusion

This chapter has provided an introduction to important concepts related to Language Models. It has covered key aspects such as the definition of LMs, their evolution, how they are trained, and the different types of LMs used today. Moreover, we have discussed reasoning tasks used to test the reasoning abilities of LMs, along with a list of datasets. This knowledge will serve as a foundation for readers to understand the following chapters better, which focus on evaluating and improving the reasoning abilities of LMs. Next, we will explore what types of reasoning SLM can effectively perform.

3

Reasoning with SLMs: Deep learning, but shallow reasoning

The first question that we address in this thesis is what different types of reasoning SLMs can effectively perform. We discuss the performance of SLMs on different reasoning tasks, including mathematical reasoning, commonsense reasoning, and logical reasoning. We point out successes and limitations of both empirical and theoretical nature. This chapter is based on the following paper [70].

> ***Chadi Helwe**, Chloé Clavel, Fabian Suchanek. **"Reasoning with Transformer-based models: Deep learning, but shallow reasoning"** (long paper) AKBC 2021*

3.1 Introduction

In this chapter, we investigate some of the most complex natural language tasks: those that involve reasoning. That is, we look at test data sets to evaluate the limitations of LMs, and we investigate to what degree SLMs really "understand" these tasks. While several survey papers have focused on Transformer-based models and their applications [158, 151, 200, 207], the capabilities of Transformer-based models in reasoning tasks have so far not been surveyed. This chapter is organized as follows. In Section 3.2, we describe some common pitfalls that all models need to handle in order to reason on natural language text; we take BERT as a case study. Section 3.3 analyzes the performance of SLMs on different reasoning tasks. In Section 3.4, we describe the theoretical limitations of the Transformer architecture and show, in Sections 3.4.2-3.4.3, that they impact natural language reasoning. We conclude in Section 3.5.

3.2 Common Pitfalls for SLM

We discuss here some common pitfalls that any approach needs to handle in order to reason on natural language. Our discussion focuses on BERT, but the phenomena may affect other SLMs as well.

3.2.1 Negation

The pre-trained BERT model cannot differentiate between positive and negative statements. As an example, take this sentence from the LAnguage Model Analysis (LAMA) dataset [143], where BERT performs well:

Example 3.1

Context: Marcel Oopa died in the city of [MASK].
Expected answer: Paris
Model answer: Paris (-2.3), Lausanne (-3.3), Brussels (-3.3)

When Kassner and Schütze [95] added the word "not", BERT delivered the exact same top-ranked result:

Example 3.2

Context: Marcel Oopa did not die in the city of [MASK].
Expected answer: any city different from Paris
Model answer: Paris (-2.4), Helsinki (-3.5),Warsaw (-3.5)

This phenomenon was also confirmed by Ettinger [49]. Kassner and Schütze [95] show that BERT can be fine-tuned to pay attention to the negation. Thus, it is essential to add examples with negation to the training set. Niven and Kao [131] point out that these examples should be diverse enough to not rely only on the word "not," and Hosseini et al. [84] propose an unlikelihood objective function to train SLMs at a token level to differentiate between positive and negative statements.

3.2.2 Mispriming

The ability to distinguish useful from distracting contexts is an essential building block for any reasoning task. We have already seen an example of mispriming in the introduction. Mispriming can, in principle, affect any task, and thus also reasoning in particular.

Interestingly, mispriming works only when the distracting word is of the same type as the expected answer (companies, in our example). The pre-trained BERT is not

easily misled by primes of other types [131]. Misra et al. [127] also show that the problem of mispriming can be overcome by providing more context. In the following sentence, for example, the mispriming fails:

Example 3.3

Context:	Samsung. The iPhone was produced by [MASK], whose CEO was Steve Jobs
Expected answer:	Apple
Model answer:	Apple

This shows that, although there is some dependency on misprimes, their power decreases when sentences provide more context.

3.2.3 Pattern Heuristics

Fine-tuned BERT models have a tendency to learn simple pattern-based heuristics. For example, BERT can be trained to perform well on textual entailment in the MNLI dataset [197]:

Example 3.4

Premise:	The actor and the professor mentioned the lawyer.
Hypothesis:	The professor mentioned the lawyer.
Expected answer:	Entailment
Model answer:	Entailment

To better understand the performance of BERT, McCoy et al. [121] designed the HANS (Heuristic Analysis for NLI Systems) dataset. HANS makes BERT fail as follows:

Example 3.5

Premise:	The doctors advised the presidents and the tourists.
Hypothesis:	The presidents advised the tourists.
Expected answer:	Non entailment
Model answer:	Entailment

This shows that the model learned the "lexical overlap heuristic", which assumes that a premise entails all hypotheses constructed from words in the premise. This problem can be addressed by adding more HANS-like examples to the training dataset.

3.2.4 Word Order

Different studies [49, 163, 144, 62] have shown that SLMs are unperturbed by grammatically incorrect sentences: If presented with a sentence of randomly shuffled words, they will still reply correctly. This insensitivity to order can also mislead textual entailment. For example, the pre-trained BERT fine-tuned on the MNLI dataset fails to provide the correct answer in the following case [121, 120]:

Example 3.6

Premise:	The doctor visited the lawyer
Hypothesis:	The lawyer visited the doctor
Expected answer:	Non entailment
Model answer:	Entailment

This issue can be solved by augmenting the training set with modified word order instances with their respective labels or by fine-tuning the model on sensitive word ordering tasks such as CoLA [186].

3.3 Types of Reasoning with SLMs

3.3.1 Horn Rule Reasoning

A rather simple way of logical reasoning is to infer a conclusion from a set of premises and rules. SLMs are able to perform such kind of reasoning [33, 176, 18] without any external knowledge, if both the rules and the facts are mentioned explicitly in the text. They can even generate the proofs [159, 174, 61]. Here is an example from the ProofWriter dataset [174]:

Example 3.7

Context:	Fact 1: Erin is young. Fact 2: Erin is not kind. Fact 3: Peter is nice. Rule 1: If someone is young and not kind then they are big.
Question:	Is Erin big?
Expected answer:	*Conclusion:* Erin is big. *Proof:* (Fact 1 & Fact 2) $\rightarrow$ Rule 1 $\rightarrow$ Conclusion

In this task, the best model, a fine-tuned T5-11B, achieves an accuracy above 95% in proof generation and question answering. A Transformer-based model can thus solve the task nearly perfectly.

3.3.2 Commonsense Reasoning

Commonsense reasoning is any reasoning task that requires background knowledge that humans commonly have. For example, the instruction "Can you do a Napoleon for the camera?" requires commonsense reasoning to realize that the word "Napoleon" expresses a specific pose [13]. Several studies have shown that BERT learned a certain amount of commonsense knowledge during pre-training [143, 41, 23, 220, 36]. Consider, for example, the LAMA dataset [143], which asks:

Example 3.8

Context: Ravens can [MASK]
Expected answer: fly
Model answer: fly

The model (the pre-trained BERT-large) is able to recall such commonsense knowledge. This good performance has prompted the research community to develop datasets that specifically probe the commonsense reasoning of Transformer models. Prominent datasets are COSMOS QA [87], CommonsenseQA [175], the Winograd Schema Challenge [102], SWAG [212], ReCoRD [214], CODAH [28], and PIQA [21]. Transformer-based models can indeed achieve a high performance (often $> 75\%$) on these datasets, but only with additional methods. These include data augmentation techniques [206], multi-task learning [115], and fusing knowledge graphs into language models [204]. The following is an example from the CommonsenseQA dataset [175]:

Example 3.9

Question: Bats have many quirks, with the exception of ?
Expected Answer: Laying eggs
Model w/o knowledge graph fusing: Eating bugs
Model w/ knowledge graph fusing: Laying eggs

The above example shows that providing the model with information from a knowledge graph helps the model to correctly answer the question. However, several studies [52, 220, 107, 22, 172] show that when the datasets are specifically changed to target the weaknesses of Transformer-based models (for example, by adversarial instances), the models fail. Here is an example from the COM2SENSE dataset [172], which asks the model to judge whether a given sentence is logically coherent or not:

Example 3.10

Context: Expecting ten fish in the net, Sammy was thrilled to see forty fish swimming in there.
Expected answer: Coherent
Model answer: Coherent

The authors created a counterpart to this question by modifying a few words:

Example 3.11

Context: Expecting ten fish in the net, Sammy was thrilled to see *five* fish swimming in there.
Expected answer: Incoherent
Model answer: Coherent

When the model (UnifiedQA-3B [96], a multi-task trained model) is tricked this way, it fails to predict correctly. This shows that the model can fall prey to relatively simple modifications, and does not really reason.

3.3.3 Event-based Commonsense Reasoning

Some commonsense reasoning tasks are concerned with the usual sequence of events. For example, the TIMEDIAL dataset [150] evaluates temporal reasoning capabilities in dialogs. The TORQUE dataset [130] asks temporal relation questions such as which events have already finished, given a short passage of text. In a similar spirit, the MCTACO dataset [217] asks:

Example 3.12

Context: Mr. Barco has refused US troops or advisers but has accepted US military aid.
Question: What happened after Mr. Barco accepted the military aid?
Choices: (A) the aid was denied, (B) he received the aid, (C) things started to progress

The best model is a fine-tuned BERT model that uses normalization to convert numerical expressions such as "30 months" to "2.5 years". It achieves an F1-score of 69.9% (while human performance has an F1-score of 87.1%). In the same spirit, Zhou et al. [218] developed TRACIE, a temporal reasoning textual entailment dataset that asks

whether one event preceded another one. The authors use distant supervision from Wikipedia, and a symbolic reasoning model called SymTime. This approach predicts the end time of an event by having two Transformer models that predict the start time and the duration of this event and symbolically compare them against the prediction of another start time event. With this, the authors achieve an accuracy of about 71% (with variations for different subtasks). Like the "normal" commonsense tasks, event-based tasks can be solved rather well by Transformer-based models. However, this works mainly when symbolic machinery (such as date normalization and symbolic reasoning) or background knowledge (such as Wikipedia) is added. Human performance, in the high nineties, remains unachieved.

3.3.4 Implicit Reasoning

We now turn to implicit reasoning tasks, where (different from the tasks in Section 3.3.1), the rules and facts are not given explicitly. Many of these tasks can be solved by Transformer-based models. Here is an example from the SNLI dataset [24]:

Example 3.13

Premise:	Three girls take cover under their umbrellas.
Hypothesis:	Nobody has umbrellas.
Expected answer:	Contradiction
Model answer:	Contradiction

In this task, a RoBERTa-large model, trained with a few-shot learning method [184], achieves an accuracy of 93.1%. However, these datasets contain superficial cues that the models can take advantage of [167, 88, 108]. To adequately evaluate the understanding of a model, several more challenging logical reasoning tasks have been designed, which mostly take the form of machine reading comprehension. LogiQA [112], for example, is a multiple choice dataset translated from the National Civil Servants Examination of China:

Example 3.14

Context:	David knows Mr. Zhang's friend Jack, and Jack knows David's friend Ms. Lin. Everyone of them who knows Jack has a master's degree, and everyone of them who knows Ms. Lin is from Shanghai.
Question:	Who is from Shanghai and has a master's degree?
Choices:	(A) David (B) Jack (C) Mr. Zhang (D) Ms. Lin

The best language model is a pre-trained RoBERTa model [113] fine-tuned on the training set and has an accuracy of 35.31% (while the best human performance is 96%) [112]. Several other benchmarks in this vein also show bad performance: ReClor [209], QuAIL [157], ConTRoL [110], StrategyQA [55], AR-LSAT [216], and CLUTRR [173]. This shows that Transformer-based models are currently unable to build a representation of a longer text and draw a logical conclusion from it. This weakness can be remedied to some degree by adding symbolic representations on top of RoBERTa, such as graph-based modules [89, 135], or logical information [185]. Other approaches develop neuro-symbolic methods, which teach reasoning strategies by gradient-based optimisation [125], or combine probabilistic logic programming with neural networks [117]. Integrating logical information into RoBERTa pushes the performance on the easier questions of ReClor to 81.4%. However, the more difficult questions of these datasets incur performances of 50%-60%. The same is true for comparison-based tasks. The RICA dataset [219], for example, asks:

Example 3.15

Context: A prindag is smaller than a flurberg,
so a flurberg is [MASK] likely to contain a prindag.
Expected answer: more

Pre-trained and fine-tuned language models such as GPT-2 [152] and RoBERTa achieve a dismal performance of 50% on unseen inferences. Thus, these models are unable to learn comparisons between (fictitious) objects.

3.3.5 Mathematical Reasoning

Mathematical reasoning is the process of reasoning about different mathematical aspects such as arithmetic operations, numerical comparison, counting, and sorting. The level of complexity can range from solving simple mathematical equations to proving theorems. The following is an example of a math problem that is not linguistically complex, taken from the DeepMind mathematics dataset [165]:

Example 3.16

Context: Calculate -841880142.544 + 411127
Expected answer: -841469015.544

This task can be solved by GPT-3 [76]. Along the same line, Lample and Charton [100] show that a Transformer network can compute function integrals, and solve differential equations. The next more complex problems are math word problems (MWP), which consist of a short text that describes a mathematical problem (such as

a one-unknown math problem) and a question. This task requires a model to extract relevant information from the text to perform mathematical reasoning to solve it. The most prominent MWP datasets are MAWPS [97] and ASDiv-A [123] (both for one-unknown arithmetic problems). However, these datasets can be solved by models even when the order of the words is modified and when the questions are omitted, proving that the models rely on heuristic patterns found in the problem narrative. To remove these artifacts, Patel et al. [138] developed the SVAMP dataset, which applies simple variations to ASDiv-A. The following is an example:

Example 3.17

Context: Jack had 8 pens and Mary had 5 pens. Mary gave 3 pens to Jack.
Question: How many pens does Jack have now?
Expected answer: 8 + 3 = 11

On this dataset, a trained model achieves an accuracy of around 65%. Among the different mathematical operators (+,-,/,*), the model accuracy ranges from 65.3% for divisions to 35.8% for multiplications. Also, the performance drops drastically when the equations have more than two numbers or more than one operator.

An even more complicated dataset is MATH [75], which consists of competition problems in mathematics:

Example 3.18

Context: Tom has a red marble, a green marble, a blue marble, and three identical yellow marbles.
Question: How many different groups of two marbles can Tom choose?
Expected answer: There are two cases here: either Tom chooses two yellow marbles (1 result), or he chooses two marbles of different colors ($\binom{4}{2}$= 6 results). The total number of distinct pairs of marbles Tom can choose is 1 + 6 = 7

Here, the best model, a fine-tuned GPT-2 model, achieves an accuracy of only 6.9%.

Another dataset at the boundary of what is currently feasible is the IsarStep benchmark [104], which is concerned with mathematical proofs:

Example 3.19

Context: $2b^2 = a^2 \Rightarrow [Missing\ Proposition] \Rightarrow \exists\ c \in\ \mathbb{Z}.\ a = 2c$

Expected answer: a is even

The authors developed a hierarchical Transformer model, which outperforms all the other tested baselines with an accuracy of 22.8% for the top-1 prediction, and an accuracy of 35.2% for the top-10 predictions. Other mathematical theorem proving datasets in the same spirit are HOList [11] and MetaMathStep [148]. In conclusion, these tasks show that Transformer-based models cannot be trained to "understand" mathematical word problems and to "generate" mathematical proofs. In contrast to simple mathematical problems (as the example we mentioned above), which are not linguistically complex, such challenging tasks require more than huge Transformer-based models to achieve high performance.

3.3.6 Summary

In all of these reasoning tasks, the SLMs rarely achieve human performance. That is not surprising, given that they are general-purpose tools that feed mainly from training data, and lack any symbolic machinery that is commonly considered essential for such tasks. In fact, it is impressive that the models perform so well at all.

Among the different reasoning tasks, we find that when these SLMs are explicitly given all the information required to perform deductive reasoning, such as facts and rules, the models can easily learn logical reasoning. However, when this information is stated only implicitly in the text or in the supervision, the models struggle. In terms of commonsense reasoning, Transformer-based models have a certain degree of commonsense knowledge learned during pre-training. However, they can be easily disrupted with adversarial commonsense instances. They are also limited in logical reasoning over events and physical commonsense. Appendix A.1 presents the model performances on different reasoning datasets.

We thus see that the strength of SLMs comes from two components: simple patterns in the training data, combined with background knowledge from the pretraining. This combination allows the models to perform well on tasks such as Horn Rule Reasoning (where the model learns a pattern on the training data), simple commonsense reasoning (where the answer was learned from the pretraining), and simple mathematical calculations (where the model learns a pattern during training). However, when these elements are absent, the models struggle. We have seen several commonsense datasets that specifically avoid patterns, or use adversarial patterns. Here, the models fail. In particular, textual understanding remains out of reach for now, if the tasks are sufficiently different from each other to avoid patterns, and if fictional entities are used (for which no background knowledge is available). Mathematical reasoning, too, falls into this category, if the tasks do not follow a pattern.

This does not mean that the tasks would be unsolvable in general: Several studies [89, 135, 185, 206, 115, 204] show that the addition of symbolic knowledge (such as date normalization, quasi-logical reasoning, and graph-based modules) and the use of supplementary techniques such as data augmentation, multi-task learning, and knowledge-base fusion improve the performance. Such tools may thus hold the key to address even the harder reasoning problems. Table 3.1 provides information on the datasets mentioned in this section, including their reasoning types and sizes (further details on each dataset can be found in Chapter 2).

Table 3.1: Dataset with Type of Reasoning and Size

Type of Reasoning	**Dataset**	**Size**
Horn Reasoning	ParaRules	$\approx$ 40,000
Horn Reasoning	ProofWriter	$\approx$ 837,000
Commonsense Reasoning	ProtoQA	9,916
Commonsense Reasoning	COM2SENSE	$\approx$ 4000
Commonsense Reasoning	CODAH	2,801
Commonsense Reasoning	CATs	N/A
Commonsense Reasoning	PIQA	21,000
Event-based Commonsense Reasoning	TIMEDIAL	$\approx$ 1100
Event-based Commonsense Reasoning	TORQUE	$\approx$ 21,000
Event-based Commonsense Reasoning	MCTACO	$\approx$ 13,000
Event-based Commonsense Reasoning	TRACIE	$\approx$ 5,500
Implicit Reasoning	SNLI	$\approx$ 550,000
Implicit Reasoning	RICA	16,000
Implicit Reasoning	LogiQA	8,678
Implicit Reasoning	ReCLOR	6,138
Implicit Reasoning	AR-LSAT	2,046
Implicit Reasoning	QuAIL	$\approx$ 15,000
Implicit Reasoning	StrategyQA	2,780
Implicit Reasoning	ConTROL	8,325
Implicit Reasoning	CLUTRR	5,100
Mathematical Reasoning	SVAMP	1,000
Mathematical Reasoning	MATH	12,500
Mathematical Reasoning	IsarStep	$\approx$ 830,000
Mathematical Reasoning	HoList	29465
Mathematical Reasoning	MetaMathStep	$\approx$ 38000

3.4 Impossible Reasoning Tasks

We have seen that SLMs can be trained and tuned to work on reasoning tasks, and that some tasks require additional machinery. In this section, we turn to tasks that

BERT will never be able to solve without additional machinery, no matter the amount of tuning.

3.4.1 Theoretical Limitations of Transformers

Hahn [66] studied the theoretical limitations of Transformers. The main limitations come from the fact that self-attention does not have the same level of expressiveness as recurrent models such as LSTMs. In particular, Transformers cannot emulate a stack and finite-state automata. Based on this insight, Hahn proved that Transformer-based networks cannot model two languages, known as Even Parity and Dyck-2. Even Parity is the set of bit strings where the number of 1s is even. Dyck-2 is the language of strings that are balanced sequences of round brackets "()" and square brackets "[]". Hahn shows that for any Transformer network, we can find an integer N such that strings of these languages longer than N cannot be recognized. That is: to recognize such strings, the number of heads and layers of the model would have to increase with the input length N. Bhattamishra et al. [20] have verified these results in practice, and showed that when the input length is bounded during training, LSTM can generalize to longer instances, whereas Transformer architectures cannot. According to Bhattamishra et al. [19], Transformers also have limitations in recognizing other formal languages. And yet, they have a very concrete impact on natural language reasoning. To show this, we designed two experiments: the light switch task and the cake task. All our datasets and the code of the experiments can be found at https://github.com/dig-team/FailBERT.

3.4.2 Light Switch Task

Our first task puts the Even Parity language into practice. The input is a word of the language $(\,(a \mid b)\ \texttt{and}\ )^\star (a \mid b)$, with a ="I ate a pizza" and b ="I operated the light switch", i.e., the input is a sentence that describes a sequence of these two activities. Assuming that the light is off in the beginning, the goal is to determine whether the light is on in the end (which corresponds to an odd number of switches). This is a simple reasoning task that a school child can solve.

Example 3.20

Context: I operated the light switch, and I ate a pizza, and I ate a pizza.
Expected answer: ON

We fine-tuned a pre-trained RoBERTa model for 50 iterations on 20k examples, each containing up to 20 a's and b's. On the training and validation datasets, the model achieves an F-score > 0.99. However, when testing on examples that contain more than 20 a's and b's, we obtain on average a random precision of 0.50. This confirms that the

theoretical limitation of the Transformer-based model has practical implications for natural language reasoning.

We also tested the performance of an SLM with a few billion parameters, the Mistral Instruct model [90], in a zero-shot setting. We provided both context and instruction on 100 examples randomly extracted from the training dataset.

Here is the prompt used for this task:

You are an expert in counting.
The initial state of the switch is off, the actions taken are: {context}
Given that operating the switch toggles its state between 'on' and 'off,' determine the final state of the switch. Answer with 'Answer: True' for 'on' or 'Answer: False' for 'off' only.
Output:

The model correctly predicted 44 out of 100 examples. This shows that having more parameters cannot solve the theoretical limitation of the Transformer architecture. In addition, we checked ChatGPT 4 [2], an LLM, using the OpenAI interface, it actually wrote Python code to solve the problem. However, when we disallowed Python, ChatGPT 4 was unable to solve the problem on its own.

3.4.3 Cake Task

Our next task puts the Dyck language into practice. The input to the task is a word of the language $S \to \epsilon|SS|aSa'|bSb'$, where $a =$"I add a peanut layer to my cake", $a' =$"I eat a peanut layer from my cake", $b =$"I add a chocolate layer to my cake", and $b' =$"I eat a chocolate layer from my cake" (with the conjunction "and" in suitable places). The goal is to determine whether this sequence of steps is possible and the cake is gone. Again, this is a simple reasoning task that a child can solve on a sheet of paper (or with suitable baking tools).

Example 3.21

Context: I add a peanut layer and I eat a peanut layer.
Expected answer: Yes

Example 3.22

Context: I add a peanut layer and I eat a chocolate layer.
Expected answer: No

We fine-tuned a pre-trained RoBERTa model on 24k examples, each with up to 20 items, and with nesting depths up to 15, for 50 iterations. Again, the model achieves an F-score > 0.99 on the training and validation sets. However, when testing on examples that contain more than 20 items, and on depths larger than 15, we obtain, as expected, on average a dismal F-score of 0.55.

Similarly to the Light Switch Task, we also tested the performance of the Mistral Instruct model in a zero-shot setting by providing both context and instruction on 100 examples randomly selected from the training dataset.

Here is the prompt used for this task:

You are an expert in counting.
Given the following string: {context}
Determine if the string is a valid Dyck-2 language. Answer with 'Answer: True' for valid or 'Answer: False' for invalid only.
Output:

The model correctly predicted 51 out of 100 examples. Furthermore, we evaluated ChatGPT 4 in the same condition as the previous task, and we observed similar results to those of the Light Switch Task.

This shows again that the theoretical limitations of Transformer-based models lead to very concrete limitations on natural language reasoning.

3.5 Conclusion

In this chapter, we have shown that SLMs can perform a shallow level of reasoning on English textual data, but lack deeper reasoning capabilities. The first stumbling stones are some common pitfalls for SLMs: word order, negation, shallow patterns, and priming problems. The models have to be explicitly trained to deal with these. We then discussed several reasoning tasks, from simple Horn rule reasoning to more complex commonsense, textual understanding, and mathematical tasks. For these tasks, the performance of SLMs is significantly behind that of human performance. However, LLMs are currently able to solve most of the mentioned tasks, but this does not necessarily translate to high reasoning capabilities. For instance, LLMs trained on "A is B" may not be able to generalize to "B is A" [16]. While GPT-4 shows relatively good performance on well-known datasets such as LogiQA and ReClor, its ability to handle newly released and out-of-distribution datasets is significantly lower [111]. A promising direction of research to have better reasoning capabilities involves adding symbolic knowledge to the system, i.e., Neuro-symbolic AI. This approach has been successfully applied to some tasks. Lastly, we have also recalled that Transformer-based models have theoretical limitations in that they cannot model the languages Even Parity and Dyck-2. Even Parity is the set of bit strings where the number of 1s is even. Dyck-2 is the language of strings that are balanced sequences of round

brackets “()” and square brackets “[]”. We have shown on small reasoning tasks that these theoretical limitations, too, can hinder reasoning in natural language; even LLMs cannot solve these tasks without accessing external tools such as a Python interpreter. Further research could explore how different types of positional encodings (such as learned embeddings, sinusoidal embeddings, or CAPE [106]) and different attention mechanisms (such as saturated attention [122]) could help the models overcome even these limitations.

4 LogiTorch: A PyTorch-based library for logical reasoning on natural language

In Chapter 3, we discussed several datasets that were created to assess the reasoning abilities of LMs. These datasets revealed that SLMs do not perform well on many reasoning tasks. We now turn to the question of how SLMs can be evaluated systematically on such reasoning tasks. We introduce LogiTorch, a PyTorch-based library that includes different logical reasoning benchmarks, different implemented SLMs, as well as utility functions such as co-reference resolution. This makes it easy to directly use the preprocessed datasets, to run the models, or to finetune them with different hyperparameters. This chapter is based on the following paper [71]:

> ***Chadi Helwe**, Chloé Clavel, Fabian Suchanek. **"LogiTorch: A PyTorch-based library for logical reasoning on natural language"** (demo paper) EMNLP 2023*

4.1 Introduction

As we discussed previously, LMs perform wells on many NLP tasks. However, in Chapter 3 we showed that SLMs can be distracted easily by trap words, syntactic variations [95], or negation [95, 49, 82, 83, 70]. Hence, the question of whether these models can reason on text is still open [131, 70]. New SLMs are being created incessantly (e.g., LogiGAN [145] and Logiformer [203] in 2022), and new datasets are being created to evaluate these models, including, e.g., LogiQA [112] and ProofWriter [174]. The initiative of open-sourcing toolkits has accelerated the progress in the field of natural language processing, driven by projects such as Transformers [199] from HuggingFace and Stanza [149] from Stanford. However, this progress has not yet arrived in the field of reasoning: researchers still have to find and download different models, parameterize them, find the corresponding datasets, bring them into suitable formats, and finetune the models. The datasets are maintained on different Web pages,

exhibit different formats (JSON vs. full text, numerical vs. textual labels, etc.), and follow different conventions, which makes it cumbersome to apply one model across several sources. The models themselves are implemented in different frameworks, have different input and output formats, require different dependencies, and differ in the way of running them, which makes it burdensome to exchange one model for another. Some models are not even available online, but have to be re-implemented from scratch based on the diagrams in the scientific publications. All of this hinders reproducibility, re-usability, comparability, and, ultimately, scientific progress in the area.

In this chapter, we propose to bring the benefits of open source libraries to the domain of reasoning: we build a Python library, LogiTorch, that includes 14 datasets and 4 implemented models for 3 different reasoning tasks. All models can be called in a unified way, all datasets of one task are available in the same standardized format, and all models can be run with all datasets of the same task. All models have been re-implemented from the research papers that proposed them, and they have been validated by subjecting them to the same experiments as the original papers, with comparable results. More models and benchmarks are in preparation. LogiTorch works on top of PyTorch [137], and uses the Transformers library. It also includes utility functions used for preprocessing, such as coreference resolution and discourse delimitation.

The rest of the chapter is organized as follows. Section 4.2 discusses the design and components of LogiTorch, and describes the datasets, utility functions, and models. Section 4.3 shows the experimental results of our implemented models on different logical reasoning tasks. We conclude in Section 4.4.

4.2 LogiTorch

LogiTorch is our Python library for logical reasoning on natural language text. Figure 4.1 shows the tree structure of our library. It is built on top of PyTorch and consists of 5 parts:

Datasets. We gathered different reasoning datasets that allow users to evaluate the reasoning capabilities of deep learning models on natural language. Once a dataset is called from LogiTorch, it is downloaded, and wrapped into an object that inherits the Dataset class of PyTorch. This means that all datasets are accessible via the same interface. We list the datasets included in LogiTorch in Section 4.2.1.

Data Collators. Different models require different preprocessing steps for the same data and same task: one model may work on numerical vectors, the other on textual input. Hence, we designed, for each pair of a dataset and a model, a data collator that brings the dataset into the format required by the model.

Utilities. Some models require supplementary features in addition to the input text. For example, the DAGN model [89] requires the discourse structure of the input in order to create a logical graph representation of it. For such cases, LogiTorch provides

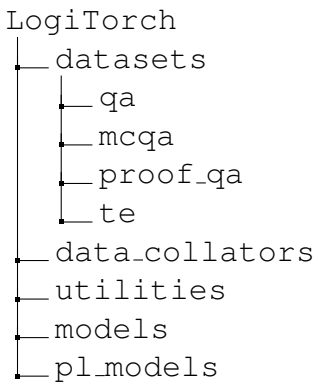


Figure 4.1: Tree structure of LogiTorch

different utility functions, most notably for discourse structure analysis, coreference resolution, and logical expression extraction, which we discuss in Section 4.2.2.

Models. LogiTorch provides several deep learning models that have been designed to perform reasoning tasks such as proof generation and textual entailment. For each model, we either provide an implementation from scratch, or a wrapper over its original implementation. For the Transformer-based models, we use the Transformers library from HuggingFace for the implementation of the models. We describe the models in detail in Section 4.2.3.

PyTorch Lightning Models. For each implemented model, we also provide a PyTorch Lightning version. It includes the model, the optimizer, the training loop, and the validation evaluation. For example, the PRover model [159] has a PyTorch Lightning version called PLPRover. This allows users to play with features such as multi-GPU and fast-low precision training without modifying the training loop.

4.2.1 Datasets

The currently implemented datasets focus on evaluating the reasoning capabilities of LMs. They cover four tasks: Multiple Choice Question Answering (MCQA), Question Answering (QA), Proof Generation, and Textual Entailment (TE).

We have integrated several MCQA datasets that require reasoning capabilities to choose the correct answer. These datasets include AR-LSAT [216], LOGIQA [112], and ReCLoR [209]. For the QA task that focuses on reasoning, we have implemented RuleTaker [33], ParaRule Plus [12], and AbductionRules [208] datasets. Additionally, for the Proof Generation task, we have implemented the ProofWriter [174] dataset. Finally, for the TE task, we have added the SNLI [24], MNLI [196], RTE [38, 67, 56, 57, 15], Negated TE [82], and ConTRoL [110] datasets. In Chapter 2, we have described each dataset included in LogiTorch, and Table 4.1 shows the task and the number of instances of each dataset.

Dataset	Task	Training Instances	Validation Instances	Testing Instances
AR-LSAT	MCQA	1,630	231	230
ReClor	MCQA	4,368	500	1,000
LogiQA	MCQA	7,376	651	651
RuleTaker	QA	587,922	84,030	173,496
ProofWriter	QA/Proof Generation	585,860	85,520	174,180
ParaRules Plus	QA	360,000	64,658	10,798
AbductionRules	QA	80,024	11,432	22,928
ConTRoL	TE	6,719	799	805
SNLI	TE	550,152	10,000	10,000
MNLI	TE	392,702	20,000	20,000
RTE	TE	2,490	277	3,000
Negated SNLI	TE	-	-	1,500
Negated MNLI	TE	-	-	1,500
Negated RTE	TE	-	-	1,500

Table 4.1: Datasets implemented in LogiTorch

4.2.2 Utilities

LogiTorch implements several utility functions that can be used for feature engineering:

Coreference Resolution is the task of finding all mentions in a text that refer to the same entity. For example, in "Zidane is one of the best footballers. He won the World Cup in 1998", the words "Zidane" and "he" refer to the same person. Coreference resolution is used by the Focal Reasoner model [135] to construct a graph of fact triples, where the same mentions are connected with an undirected edge. In LogiTorch, we implemented a wrapper over a finetuned SpanBERT [93] for coreference resolution.

Logical Expression Extraction is the task of extracting a logical representation from a text, in order to infer new logical expressions. For example, the sentence "If you have no keyboarding skills, you will not be able to use a computer" can be split into α = "you have no keyboarding skills" and β = "you are not be able to use a computer". The sentence can then be rewritten as $\alpha \rightarrow \beta$. From this, we can infer by transposition that $\neg\beta \rightarrow \neg\alpha$, which corresponds to "If you are able to use a computer, you have keyboarding skills". The LReasoner model [185] uses this utility function to extend the input with logical expressions. In LogiTorch, we developed a wrapper over the code provided by LReasoner for this purpose.

Discourse Delimitation is the task of splitting a text into elementary discourse units (EDU). It is used for the rhetorical structure theory (RST), in which it is a tree representation of a text where the leaves are EDUs, and the edges are rhetorical relations. For example, "A signal in a pure analog system can be infinitely detailed, while digital systems cannot produce signals that are more precise than their digital unit" is split into two EDUs: "A signal in a pure analog system can be infinitely detailed", and

"digital systems cannot produce signals that are more precise than their digital unit". The DAGN model [89] requires EDUs to construct a graph of discourse units.

4.2.3 Models

LogiTorch currently implements four SLMs:

RuleTaker (QA task) [33] is a RoBERTa-Large model [113] that has been finetuned first on the RACE dataset [98], and then finetuned again for rule-based reasoning. The model takes as input facts and rules and a boolean question. The output is either True or False. A description of the RoBERTa model can be found in Chapter 2.

ProofWriter (QA and proof generation) [174] is a T5 model [154] finetuned to perform rule-based reasoning. It takes as input facts and rules and a question. The output is either True, False, or Unknown (if the trained dataset considers the open-world assumption). A description of the T5 model can be found in Chapter 2.

PRover (QA and proof generation) [159] is built on RoBERTa with three modules: the QA module, Node module, and Edge module. The QA module is responsible for answering a question as either True or False. The Node and Edge modules are responsible for generating proofs. The Node module predicts the relevant rules and facts used to generate the answer, and the Edge module predicts the link between two relevant facts and between a relevant fact and a relevant rule.

BERTNOT (TE task) [84] is a BERT model that is pretrained using the unlikelihood loss and knowledge distillation functions for the MLM task to model negation. Then it is finetuned on textual entailment tasks. This model is more robust on examples containing negations, and performs better on the negated NLI dataset than the original BERT.

Future releases will include newer models such as LReasoner [89], Focal Reasoner [135], AdaLoGN [105], Logiformer [203], and LogiGAN [145].

4.2.4 Library Usage

Listing 1 shows a detailed example of how a model can be trained on a rule-based reasoning dataset for QA. The RuleTaker model is trained on its corresponding dataset. In Lines 9-10, we initialize the training and validation datasets with the RuleTakerDataset. We specify which sub-dataset and which split we want to use. In Line 12, we initialize the RuleTaker data collator for preprocessing the datasets. We then use the Dataloader to pre-load the datasets and use them as batches. In Line 17, we initialize the PyTorch Lightning version of RuleTaker and specify the learning rate, and the weight decay. PyTorch Lightning provides the ModelCheckpoint, which allows monitoring the valida-

tion loss and saving the best model. In Line 26, we use the PyTorch Lightning's Trainer to automate the training loop. It takes several parameters, including the accelerator, which allows training on different devices such as CPUs, GPUs, and TPUs. Finally, we train the model with the fit function. Future releases will also provide pre-configured pipelines to train models.

Listing 1 Training the RuleTaker Model

```
1  import pytorch_lightning as pl
2  from pytorch_lightning.callbacks import ModelCheckpoint
3  from torch.utils.data.dataloader import DataLoader
4
5  from logitorch.data_collators.ruletaker_collator import RuleTakerCollator
6  from logitorch.datasets.qa.ruletaker_dataset import RuleTakerDataset
7  from logitorch.pl_models.ruletaker import PLRuleTaker
8
9  train_dataset = RuleTakerDataset("depth-5", "train")
10 val_dataset = RuleTakerDataset("depth-5", "val")
11
12 ruletaker_collate_fn = RuleTakerCollator()
13
14 train_dataloader = DataLoader(train_dataset, batch_size=32, collate_fn=ruletaker_collate_fn)
15 val_dataloader = DataLoader(val_dataset, batch_size=32, collate_fn=ruletaker_collate_fn)
16
17 model = PLRuleTaker(learning_rate=1e-5, weight_decay=0.1)
18
19 checkpoint_callback = ModelCheckpoint(
20     save_top_k=1,
21     monitor="val_loss",
22     mode="min",
23     dirpath="models/",
24     filename="best_ruletaker.ckpt",
25 )
26 trainer = pl.Trainer(callbacks=[checkpoint_callback], accelerator="gpu", gpus=1)
27 trainer.fit(model, train_dataloader, val_dataloader)
```

Listing 2 shows the code for testing the best-saved model of Listing 1. In Line 3, we load the best model. In Line 8, we use the predict function, which takes as input a context and a question, and predicts either 0 (for False) or 1 (for True).

Listing 2 Predicting with the RuleTaker Model

```
1  from logitorch.pl_models.ruletaker import PLRuleTaker
2
3  model = PLRuleTaker.load_from_checkpoint("models/best_ruletaker.ckpt")
4
5  context = "Bob is smart. If someone is smart then he is kind."
6  question = "Bob is kind."
7
8  model.predict(context, question)
```

4.3 Evaluation

We compared the performance of each model in LogiTorch to the performance of the model in the original paper on the same datasets: we trained the Ruletaker model on the training set of RuleTaker with language reasoning paths up to depth 5 and tested it

on its testing set; we trained the PRover and ProofWriter models on the training set of ProofWriter with language reasoning paths up to depth 5 and tested them on the corresponding testing set; and we trained the BERTNOT model (a pretrained BERT Base Cased model) on the MLM task, with the negated Wikipedia corpus provided by Hosseini et al. [84] (included in LogiTorch), finetuned the model on each TE dataset (MNLI, SNLI, and RTE) and tested it on its negated counterparts [82]. All models use the same settings as in the original papers.

Table 4.2 shows the results of the three different models on the QA task at different reasoning depths. Our model implementations achieve near-perfect accuracies, which are comparable to the performance in the original papers. Table 4.3 shows the performance on the TE task on each TE training dataset (SNLI, MNLI, and RTE). Again, our model achieves nearly the same results as reported in the original paper [84] on the MNLI and SNLI datasets. We are getting lower results on the RTE dataset. We assume that this is because the finetuned model has a high variance due to the small size of the training set of RTE.

Depth	*RuleTaker*[1]		*PRover*[2]		*ProofWriter*[2]	
	LogiTorch	*Original*	*LogiTorch*	*Original*	*LogiTorch*	*Original*[3]
0	99.9	100	100	100	99.9	100
1	98.6	98.4	99.7	99.0	98.0	99.1
2	99.1	98.4	99.5	98.8	96.7	98.6
3	99.2	98.9	99.7	99.1	97.2	98.5
4	99.7	99.2	99.7	98.8	98.1	98.7
5	99.3	99.8	99.5	99.3	99.1	99.3
All	99.3	99.2	99.7	99.3	98.4	99.2

Table 4.2: Accuracies of different models for the QA task at different reasoning depths. [1] Depth-5 of the testing set of RuleTaker dataset. [2] Depth-5 of the testing set of ProofWriter dataset. [3] The original implementation uses a (more powerful) T5-11B model.

Dataset		*LogiTorch's BERTNOT*	*Original BERTNOT*
SNLI	Val	90.4	89.00
	Neg	47.8	45.96
MNLI	Val	83.2	84.31
	Neg	64.0	60.89
RTE	Val	65.6	69.68
	Neg	57.7	74.47

Table 4.3: Results of our BERTNOT implementation on different textual-entailment datasets.

4.4 Conclusion

We have introduced LogiTorch, a Python library for reasoning on natural language. It is built on top of PyTorch in combination with the Transformers and PyTorch Lightning libraries. LogiTorch includes an extensive list of textual logical reasoning datasets and utility functions, and different implemented SLMs. The library allows researchers and developers to easily use reasoning datasets and train reasoning models with just a few lines of code. The library is available on GitHub and is under active development.

For future work, we will add new datasets, and implement models such as DAGN, Focal Reasoner, and LogiGAN with their utility functions for feature engineering. Additionally, we aim to adapt LogiTorch to be compatible with different LLMs like GPT-4. Finally, we want to invite researchers and developers to contribute to LogiTorch. We believe that such a library will lower the hurdles to research in the area, foster re-usability, encourage comparative evaluation, strengthen reproducibility, and advance the culture of open software and data.

5

TINA: Textual Inference with Negation Augmentation

As mentioned, LMs achieve state-of-the-art results on several natural language processing tasks. One of these is *textual entailment*, i.e., the task of determining whether a premise logically entails a hypothesis. However, SLMs perform poorly on this task when the examples contain negations (as we discussed in Chapter 3). In this chapter, we propose a new definition of textual entailment that captures also negation. This allows us to develop TINA (Textual Inference with Negation Augmentation), a principled technique for negated data augmentation that can be combined with the unlikelihood loss function. Our experiments with different SLMs show that our method can significantly improve the performance of the models on textual entailment datasets with negation – without sacrificing performance on datasets without negation. This chapter is based on the following paper [72].

> ***Chadi Helwe**, Simon Coumes, Chloé Clavel, Fabian Suchanek.* ***"TINA: Textual Inference with Negation Augmentation"** (long paper) Findings of EMNLP 2023*

5.1 Introduction

In Chapter 2, we already provide a brief definition of Textual entailment (TE, also called Natural Language Inference). As a reminder, TE is the task of recognizing whether one natural language sentence (the *premise*) semantically entails another one (the *hypothesis*). For example, the premise "I live in Paris" entails the hypothesis "I live in France". TE is at the heart of natural language understanding, as it is closely related to question answering and natural language reasoning [38, 147]. Nowadays, the state-of-the-art performance in TE is achieved by Transformer-based models such as BERT [44].

However, SLMs can get derailed easily by trap words or syntactic variations (as

shown in Chapter 3). In particular, such models have difficulties with negation in textual entailment [82, 84]. Here is an example from Hossain et al. [82]'s dataset:

Example 5.1

Premise:	Green cards are **not** becoming more difficult to obtain.
Hypothesis:	Green card is now difficult to receive.
BERT Prediction:	Entailment
Label:	Not Entailment

In this chapter, we provide a principled analysis of negation in textual entailment. In particular, we propose a probabilistic definition of entailment that can capture also negation. This allows us to develop TINA (Textual Inference with Negation Augmentation), an approach to automatically augment TE training datasets with negated instances. TINA uses logical deduction to generate new negated training examples from existing ones. For example, we can generate that "I don't live in France" entails "I don't live in Paris". We can then show that models finetuned on our augmented datasets are more resilient to negation, especially when combined with the unlikelihood loss. At the same time, the finetuned models perform just as well on datasets without negation. The contributions of this chapter are as follows:

- a novel probabilistic definition of entailment that also considers negation;
- provably correct rules to derive new entailment relationships;
- a method to automatically augment TE datasets using these derivations;
- experiments showing that models that are finetuned on the augmented datasets are more resilient to negation in TE.

The rest of the chapter is organized as follows. In Section 5.2, we review the related work. Section 5.3 describes TINA, our approach to defining textual entailment, and to making Transformer-based models robust to negation in textual entailment. In Section 5.4, we evaluate our approach on different datasets. We conclude in Section 5.5. All data and code is available on GitHub[1].

5.2 Related Work

5.2.1 Negation in Language Models

In Chapter 3, we have discussed how SLMs often struggle with understanding negation, as several studies have shown [49, 70, 95, 131]. However, one interesting attempt to improve the robustness of language models to negation is BERTNOT [84]. BERTNOT is a BERT-based model that uses an unlikelihood objective function during training for the task of language modeling. This model is trained at a token level, which helps the model learn to differentiate between affirmative and negative sentences.

[1] https://github.com/ChadiHelwe/TINA

5.2.2 Data Augmentation

Data augmentation is a technique to automatically create new instances in order to increase the size of a training dataset. It can mitigate problems of low-resource languages, class imbalance, and bias in datasets. Data augmentation techniques can be categorized into rule-based approaches, model-based approaches, and example interpolation [51]. We are interested here in the rule-based category, which uses predefined rules to generate new instances [69, 168, 136, 188, 201, 161, 185]. Our approach is inspired by the work of Wang et al. [185], which uses logical rules for data augmentation. We go further by logically deriving new rules for data augmentation, and by combining the data augmentation with the unlikelihood loss for finetuning Transformer-based models.

5.2.3 Textual Entailment Datasets

In Chapter 2, we already described the different textual entailment datasets. These datasets are SNLI (Stanford Natural Language Inference) [24], MNLI (Multi-Genre Natural Language Inference) [196], and Pascal RTE [38, 67, 56, 57, 15]. The current state-of-the-art models achieve an accuracy of approximately 92-95% on these datasets. The top-performing models are EFL (a RoBERTa-based model) [184] for SNLI, T5-11B [154] for MNLI, and Google's Pathways Language Model (PaLM) [31] for RTE.

5.2.4 Negated Textual Entailment

The good performance of language models on textual entailment datasets raises the question of whether this good performance persists in the presence of negation [82, 83]. Negation is generally underrepresented in TE datasets (Hossain et al. [82]), with 7.16% of SNLI's sentences containing a negation, 22.63% in MNLI, and 1.19% in RTE. Therefore, Hossain et al. [82] created new benchmarks by taking instances from SNLI, MNLI, and RTE and introducing a negation. They showed that language models perform poorly on these datasets. Hosseini et al. [84] introduced the previously mentioned BERTNOT model to improve performance. In our work, we will show how that performance can be improved even further by using a principled way to augment the training datasets.

5.3 Our Approach: TINA

TINA (Textual Inference with Negation Augmentation) is our proposed approach to build a language model that is robust to negation in textual entailment tasks. Our main idea is to finetune Transformer-based models on a textual entailment dataset that has been augmented with negated instances. For this purpose, let us first revisit the definition of entailment.

5.3.1 Defining Entailment

We say that a text fragment A *entails* a text fragment B (written $A \triangleright B$) if, typically, a human reading A would infer that B is most likely true [38]. Here, A is called the *premise* and B is called the *hypothesis*. For our purposes, we need a more formal definition of entailment. i.e. a definition in mathematical terms that matches the intuitive definition.

Entailment cannot be modeled as a material implication $A \Rightarrow B$ for two reasons: First, a material implication $A \Rightarrow B$ is true if B is true. Thus, "It rains" would entail "Paris is in France" – which is not the usual understanding of entailment. Propositional logic knows no satisfying way to avoid this. We could write $A \triangleright B := (A \Rightarrow B) \wedge (\neg A \Rightarrow \neg B)$; but that is just equivalent to $A \Leftrightarrow B$, which is not what entailment means. The second problem with defining entailment as a logical implication is that it does not allow for exceptions. For example, "I obtained a university diploma" entails "I have a university diploma", even if diplomas can be withdrawn in rare cases of fraud. Propositional logic has no means to say that an implication holds "usually" or "in the majority of cases".

Therefore, previous work [59] has proposed a probabilistic definition of entailment. In what follows, we assume a probabilistic universe Ω and two events (the *premise* A and the *hypothesis* B). Glickman et al. [59] then defines

Definition 5.1: Entailment by Glickman et al. [59]

$$A \triangleright_G B := P(B|A) > P(B)$$

This definition says that A entails B if A increases the probability of B. Unfortunately, this definition has several problems: First, it is symmetric. We show in Proposition 5.3.1 that $(A \triangleright_G B) \Leftrightarrow (B \triangleright_G A)$. For example, "I live in Paris" $\triangleright_G$ "I live in France", because the probability of living in France increases to 100% once we know the person lives in Paris. However, knowing that someone lives in France also increases the probability that this person lives in Paris (from one in several million cities in the world to one in several thousand cities in France). Therefore "I live in France" $\triangleright_G$ "I live in Paris" – which is not our common understanding of entailment.

Proposition 5.3.1. *For all events A and B, if $A \triangleright_G B$ then $B \triangleright_G A$ (with $\triangleright_G$ defined in Definition 5.3.1).*

Proof: Let A and B be two events with $A \triangleright_G B$. We have:

$$\begin{aligned} & P(B|A) = \frac{P(A \cap B)}{P(A)} > P(B) \\ \Rightarrow\ & P(A|B) = \frac{P(A \cap B)}{P(B)} > P(A) \\ \Rightarrow\ & B \triangleright_G A \end{aligned}$$

□

The second problem with Definition 5.3.1 is that $A \triangleright_G B$ even if A increases the probability of B only marginally. For example "I play in the lottery" $\triangleright_G$ "I win the lottery". This is because the probability of winning the lottery increases by playing in the lottery. Again, this is not our usual understanding of entailment.

Therefore, we propose to add the condition $P(B|A) > \theta$, where θ is a threshold for the acceptance of an entailment (say, 90%). Thus, our definition becomes $A \triangleright_\theta B := P(B|A) > P(B) \wedge P(B|A) > \theta$. This also makes the definition asymmetric, thus solving both the first problem and the second problem.

However, the definition is still vulnerable to a third problem: It may get carried away by hypotheses B with a high baseline probability. For example, most people survive the yearly Flu season. Washing your hands further decreases the risk of attracting the Flu (and thus increases the probability of survival). Hence "Alice washes her hands this Monday" $\triangleright_\theta$ "Alice survives this year's Flu season". This is because (1) washing hands indeed increases the probability of survival, and (2) the probability of surviving is already larger than θ (for $\theta = 90\%$). However, we would not say that the entailment holds. To guard against such cases, we propose to add another condition, $P(\neg A|\neg B) > \theta$. Our definition is thus:

Definition 5.2: Entailment

$$A \triangleright B := P(B|A) > P(B) \wedge P(B|A) > \theta \wedge P(\neg A|\neg B) > \theta$$

with a given constant parameter $\theta \in [0; 1]$.

We write $A \not\triangleright B$ to say that A does not entail B. We can then use our notion of entailment to define contradiction and neutrality.

Definition 5.3: Contradiction

$$A \blacktriangleright B := A \triangleright \neg B$$

Definition 5.4: Neutrality

$$A \multimap B := A \not\triangleright B \wedge A \not\blacktriangleright B$$

5.3.2 Deriving New Instances

We can now use our definition of entailment to derive new premise-hypothesis pairs from a given pair. In what follows, let us denote the negation of a sentence A by $\neg A$.

Formally, $\neg A := \Omega - A$. For example, the negation of "I live in Paris" is "I don't live in Paris". The negation of natural language sentences is a research topic on its own. For example, the negation of Noam Chomsky's famous nonsensical sentence "Colorless green ideas sleep furiously" is not "Colorless green ideas do not sleep furiously", as both are nonsensical. We refer the reader to Horn [81], Löbner [114], Penka [142] and Homer et al. [79] for a discussion. Here, we assume that both the premise and the hypothesis of a textual entailment instance are simple sentences that can be negated.

Now assume that we have $A \rhd B$. Then Definition 5.3.1 allows us to formally derive $\neg B \rhd \neg A$ (as shown in Proposition 5.3.2). For example, "I live in Paris" $\rhd$ "I live in France", and hence "I don't live in France" $\rhd$ "I don't live in Paris". This type of reasoning is known as *Modus Tollens*. Table 5.2 shows other ways to derive new instances from a given instance, together with references to their proofs. A particularly interesting result is that $\blacktriangleright$ is symmetric, i.e., $(A \blacktriangleright B) \Leftrightarrow (B \blacktriangleright A)$.

Proposition 5.3.2 (Modus Tollens)**.** *For all events A and B, if $A \rhd B$ then $\neg B \rhd \neg A$.*

Proof: By definition of $A \rhd B$ we have all of the following:
- $P(B|A) > P(B)$
- $P(B|A) > \theta$
- $P(\neg A|\neg B) > \theta$

We need to prove all of the following:
- $P(\neg A|\neg B) > P(\neg A)$
- $P(\neg A|\neg B) > \theta$
- $P(\neg\neg B|\neg\neg A) > \theta$

The last condition is equivalent to $P(B|A) > \theta$. Hence we need to prove only $P(\neg A|\neg B) > P(\neg A)$.

To simplify the proof we introduce: $a = P(A \cap \neg B)$, $b = P(\neg A \cap B)$, $c = P(A \cap B)$, $d = P(\neg A \cap \neg B)$ (summarized in Table 5.1). Then we have:

$$
\begin{aligned}
& P(B|A) = \frac{P(A \cap B)}{P(A)} > P(B) \\
\Rightarrow\ & \frac{P(A \cap B)}{P(A \cap B) + P(A \cap \neg B)} > P(A \cap B) + P(\neg A \cap B) \\
\Rightarrow\ & \frac{c}{a + c} > b + c \\
\Rightarrow\ & \frac{1 - a - b - d}{1 - b - d} > 1 - a - d \\
\Rightarrow\ & \frac{d}{a + d} > b + d \\
\Rightarrow\ & P(\neg A|\neg B) > P(\neg A)
\end{aligned}
$$

□

	B	$\neg B$
A	c	a
$\neg A$	b	d

Table 5.1: Shorthand notations. For example, b is equal to $P(\neg A \cap B)$.

Some of the derivations in Table 5.2 give us a label that an instance cannot have, rather than telling us which label it must have. We call such a label a *rejected label*. For example, an instance with the label $A \rhd B$ (*entailment*) generates a new instance with the rejected label $\neg A \not\rhd B$ (*non-entailment*, $\neg A$ does not entail B). This means that the true label cannot be an *entailment*, and that it has to be either *neutral* or a *contradiction*.

We are interested in entailments that logically follow from $A \rhd B$, from $A \blacktriangleright B$, from $A \multimap B$ and from $A \not\rhd B$, as these are the labels that common textual entailment datasets use: MNLI and SNLI use the first three labels, while RTE uses the first and last label. While Table 5.2 shows all derivations that must hold, Table 5.3 shows all other hypothetical derivations, and proves them wrong. We can thus use Table 5.2 to derive, for a given labeled instance, new labeled instances. Most of these contain negation.

Original	Derivation	Proof	Example
$A \rhd B$	$A \rhd B$	-	I live in Paris $\rhd$ I live in France
	$A \blacktriangleright \neg B$	Per definition of $\blacktriangleright$	I live in Paris $\blacktriangleright$ I don't live in France
	$\neg B \rhd \neg A$	Proposition 5.3.2 (Modus Tollens)	I don't live in France $\rhd$ I don't live in Paris
	$\neg A \not\rhd B$	Proposition 5.3.3	I don't live in Paris $\not\rhd$ I don't live in France
	$\neg B \blacktriangleright A$	Per definition of $\blacktriangleright$ with Modus Tollens	I don't live in France $\blacktriangleright$ I live in Paris
	$A \not\rhd \neg B$	Proposition 5.3.4	I live in Paris $\not\rhd$ I don't live in France
	$B \not\rhd \neg A$	Proposition 5.3.5	I live in France $\not\rhd$ I don't live in Paris
	$\neg B \not\rhd A$	Proposition 5.3.6	I don't live in France $\not\rhd$ I live in Paris
$A \blacktriangleright B$	$A \blacktriangleright B$	-	I live in Paris $\blacktriangleright$ I live in Italy
	$A \rhd \neg B$	Per definition of $\blacktriangleright$	I live in Paris $\rhd$ I don't live in Italy
	$\neg A \not\blacktriangleright B$	Proposition 5.3.7	I don't live in Paris $\not\blacktriangleright$ I live in Italy
	$B \rhd \neg A$	Equivalent to Proposition 5.3.2 by definition	I live in Italy $\rhd$ I don't live in Paris
	$B \blacktriangleright A$	Per definition of $\blacktriangleright$	I live in Italy $\blacktriangleright$ I live in Paris
	$B \not\blacktriangleright \neg A$	Reduces to $A \rhd B' \Rightarrow \neg B' \not\rhd A$ with $B' = \neg B$	I live in Italy $\not\blacktriangleright$ I don't live in Paris
	$\neg B \not\blacktriangleright A$	Apply Proposition 5.3.2 then 5.3.3	I don't live in Italy $\not\blacktriangleright$ I live in Paris
	$A \not\rhd B$	Reduces to $A \rhd B' \Rightarrow A \not\rhd \neg B'$ with $B' = \neg B$	I live in Paris $\not\rhd$ I live in Italy
$A \multimap B$	$A \multimap B$	-	I live in France $\multimap$ I live in Paris
	$A \multimap \neg B$	Proposition 5.3.8	I live in France $\multimap$ I don't live in Paris
$A \not\rhd B$	$A \not\rhd B$	-	I live in France $\not\rhd$ I live in Paris
	$\neg B \not\rhd \neg A$	Proposition 5.3.9	I don't live in Paris $\not\rhd$ I don't live in France

Table 5.2: Rules for deriving textual entailment instances.

Original	Derivation	Counterexample, reduction, or proof	Illustrative counterexample
$A \rhd B$	$A \not\rhd B$	Trivial	I live in Paris $\rhd$ I live in France
	$A \rhd \neg B$	$a = 0, b = 0, c = 0.125, d = 0.875, \theta = 0$	I live in Paris $\not\rhd$ I don't live in France
	$\neg A \rhd B$	$a = 0, b = 0, c = 0.125, d = 0.875, \theta = 0$	I don't live in Paris $\not\rhd$ I live in France
	$\neg A \rhd \neg B$	$a = 0.02, b = 0.72, c = 0.18, d = 0.08, \theta = 0$	I don't live in Paris $\not\rhd$ I don't live in France
	$B \rhd A$	$a = 0, b = 0.01, c = 0.01, d = 0.98, \theta = 0.8$	I live in France $\not\rhd$ I live in Paris
	$B \rhd \neg A$	Contradicts propositions 5.3.3 and 5.3.2 (Modus Tollens)	I live in France $\not\rhd$ I don't live in Paris
	$\neg B \rhd A$	$a = 0, b = 0, c = 0.01, d = 0.99, \theta = 0$	I don't live in France $\not\rhd$ I live in Paris
	$\neg A \not\rhd \neg B$	$a = 0, b = 0, c = 0.01, d = 0.99, \theta = 0$	I don't live in France $\rhd$ I don't live in France
	$B \not\rhd A$	$a = 0, b = 0, c = 0.01, d = 0.99, \theta = 0$	I live in France $\rhd$ I live in France
$A \blacktriangleright B$	$A \not\blacktriangleright B$	Trivial	I live in Paris $\blacktriangleright$ I live in Italy
	$A \blacktriangleright \neg B$	Reduces to $A \rhd B' \Rightarrow A \rhd \neg B'$ with $B' = \neg B$	I live in Paris $\not\blacktriangleright$ I don't live in Italy
	$\neg A \blacktriangleright B$	Reduces to $A \rhd B' \Rightarrow \neg A \rhd B'$ with $B' = \neg B$	I don't live in Paris $\not\blacktriangleright$ I live in Italy
	$\neg A \blacktriangleright \neg B$	Reduces to $A \rhd B' \Rightarrow \neg A \rhd \neg B'$ with $B' = \neg B$	I don't live in Paris $\not\blacktriangleright$ I don't live in Italy
	$B \blacktriangleright \neg A$	Reduces to $A \rhd B' \Rightarrow \neg B' \rhd A$ with $B' = \neg B$	I live in Italy $\not\blacktriangleright$ I don't live in Paris
	$\neg B \blacktriangleright A$	Reduces to $A \rhd B' \Rightarrow B' \rhd \neg A$ with $B' = \neg B$	I don't live in Italy $\not\blacktriangleright$ I live in Paris
	$\neg B \blacktriangleright \neg A$	Reduces to $A \rhd B' \Rightarrow B' \rhd A$ with $B' = \neg B$	I don't live in Italy $\not\blacktriangleright$ I don't live in Paris
	$\neg A \not\blacktriangleright \neg B$	Reduces to $A \rhd B' \Rightarrow \neg A \not\rhd \neg B'$ with $B' = \neg B$	I don't live in Paris $\blacktriangleright$ I live in Paris
	$B \not\blacktriangleright A$	Contradicts Proposition 5.3.2 (Modus Tollens)	I live in Italy $\blacktriangleright$ I live In Paris
	$\neg B \not\blacktriangleright \neg A$	Reduces to $A \rhd B' \Rightarrow B' \not\rhd A$ with $B' = \neg B$	I don't live in Paris $\blacktriangleright$ I live in Paris
$A \multimap B$	$\neg A \multimap B$	$a = 0.02, b = 0.69, c = 0.06, d = 0.23, \theta = 0$	I don't live in France $\not\multimap$ I live In Paris
	$\neg A \multimap \neg B$	$a = 0.02, b = 069, c = 0.06, d = 0.23, \theta = 0$	I don't live in France $\not\multimap$ I don't live in Paris
	$B \multimap A$	$a = 0.02, b = 0.72, c = 0.08, d = 0.18, \theta = 0$	I live in Paris $\not\multimap$ I live in France
	$B \multimap \neg A$	$a = 0.02, b = 0.72, c = 0.08, d = 0.18, \theta = 0$	I live in Paris $\not\multimap$ I don't live in France
	$\neg B \multimap A$	$a = 0.02, b = 0.69, c = 0.06, d = 0.23, \theta = 0$	I win the lottery $\not\multimap$ I play the lottery
	$\neg B \multimap \neg A$	$a = 0.02, b = 0.69, c = 0.06, d = 0.23, \theta = 0$	I don't live in France $\not\multimap$ I live in Paris
$A \not\rhd B$	$A \not\rhd \neg B$	$a = 0.01, b = 0.01, c = 0, d = 0.98, \theta = 0$	I live in Paris $\rhd$ I don't live in Italy
	$\neg A \not\rhd B$	$a = 0.01, b = 0.01, c = 0, d = 0.98, \theta = 0$	I don't live in France $\rhd$ I don't live in Paris
	$\neg A \not\rhd \neg B$	$a = 0.02, b = 0.69, c = 0.06, d = 0.23, \theta = 0$	I live in France $\rhd$ I don't live in Paris
	$B \not\rhd A$	$a = 0.01, b = 0, c = 0.01, d = 0.98, \theta = 0.8$	I live in Paris $\rhd$ I live in France
	$B \not\rhd \neg A$	$a = 0.01, b = 0.01, c = 0, d = 0.98, \theta = 0$	I live in Paris $\rhd$ I don't live in France
	$\neg B \not\rhd A$	$a = 0.01, b = 0.01, c = 0, d = 0.98, \theta = 0$	I don't live in France $\rhd$ I don't live in Paris

Table 5.3: False derivations are refuted in various ways like counterexamples, reduction to another derivation or contradiction with true derivations.

5.3.3 Proofs of the Derived Rules

Find below the propositions and their corresponding proofs for the different derived rules of Table 5.2.

Proposition 5.3.3. *For all events A and B, if $A \rhd B$ then $\neg A \not\rhd B$.*

Proof: Assume that there exist A and B such that $A \rhd B$ and $\neg A \rhd B$. Then $P(B|A) > P(B)$ and $P(B|\neg A) > P(B)$. Hence, we have:

$$\begin{aligned} & P(B) = P(A) \times P(B|A) + P(\neg A) \times P(B|\neg A) \\ & \Rightarrow P(B) > P(A) \times P(B) + P(\neg A) \times P(B) \\ & \Rightarrow P(B) > P(B) \end{aligned}$$

This is a contradiction, which proves the claim. □

Proposition 5.3.4. *For all events A and B, if $A \rhd B$ then $A \not\rhd \neg B$.*

Proof: Assume $A \rhd B$ and $A \rhd \neg B$. We have $P(B|A) > P(B)$ and $P(B|\neg A) > P(B)$. Hence $P(B) > P(B)$. Contradiction. □

Proposition 5.3.5. *For all events A and B, if $A \rhd B$ then $B \not\rhd \neg A$.*

Proof: If $B \rhd \neg A$ then by Modus Tollens (Proposition 5.3.2), $A \rhd \neg B$. By Proposition 5.3.4 we have $A \not\rhd \neg B$. Contradiction. □

Proposition 5.3.6. *For all events A and B, if $A \rhd B$ then $\neg B \not\rhd A$.*

Proof: If $\neg B \rhd A$ then by Modus Tollens (Proposition 5.3.2), $\neg A \rhd B$. By Proposition 5.3.3 we have $\neg A \not\rhd B$. Contradiction. □

Proposition 5.3.7. *For all events A and B, if $A \blacktriangleright B$ then $\neg A \not\blacktriangleright B$.*

Proof: By definition, our proposition is equivalent to $(A \rhd \neg B) \Rightarrow (\neg A \not\rhd \neg B)$. This is true according to Proposition 5.3.3. □

Proposition 5.3.8. *For all events A and B, if $A \multimap B$ then $A \multimap \neg B$.*

Proof:

$$\begin{aligned} A \multimap B &\equiv (A \not\rhd B \text{ and } A \not\rhd \neg B) \\ &\equiv A \not\rhd \neg B \text{ and } A \not\rhd \neg\neg B \\ &\equiv A \multimap \neg B \end{aligned}$$

□

Proposition 5.3.9. *For all events A and B, if $A \not\rhd B$ then $\neg B \not\rhd \neg A$.*

Proof: Assume A and B such that $A \not\rhd B$ and $\neg B \rhd \neg A$. Then by Modus Tollens (Proposition 5.3.2), $\neg\neg A \rhd \neg\neg B$, which we can restate as $A \rhd B$. Contradiction. □

5.3.4 Unlikelihood Loss

The previous step has given us a way to derive new labeled instances – with either rejected or accepted labels. For the rejected labels, we want to penalize the likelihood of a language model predicting the rejected label. For this purpose, we use the *Unlikelihood Loss*. This loss has been used in many tasks, including in language modeling [84, 132] and text generation [190]. In our case, the loss is defined as:

$$\mathcal{L} = -\frac{1}{N}\sum_{n=1}^{N} v_n log(p_{n,y_n}) + (1 - v_n) log(1 - p_{n,y_n})$$

Here, n runs over all N instances of the dataset. For each instance n and label y, $p_{n,y}$ is the score that the model assigns to the label y for the instance n. To each n we associate a ground truth label y_n, and we know whether this label is accepted or rejected. To distinguish these two cases, v_n is an indicator that takes the value 1 if the label is accepted, and the value 0 if the label is rejected. Our loss is thus the sum of the cross-entropy loss of the accepted labels and the unlikelihood loss of the rejected labels.

5.3.5 Dataset Augmentation

To augment a textual entailment dataset with negated instances, we consider all instances one by one. We first check if the instance consists of a grammatically correct single-sentence premise and single-sentence hypothesis. We use DistillBERT [162] to that end, a model that was finetuned on the The Corpus of Linguistic Acceptability (COLA) dataset [187]. If the instance does not pass this test, we skip it. Otherwise, we check if we can negate both the premise and the hypothesis of the instance. We use the method developed by Hosseini et al. [84] for this purpose, a rule-based approach with pre-defined rules written in Semgrex [27]. It takes as input a sentence with part-of-speech tags (POS tags), the dependency parse, and the morphological features of the words, and it produces as output a negated sentence. We used Stanza [149] to get the POS tags, the dependency parse, and the morphological features. Here is an example: "The man is somewhere near the parade" $\rightsquigarrow$ "The man is **nowhere** near the parade" .

If both the premise and the hypothesis can be negated, we derive possible new instances as per Table 5.2. We illustrate this data augmentation process with an instance from SNLI[2]:

Example 5.2

Premise:	The two boys are in martial arts poses in an outside basketball court.
Hypothesis:	The two boys are outdoors.
Expected Answer:	$A \triangleright B$ (Entailment)

[2]Since SNLI instances are always about a given scene, we added the determiner "the" here.

Example 5.3: Derivation $A \blacktriangleright \neg B$

Premise: The two boys are in martial arts poses in an outside basketball court.
Hypothesis: The two boys are **not** outdoors.
Expected Answer: Contradiction

Example 5.4: Derivation $\neg B \rhd \neg A$

Premise: The two boys are **not** outdoors.
Hypothesis: The two boys are **not** in martial arts poses in an outside basketball court.
Expected Answer: Entailment

Example 5.5: Derivation $\neg B \blacktriangleright A$

Premise: The two boys are **not** outdoors.
Hypothesis: The two boys are in martial arts poses in an outside basketball court.
Expected Answer: Contradiction

Example 5.6: Derivation $\neg A \not\rhd B$

Premise: The two boys are **not** in martial arts poses in an outside basketball court.
Hypothesis: The two boys are outdoors.
Expected Answer: Not Entailment

This last example should actually be labeled *neutral*, as the boys can be outside without martial arts poses. However, not all pairs of $\neg A$ and B are neutral when $A \rhd B$, they can also be in a contradiction: with A="I live in Paris" and B="I live in the capital of France", we have $A \rhd B$, and $\neg A \blacktriangleright B$. The relation of $\neg A$ and B thus cannot be determined just by knowing $A \rhd B$. However, our approach can still generate a rejected label that can be used for training.

5.4 Experiments

We conducted several experiments to investigate the robustness of models trained with our data augmentation technique, TINA, for the task of textual entailment with negation.

Dataset	*Train*	*Negated Train*
SNLI	550,152	78,116
MNLI	392,702	199,648
RTE	2,490	2,308

Table 5.4: Number of instances in each training dataset that were negated

Dataset	*Train*	*Dev*	*Aug*	*Neg*
SNLI	550,152	10,000	233,024	1,500
MNLI	392,702	9,815	601,441	1,500
RTE	2,490	277	2,408	1,500

Table 5.5: Number of instances in each dataset

5.4.1 Settings

Datasets. We use the most common datasets for textual entailment, namely Stanford Natural Language Inference (SNLI) [24], Multi-Genre Natural Language Inference (MNLI) [196], and Pascal RTE (RTE) [38, 67, 56, 57, 15]. Each dataset comes with a *Train* dataset for training, and a *Dev* dataset for development. Following Hossain et al. [82], we used the development dataset as the testing set because the GLUE [181] and SuperGLUE [182] benchmarks do not provide gold labels for the test splits. To evaluate the understanding of negation in language models, we used the negated variants of SNLI, MNLI, and RTE created by Hossain et al. [82] (a description of each dataset can be found in Chapter 2). Finally, for each dataset, we generate an augmented variant *Aug* by our methodology from Section 5.3. We made sure that the generated instances were not in the negated benchmarks. Table 5.4 shows the number of instances from the training set of each dataset that were negated before deriving new instances. Table 5.5 shows the sizes of the datasets.

Models. We want to see whether TINA makes SLMs more robust to negation in textual entailment. Our experiments cover the following models: BERT, RoBERTa, XLNet, BART, and GPT-2. In Chapter 2, we have described each of these models.

We finetune BERT (Base Cased), RoBERTa (Base), and XLNet (Base Cased) on each training set and evaluate them on each testing set. We use the same hyperparameters as Hossain et al. [82] for the number of epochs, batch size, learning rate, and weight decay. We recall them in Table 5.6. However, unlike the original work, we set the maximum sequence length to 512 instead of 128. We also applied our approach to BART (Base) and GPT-2. We split the training dataset as 90/10 for training and validation sets for these two models. We evaluated on each testing set with the best-performing models based on the validation set. We conducted a basic hyperparameter

search and the hyperparameters used are listed in Table 5.7. All models were trained on an NVIDIA A100 GPU with 40GB memory.

Competitors. The only other approach that specifically targets negation in textual entailment is BERTNOT [84]. It was trained to model negation in the MLM task, and then it was finetuned on each TE training set. For reference, we also show the performance of a T5-Base model. This model is very powerful, as it was pretrained on a mixture of NLP tasks that include textual entailment, coreference resolution, linguistic acceptability, and semantic equivalence.

	SNLI			*MNLI*			*RTE*		
	BERT	*RoBERTa*	*XLNet*	*BERT*	*RoBERTa*	*XLNet*	*BERT*	*RoBERTa*	*XLNet*
Epochs	3	3	3	3	3	3	50	10	50
Batch Size	32	32	32	32	32	32	8	16	8
Learning Rate	1e-5	1e-5	1e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5
Weight Decay	0.1	0.1	0.1	0	0	0	0	0	0

Table 5.6: Hossain et al. [82] hyperparameter configurations

	SNLI		*MNLI*		*RTE*	
	BART	*GPT-2*	*BART*	*GPT-2*	*BART*	*GPT-2*
Epochs	10	10	10	10	10	10
Batch Size	32	32	32	32	8	8
Learning Rate	1e-5	1e-5	2e-5	2e-5	2e-5	2e-5
Weight Decay	0.1	0.1	0	0	0	0

Table 5.7: BART and GPT-2 hyperparameter configurations

5.4.2 Results

Table 5.8 shows the performance of TINA applied to different SLMs averaged over 3 runs. TINA$^-$ is a variant of TINA that does not generate instances with rejected labels. We show, for each model, how the performance changes when TINA$^-$ and TINA are used. We compute a binomial confidence interval for each result (at a confidence level of $\alpha = 0.05$), based on the total number of instances and the number of correctly predicted labels.

The main outcome is that, on the negated datasets, TINA$^-$ always improves the results, and TINA improves the results even more. At the same time, the augmentation techniques do not lower the results significantly on the original datasets. This is true across all models.

On the SNLI dataset, the improvement of the performance is considerable, with gains up to 20 percentage points, depending on the model. On MNLI, the gains are less. We assume that this is because MNLI contains many ungrammatical sentences, and also because it already contains some proportion of negated training examples.

Model	*SNLI*		*MNLI*		*RTE*	
	$SNLI_{Dev.}$	$SNLI_{Neg.}$	$MNLI_{Dev.}$	$MNLI_{Neg.}$	$RTE_{Dev.}$	$RTE_{Neg.}$
BERTNOT [84]	89.00±0.62	45.96±2.53	84.31±0.73	60.89±2.51	69.68±5.65	74.47±2.27
Pretrained T5-Base (Beam Search)	78.61±0.81	60.33±2.56	86.04±0.70	66.46±2.48	66.06±6.12	83.13±2.04
Pretrained T5-Base (Greedy Search)	78.29±0.81	61.40±2.48	85.61±0.71	67.00±2.42	67.87±6.08	82.60±2.00
BERT	89.19±0.62	49.10±2.55	83.38±0.75	65.21±2.45	67.62±5.83	58.30±2.54
+ TINA⁻	+ 0.0	+ 3.56	- 1.33	+ 4.29	+ 0.84	+ 21.63
+ TINA	- 0.21	+ 20.09	- 2.81	+ 4.21	-	-
RoBERTa	90.18±0.59	54.46±2.58	86.55±0.69	66.93±2.48	76.54±5.33	74.35±2.28
+ TINA⁻	- 0.1	+ 0.89	- 0.45	+ 1.62	+ 0.11	+ 7.18
+ TINA	- 0.05	+ 13.05	- 0.45	+ 2.04	-	-
XLNet	89.98±0.60	53.77±2.56	85.76±0.70	67.06±2.48	70.15±5.75	68.08±2.41
+ TINA⁻	- 0.26	+ 2.31	- 0.31	+ 3.03	- 5.66	+ 6.65
+ TINA	- 0.34	+ 12.8	- 1.01	+ 3.8	-	-
BART	89.79±0.60	53.17±2.56	84.90±0.73	66.60±2.49	70.51±5.73	60.30±2.53
+ TINA⁻	- 0.20	- 0.6	- 0.65	+ 3.04	+ 0.10	+ 17.03
+ TINA	- 0.09	+ 17.6	- 1.37	+ 3.66	-	-
GPT-2	87.56±0.66	48.77±2.55	80.94±0.79	62.24±2.52	61.97±6.08	57.37±2.55
+ TINA⁻	+ 0.04	+ 2.09	- 0.37	+ 4.73	+ 4.45	+ 17.56
+ TINA	+ 0.01	+ 6.67	- 0.42	+ 5.93	-	-

Table 5.8: Results of our approach applied to different language models on different textual-entailment datasets. Accuracies are averaged across 3 runs. Significant changes have a gray background.

Nevertheless, the gains of TINA are still significant. On RTE, TINA and TINA⁻ are identical, as the dataset only has two labels (*entailment* and *non-entailment*). The confidence intervals on RTE_{Dev} are much larger, because the dataset is much smaller. Nevertheless, the gains on the negated dataset are significant, and can reach up to 21 percentage points, depending on the model.

For reference, we also show the performance of an off-the-shelf pretrained T5-Base model. It has a very good performance, and most notably outperforms our competitor BERTNOT significantly on the negated datasets. We assume that this is because it was pretrained on a large mixture of NLP tasks. Nevertheless, our method comes close to T5 on RTE, and outperforms the T5 model on SNLI and MNLI.

Most importantly, however, our approach serves its purpose in that it increases the performance of SLMs on negated textual entailment by a large margin across different models and all datasets. With this, our approach improves over the current state-of-the-art [84]. Figure 5.1 shows a graphical illustration of the performances in Table 5.8 of the tested models.

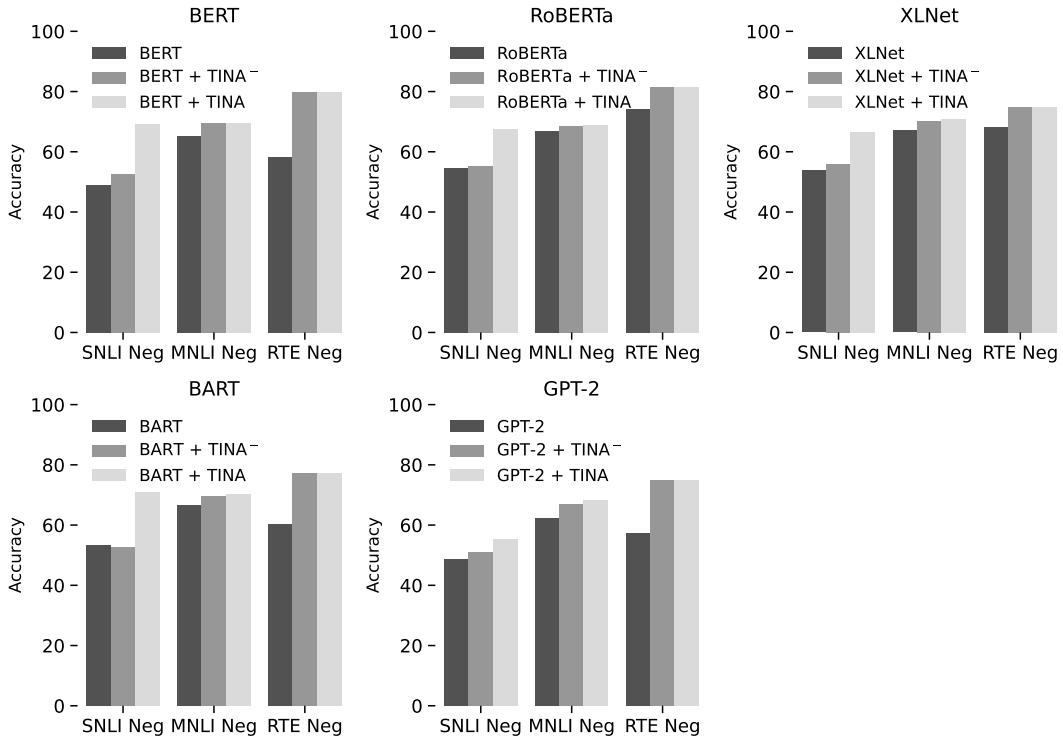


Figure 5.1: Evaluation of different finetuning methods applied to different SLMs on the negated textual entailment datasets. Accuracies are averaged across 3 runs.

5.4.3 Qualitative Analysis

To better understand the performances of TINA, we manually checked a sample of sentences from each augmented dataset. For SNLI, we find that the sentences are simple. They just contain one verb, which is easy for Hosseini et al. [84]'s tool to negate. In contrast, MNLI and RTE have longer and more complex premises, which are not always grammatical. This leads to problematic cases where the negation does not work, which we group into the following categories:

Ungrammatical sentences cannot be negated properly: "would i swim that river every night twice if that's what it took you know i don't care whatever it would take i have real sympathy for those people i really do and you can." $\rightsquigarrow$ "would **not** i swim that river every night twice if that 's what it took you know i don't care whatever it would take i have real sympathy for those people i really do and you can."

Conjunctions are negated only in their first conjunct: "The motion set waves of nausea running through him, but he could see the doctor" $\rightsquigarrow$ "The motion **did not** set waves of nausea running through him, but he could see the doctor" . The same goes for adjectives and prepositions that take a role akin to a conjunction, as in "despite concerns about the drinking water".

Verbs of assertion are negated, but not the assertion itself: "The actor was outside a movie theater in central London's Leicester Square, London's Metropolitan Police said" $\rightsquigarrow$ "The actor was outside a movie theater in central London's Leicester Square, London's Metropolitan Police **did not** say". In this case, the negation does not work as intended, as the main verb merely states the source of the assertion. In other cases, the main verb may indeed be the intended target of the negation.

Negation errors occur at times with Hosseini et al. [84]'s tool, as e.g. in "cannot not do" and "has did not given".

Our filtering step with DistillBERT [162] was apparently insufficient to remove the ungrammatical sentences. For the conjuncts, we found that the erroneous negation is mostly harmless: if a conjunction is negated only in its first conjunct, that might still be the conjunct that is relevant for the entailment. The same goes for verbs of assertion: the entailment may sometimes target the fact of asserting something (in which case the negation works correctly). Negation errors, too, may be harmless: while these can disturb a human reader, they may still yield useful signals for a machine learning model.

The negation of sentences thus remains a challenge in practice. It is, however, largely orthogonal to our contribution of creating negated training examples for textual entailment. We are thus hopeful that an improvement of these tools will confer even higher performance gains to TINA.

5.5 Conclusion

In this chapter, we have studied the problem of negation in textual entailment in detail. We have argued that the previous formal definition of textual entailment is problematic, and we have proposed a new probabilistic definition. Based on this definition, we have proposed TINA, a principled negated data augmentation technique. TINA can be combined with the unlikelihood loss to improve the robustness of language models to negation in textual entailment tasks. Our experimental results across different negated textual entailment benchmarks show that our method can significantly increase the performance of different SLMs. Future work can explore how different loss functions, such as contrastive loss, could be used with our augmented datasets.

6

MAFALDA: A Benchmark and Comprehensive Study of Fallacy Detection and Classification

In the previous chapters, we have considered smaller LMs. In this chapter, we aim to explore the frontier of reasoning capabilities of LLMs, with a focus on GPT-3.5 as our case study. We also assess the performance of SLMs. We study fallacy classification and detection, a complex task requiring high-level reasoning even for humans. We introduce MAFALDA, a benchmark for fallacy classification that merges and unites previous fallacy datasets. It comes with a taxonomy that aligns, refines, and unifies existing classifications of fallacies. We further provide a manual annotation of a part of the dataset together with manual explanations for each annotation. We propose a new annotation scheme tailored for subjective NLP tasks and a new evaluation method designed to handle subjectivity. On MAFALDA, we evaluate several LMs under a zero-shot learning setting and human performances to assess their capability to detect and classify fallacies. This chapter is based on the following paper [74].

> ***Chadi Helwe**, Tom Calamai, Pierre-Henri Paris, Chloé Clavel, Fabian Suchanek. **"MAFALDA: A Benchmark and Comprehensive Study of Fallacy Detection and Classification"** (long paper) NAACL 2024*

6.1 Introduction

A fallacy is an erroneous or invalid way of reasoning. Consider, e.g., the argument "*You must either support my presidential candidacy or be against America!*". This argument is a *false dilemma* fallacy: it wrongly assumes no other alternatives. Fallacies can be found in various forms of communication, including speeches, advertisements [39], Twitter/X posts [116], and political debates [10, 60]. They are also part of propaganda techniques employed to shape public opinion and promote specific agendas. Most notably, fallacies played a role in the 2016 Brexit referendum [211], and the debate

about COVID-19 vaccinations [48], where fake news spread on news outlets and in social networks [118, 160, 10]. Detecting and identifying these fallacies is thus a task of broad importance.

The recent advances in deep learning and the availability of more data have given rise to approaches for detecting and classifying fallacies in text automatically [118, 4, 10, 160, 1]. And yet, this work is fragmented: most approaches focus on specific types of corpora (e.g., only speeches) or specific types of fallacies (e.g., only *ad hominem* fallacies). Furthermore, not all works use the same types of fallacies, there is no consensus on a common terminology [68], and fallacies come at different levels of granularity: an *appeal to emotion* can be, for instance, an *appeal to anger*, *fear*, *pride*, or *pity*. Most importantly, annotating fallacies is an inherently subjective task. While previous works acknowledge the subjectivity, none explicitly embraces it. On the contrary, the annotators typically aim for a unique annotation – by discussion or vote. Additionally, existing works do not give human performances on the benchmarks and evaluate only models.

This chapter addresses these drawbacks by introducing the Multi-level Annotated Fallacy Dataset MAFALDA – a manually created fallacy classification benchmark. Our contributions are:

- A taxonomy of fallacies that aligns, consolidates, and unifies existing public fallacy collections (Section 6.3).
- A new annotation scheme – coined disjunctive annotation scheme – that accounts for the inherent subjectivity of fallacy annotation by permitting several correct annotations (Section 6.4).
- A corpus that merges existing corpora, with 9,545 non-annotated texts and 200 manually annotated texts with 260 instances of fallacies, each with a manual justification (Section 6.5).
- A study of the performance of state-of-the-art language models and humans on our benchmark (Section 6.6).

All our code and data are publicly available under a CC-BY-SA license[1] at `https://github.com/ChadiHelwe/MAFALDA`, allowing our study to be reproduced and built upon. We start our chapter by discussing related work in Section 6.2.

6.2 Related Work

6.2.1 Datasets

Numerous works have created datasets of fallacies. Habernal et al. [65] created a dataset for *ad hominem* fallacies from the "Change My View" subreddit. Martino et al. [118] created a news article dataset featuring 18 fallacies such as *red herring, appeal to authority, bandwagon, etc.*. Balalau and Horincar [10] developed a dataset from online

[1] as imposed by the dataset from Goffredo et al. [60]

forums for the task of identifying propaganda techniques. Sahai et al. [160] compiled a Reddit-based corpus for fallacy detection with eight types of fallacies. Goffredo et al. [60] introduced a dataset from American political debates with six different fallacy types. Along the same line, Jin et al. [92] curated a claim dataset, containing 13 types of fallacies, based on online quizzes and the Climate Feedback website, employing a novel approach that mimics first-order logic. To address data annotation challenges, Habernal et al. [63] created the Argotario game for fallacy detection in QA pairs. It created a corpus of 5 fallacy types. In the domain of (dis/mis)information, Musi et al. [128] and Alhindi et al. [5] annotated fallacies in COVID-19 and climate change articles with ten types of fallacies. Lastly, Payandeh et al. [139] developed LOGICOM to evaluate Large Language Models' (LLMs) robustness against logical fallacies in debate scenarios. While all of these works advanced the understanding of fallacy detection, the studied fallacies are not the same across different works and are sometimes outright disjoint. The only work that creates a comprehensive taxonomy of fallacies is the (not yet peer-reviewed) work of Hong et al. [80]. However, this work enumerates 232 fallacies, which is clearly too many to be handled by a human. And indeed, their dataset is composed only of toy examples generated by GPT-4.

In this chapter, we propose a benchmark that not only unifies public datasets on fallacy detection in a handy yet all-embracing taxonomy, but also comes with human annotations, human explanations, and evaluations for both language models and humans.

6.2.2 Subjectivity and Annotation Challenges

Human label variation is inherently part of annotating complex and subjective tasks [146]. It is usually addressed with strategies such as simplifying the task, majority votes, or reconciliation of discrepancies. Goffredo et al. [60] computed the Krippendorff's α on a subset of fallacies and reached inter-annotator agreements (IAAs) ranging from 0.46 to 0.60, which is a moderate agreement. On simpler tasks such as identifying only *ad hominem* using two groups of 6 workers, Habernal et al. [65] reported a Cohen's κ of 0.79, which is a good agreement. However, they acknowledge the difficulty of annotated sub-categories such as *tu quoque* and *guilt by association* (they found a low IAA, but the value is not provided). When annotating spans of propaganda techniques, a complex task, Martino et al. [118] found a γ IAA of 0.26, which is low. However, they could increase the IAA up to 0.60 when adding a reconciliation step. In Sahai et al. [160], the annotator had to identify one fallacy at a time, and they reached a Cohen's κ of 0.515 (ranging from 0.38 to 0.64 based on the fallacy), which is a moderate agreement. They also computed the γ for the span selection per fallacy type and found values between 0.60 to 0.80, which is a good agreement. This was expected since it is a binary classification task. Sahai et al. [160], Jin et al. [92], Musi et al. [128], Alhindi et al. [5] used a reconciliation step too to tackle discrepancies in the annotations.

In summary, IAA in related work is usually only moderate. Disagreements are interpreted as noise, and are removed with various strategies. In this chapter, we propose not just to acknowledge the subjectivity of fallacy annotation but actually

to follow through with it. We contend that there are cases where multiple, equally valid annotations can coexist for the same textual span. Therefore, we propose a new subjective annotation scheme that allows for several alternative labels for the same span.

6.2.3 Taxonomies of Fallacies

Logical fallacies have been studied and classified since the time of Aristotle. There is a notable diversity in approaches and contents across various sources. The works of Aristotle (see Wikipedia contributors [195]) and Whately [193], despite their historical significance, present limitations in terms of the breadth of fallacies covered, listing only 13 fallacies each (our work, in contrast, finds more than 20). Downes [46] offers a more extensive list with 36 fallacies. However, it still fails to mention common fallacies such as *appeal to nature*, *appeal to tradition*, and *guilt by association*. Curtis [37] provides an exhaustive list of 87 fallacies. Yet, it provides only a rudimentary hierarchy (classifying, e.g., *no true Scotsman* as a sub-category of *equivocation*). Fallacies [50] lists 48 fallacies – but lacks a hierarchical framework altogether. At the other end of the spectrum, Dowden [45], Bennett [14], Hong et al. [80], and Wikipedia [194] offer extensive compilations of 231, 300+, 232, and 149 fallacies respectively. Yet, such a sheer volume of fallacies would be challenging in practical annotation tasks, as the annotator would have to scan (or memorize) hundreds of different fallacies.

Our work, in contrast, is driven by today's practical application scenarios. It aims to systematize and classify the fallacies used in current works on fallacy annotation, detection, and classification.

6.3 A Unified Taxonomy of Fallacies

6.3.1 Definitions

We start with the definition of an argument, following Copi et al. [34], Britannica [26]:

Definition 6.1: Argument

An argument consists of an assertion called the *conclusion* and one or more assertions called *premises*, where the premises are intended to establish the truth of the conclusion. Premises or conclusions can be implicit in an argument.

Thus, an argument is typically of the form "***Premise***$_1$***:*** *All humans are mortal.* ***Premise***$_2$***:*** *Socrates is human.* ***Conclusion:*** *Therefore, Socrates is mortal.*". However, premises and conclusion can also appear in the opposite order and/or in the same sentence, as in "*Socrates is mortal because he is a human and all humans are mortal*". In many real-world arguments, the premise and the conclusion are spread apart (as in "*Of course, Socrates is mortal! How can you doubt this? After all, he's human, and all*

humans are mortal!"). Sometimes, premises are left implicit (as in "*Socrates is mortal because he is human*"). Even the conclusion can be implicit (as in "*Socrates is human and all humans are mortal*"). In the context of a discussion, an argument can attack another argument [47], in which case the conclusion is implicitly negated ("*Socrates is immortal! – But he is human!*").

A valid argument is one where the truth of the premises guarantees the truth of the conclusion; otherwise, following Copi et al. [34], Britannica [26], the argument is a fallacy:

Definition 6.2: Fallacy

A fallacy is an argument where the premises do not entail the conclusion.

We refer the reader to Chapter 5 for a discussion of a formal definition of textual entailment.

6.3.2 Taxonomy of Fallacies

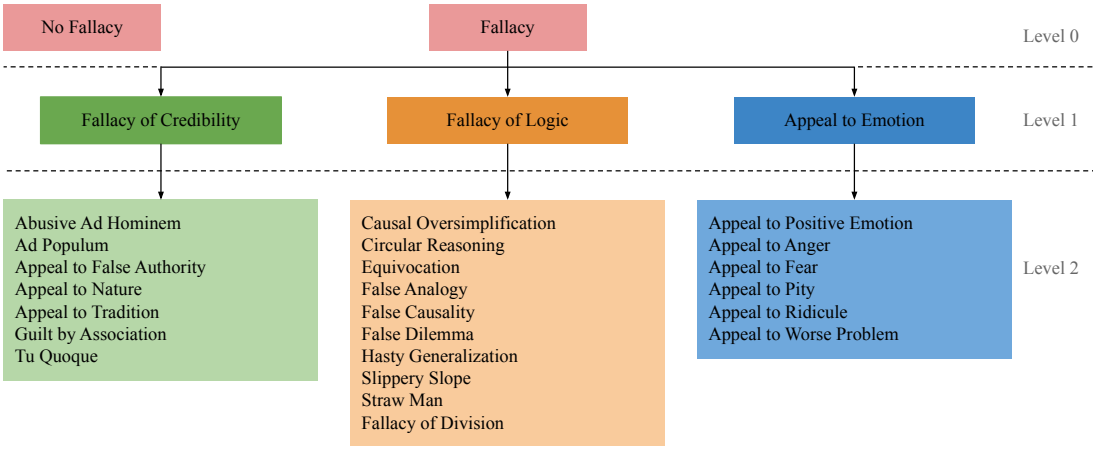


Figure 6.1: Tree structure of our taxonomy. Detailed definitions of the fallacies are in Appendix B.1.

In this chapter, we propose a taxonomy that unifies and consolidates all types of fallacies used in current work on fallacy detection. We built our taxonomy manually, starting with a collection of fallacy types that are used in related work. Since the same fallacy can appear in different datasets under different names, we aligned equivalent fallacies manually. We used the definitions and guidelines in the source datasets to determine whether two fallacies are equivalent. We removed fallacies that were too broad (e.g., *appeal to emotion* could cover many emotions), fallacies that appeared in only a single work (e.g., *confusion fallacy* appears only in Martino et al. [118]), and we merged fallacies that were too similar in their definitions (like *begging the question* and *circular reasoning*). Some fallacies were not taken into account because they were not actually fallacies in our definition. These are, e.g., rhetorical techniques such as

"We know God exists because he made everything."

(a) Circular Reasoning

"if you would have supported that attitude, we would not have won the Cold War. We won the Cold War because we invested and we went forward." (APPLAUSE.)

(b) Causal Oversimplification

"In the last New Hampshire primary election my favorite candidate won. Therefore, he will also win he next primary election."

(c) Causal Oversimplification **OR** False Causality

"TITLE: Can I get into finance with a Law degree? POST: (...) This is law school arrogance at its finest. Why not a brain surgeon?"

(d) Appeal to Ridicule

Figure 6.2: Examples of Fallacies. The spans of the fallacies are underlined. Example 6.2a is from Jin et al. [92], 6.2b from Goffredo et al. [60], and 6.2d from Sahai et al. [160].

flag waving or *repetition*. Our list can obviously be extended in the future with new fallacies.

We grouped our fallacies into broader categories to create a taxonomy on top of our collection. We chose the categories that Aristotle originally proposed [198], because it has been shown to be applicable across various forms of communication – from political speeches to advertisements [198]. This yields the following taxonomy:

1. **Level 0** is a binary classification, categorizing text as either fallacious or non-fallacious.
2. **Level 1** groups fallacies into Aristotle's categories: 'Pathos' (appeals to emotion), 'Ethos' (fallacies of credibility), and 'Logos' (fallacies of logic, relevance, or evidence).
3. **Level 2** contains fine-grained fallacies within the broad categories of Level 1. For instance, under *fallacy of credibility*, we have specific fallacies such as *appeal to tradition*, *ad populum*, and *guilt by association*.

Previous works have studied a large number of different fallacy types. The earliest works focused on *ad hominem*, while later works included dozens of other types. To build our taxonomy (as shown in Figure 6.1), we tried to unify most fallacy types in the literature. Table 6.1 shows each type of fallacy studied by each paper that proposed a dataset. Most fallacies from our taxonomy are part of at least two already existing datasets. Based on our definition, rhetorical techniques that are not based on an actual argument are not considered fallacies. Thus, we did not include techniques such as *repetition* or *slogans*. During the initial annotation phase, we observed that the *red herring* fallacy was too vague, so we replaced it with more precise sub-categories, such as *appeal to worse problem*. This explains why *appeal to worse problem* , which is present in only one other dataset, is part of our taxonomy. Similarly, during the

annotation, we found multiple examples of *fallacy of division*, which is related to *hasty generalization* but does not fit its description. Hence, we added *fallacy of division* in the taxonomy.

For each fallacy, we provide both a formal and an informal definition in Appendix B.1 (inspired by Bennett [14]). For instance, the *appeal to ridicule* is informally defined as "an argument that portrays the opponent's position as absurd or ridiculous with the intention of discrediting it.". Formally, it is defined as "E_1 claims P. E_2 makes P look ridiculous, by misrepresenting P (P'). Therefore, $\neg P$.", where E_i are entities (e.g., people, organizations, etc.), and P is a proposition. Let's take Example (d) from Figure 6.2, which shows an *appeal to ridicule*: the post argues against the possibility of working in finance with a law degree by exaggerating the position and thus portraying it as ridicule. Breaking down the example with the formal definition yields:

Example 6.1

TITLE: Can I get into finance with a Law degree? POST: (...) This is law school arrogance at its finest. Why not a brain surgeon?

- E_1= The original poster
- P= It might be possible to work in finance with a law degree
- E_2= The author of the post.
- P'= Law school students are so intelligent that they can do any job, even surgeons.

Here, E_i are entities (persons, organizations) or groups of entities, P and P' are premises, properties, or possibilities.

6.4 Disjunctive Annotation Scheme

6.4.1 Subjectivity in Fallacy Annotation

Annotating fallacies is an inherently subjective endeavor. To see this, consider Example (c) in Figure 6.2. The argument goes that the candidate has to win again because he won last time. This can be seen as a *false causality* fallacy: a cause-effect relationship is incorrectly inferred between two events that have nothing to do with each other. However, it can also be seen as a *causal oversimplification* fallacy. This is because we can contend that having won the last election gives the candidate an edge over other candidates in terms of visibility, and thus makes it more likely that he wins this year's election as well. The argument is thus fallacious mainly because it fails to acknowledge other factors that play a role in re-election.

This simple example already shows subjectivity in fallacy annotations, where several annotations can be defended. It would be counter-productive if the annotators converged on, say, *causal oversimplification*, so that every approach of fallacy annotation is penalized for proposing an (equally plausible) *false causality*. There are other

Our Taxonomy				Alhindi et al. [5]							
Level 1	Level 2	Habernal et al. [65]	Delobelle et al. [42]	Martino et al. [118] [10]	Jin et al. [92]	Musi et al. [128]	Habernal et al. [63]	Goffredo et al. [60]		Reisert et al. [156]	Sahai et al. [160]	Hong et al. [80]
Fallacy of Credibility	Abusive Ad Hominem	abusive	abusive attack		Ad Hominem				General			
	Tu Quoque	tu quoque		whataboutism					Tu quoque			
		circumstantial bias					ad hominem	ad hominem	Bias ad hominem			
	Guilt by Association	guilt by association		reductio ad hitlerum								
			name calling	name calling					Name-calling			
	Ad Populum			doubt bandwagon	Ad Populum				Popular opinion		Appeal to Majority	
	Appeal to Nature										Appeal to Nature	
	Appeal to Tradition										Appeal to Tradition	
	Appeal to False Authority			appeal to false authority	Fallacy of Credibility	Appeal to Inappropriate Authority	irrelevant authority	Appeal to authority	False authority without evidence		Appeal to Authority	
Fallacy of Logic	Causal Oversimplification			causal oversimplification		Evading the Burden of Proof						Probability Proposition Quantification Syllogism Ambiguity Inconsistency Irrelevance Insufficiency Inappropriate Presumption
	Hasty Generalization				Faulty Generalization	Cherry Picking of Evidence Hasty Generalization	Hasty Generalization			Hasty Generalization	Hasty Generalization	
	False Causality				False Causality	False Cause Post Hoc (Correlation presented as Causation)		False cause		Questionable Cause		
	Circular Reasoning				Circular Claim					Begging the Question		
	False Dilemma			black-and-white fallacy	False Dilemma						Black-or-White	
	Slippery Slope								Slippery Slope		Slippery Slope	
	False Analogy					False Analogy						
	Straw Man			straw man	Fallacy of Extension	Strawman						
	Fallacy of Division				Deductive Fallacy							
	Equivocation			red herring obfusc. int. vagueness confusion	Fallacy of Relevance Equivocation	Red Herring Vagueness	Red Herring			Red Herring		
		Technique based on use of vocabulary		thought-terminating cliches exaggeration/minimization repetition slogans loaded language				Slogan	Loaded Language			
Fallacy of Emotion	Appeal to Positive Emotion			flag-waving					Flag waving			
	Appeal to Anger Appeal to Fear			appeal to fear/prejudice	Appeal to Emotion		Appeal to Emotion	Appeal to Emotion	Appeal to fear			
	Appeal to Pity								Appeal to pity			
	Appeal to Ridicule											
	Appeal to Worse Problem										Appeal to Worse Problems	
				red herring	Intentional Fallacy of Relevance	Red Herring				Red Herring		

Table 6.1: List of fallacies per paper and in our taxonomy.

cases of legitimately differing opinions: One annotator may see implicit assertions that another annotator does not see. In "*Are you for America? Vote for me!*" one reader may see the implicit "*or you must be against America*" (which makes this a *false dilemma*), while another annotator may see no such implicature. Annotators may also have different thresholds for fear (*appeal to fear*) or insults (*ad hominem*).

Finally, different annotators have different background knowledge: A sentence such as "*Use disinfectants or you will get Covid-19!*" may be read as a plausible warning by one annotator but as an *appeal to fear* fallacy by an annotator who knows that Covid-19 does not spread via contaminated surfaces. We will now present a disjunctive annotation scheme that accounts for this inherent subjectivity.

6.4.2 Annotating with Alternatives

Before presenting our annotation scheme, we need to establish some common ground:

Definition 6.3: Text

A text is a sequence of sentences $st_1, \ldots, st_n$.

A *span* on a text is a contiguous sequence of sentences. The set S of all spans of a text $st_1, \ldots, st_n$ is thus $S = \{st_i \ldots st_j \mid 0 < i \leq j \leq n\}$.

Definition 6.4: Span

The span of a fallacy in a text is the smallest contiguous sequence of sentences that comprises the conclusion and the premises of the fallacy. If the span comprises a pronoun that refers to a premise or to the conclusion, that premise or conclusion is not included in the span.

We work on the level of sentences, because previous work has shown that agreement on the token level is even harder to achieve [92]. We allow the use of pronouns to decrease the size of the spans: When a sentence refers to another sentence by a pronoun, that other sentence does not have to be part of the span. For instance, in Example (d) of Figure 6.2, the premise of the fallacy is in the title of the post, and the conclusion is at the end of the text. Thus, a span that covers the entire fallacy would have to cover the entire post from title to end. However, the pronoun "This" refers to the title, and thus we can omit the title from the span. Nevertheless, a span can comprise several sentences.

A span can be annotated with a label (such as a fallacy type) by an annotator (or a group of them) or by a system. We now propose the key element of our disjunctive annotation scheme, in which subjectivity is not projected away, but explicitly embraced by allowing for several equally valid labels for the same span.

Definition 6.5: Gold Standard

Let $\mathcal{F}$ be the set of fallacy types and $\perp$ be a special label that means "no fallacy".
Given a text and its set of spans S, a gold standard G is a set of pairs of a span $s \in S$ and a set of labels from $\mathcal{F} \cup \{\perp\}$:

$$G \subseteq S \times (\mathcal{P}(\mathcal{F} \cup \{\perp\}) \setminus \{\emptyset, \{\perp\}\})$$

Here, $\mathcal{P}(\cdot)$ denotes the powerset.

The gold standard associates a given span with one or more fallacy labels. If more than one label is present, this means that any label is acceptable as an annotation. The gold standard can also associate a span with $\perp$, which means that the annotation of this span is optional. However, in this case, the gold standard has to associate the span also with at least one other label, as we are not interested in annotating non-fallacious sentences. The gold standard can also contain the same span twice, which means that the span has to be annotated with two labels. The alternative labels for a span can be generated through various methods during the annotation process, e.g., one annotator giving alternatives, a group of annotators proposing different labels due to lack of consensus, or multiple independent annotators combining their labels (see Example 6.4.2).
We define a prediction as the annotation of a text by a system or a user:

Definition 6.6: Prediction

Given a set of fallacy types $\mathcal{F}$, a text, and its set of spans S, a prediction P is a set of pairs of a span $s \in S$ and a label $l \in \mathcal{F}$:

$$P \subseteq S \times \mathcal{F}$$

The following example gives substance to these definitions:

Example 6.2

Let "$a\ b\ c\ d$" be a text where a, b, c, and d are sentences.
Suppose $S = \{a\ b, d\}$ (i.e., the sentences a and b are one fallacious span, and d is a span of one fallacious sentence), $a\ b$ has labels $\{l_1, l_2\}$, and d has label $\{l_3\}$. In that case, $G = \{(a\ b, \{l_1, l_2\}), (d, \{l_3\})\}$
An example of prediction P could be $P = \{(a, l_1), (a, l_2), (b, l_3), (c, l_4), (d, l_1)\}$

6.4.3 Evaluation Metrics

To compare two annotated spans, we adapt the precision and recall of Martino et al. [118] to alternatives. Given two spans, p with its label l_p, and g with its set of labels l_g,

respectively, and a normalizing constant h, these metrics compute a comparison score as follows:

$$C(p, l_p, g, l_g, h) = \frac{|p \cap g|}{h} \times \delta(l_p, l_g)$$

δ is a similarity function. We use $\delta(x, y) = [x \in y]$, where $[\cdot]$ is the Iverson bracket.

Let G be the gold standard, and let P be the prediction of a user or a system. The precision for P is computed by comparing each span in P against all spans in G, and taking the score of the best-matching one:

$$\textit{Precision}(P, G) = \frac{\sum_{(p,l_p)\in P} \max_{(g,l_g)\in G} C(p, l_p, g, l_g, |p|)}{|P|}$$

If there are no annotations in P (i.e., $|P| = 0$), we set precision to 1. This choice is inspired by the intuition that a loss in precision should result only from false predictions. If there are no such false predictions, then precision should not be harmed. Figure 6.3 shows an example of the calculation of our precision.

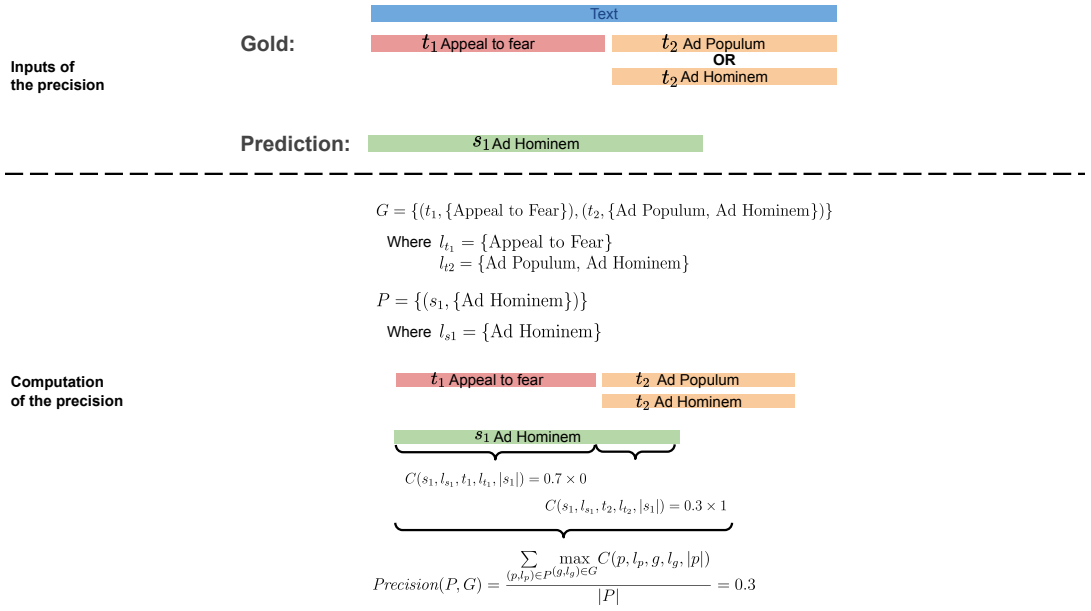


Figure 6.3: Example of Precision computation with alternatives.

To calculate the recall, we exclude all the spans from the gold standard that contain a $\perp$. The rationale for this choice is that when a span has also been marked as "no fallacy", its annotation is considered optional. Therefore, we do not want to penalize models that do not provide an annotation for such a span. We define the set G^-, which is G restricted to the spans that do not map to $\perp$, i.e., $G^- = \{(s, L) \in G \mid \perp \notin L\}$. The recall is then computed as:

$$\textit{Recall}(P, G) = \frac{\sum_{(g,l_g)\in G^-} \max_{(p,l_p)\in P} C(p, l_p, g, l_g, |g|)}{|G^-|}$$

If $|G^-| = 0$, we set the recall to 1. The intuition is that a model should be penalized in recall only for the annotations it misses from the gold standard. If there are no such missed annotations, recall should not suffer. Figure 6.4 shows an example of the calculation of our recall. Appendix B.2 shows how our metrics handle various edge cases.

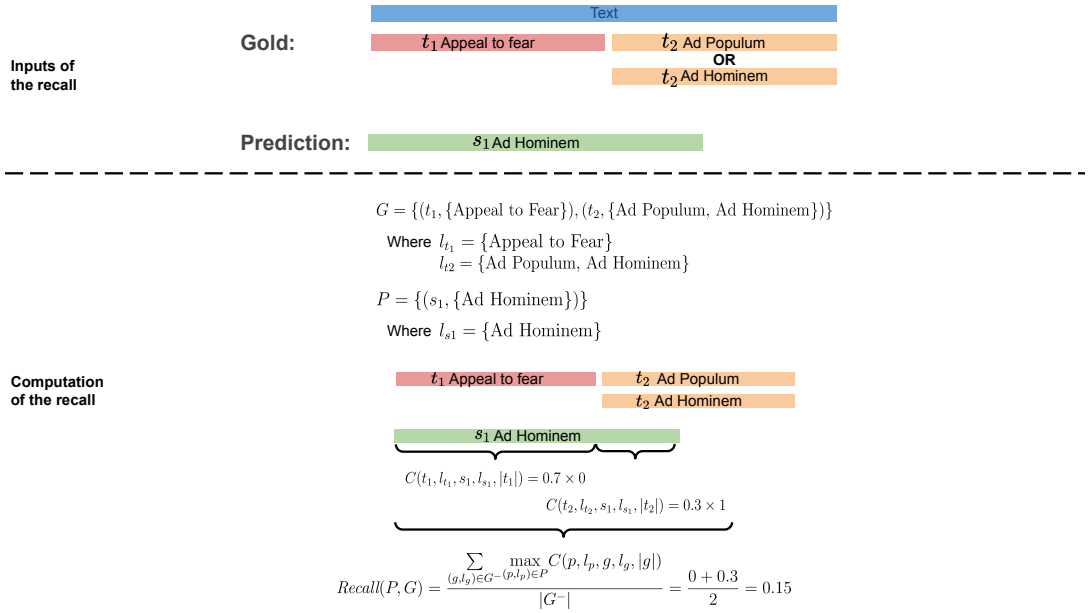


Figure 6.4: Example of Recall computation with alternatives.

The F1-score is computed as usual as the harmonic mean of precision and recall. It is easy to see that our definitions of precision and recall fall back to the standard definitions if G does not have alternatives and all spans comprise only a single sentence. In that case, the score C is 1 if the spans are identical and identically labeled. We show these in Propositions 6.4.1 and 6.4.2.

Proposition 6.4.1. *Given a gold standard G, where each span comprises only a single sentence, and where each fallacy set contains only one element, and given a prediction P, where each span comprises only a single sentence, our precision coincides with the standard precision.*

Proof: By definition, we have, for any spans p, g, and for any label l_p, and set of labels l_g:

$$C(p, l_p, g, l_g, |p|) \tag{6.1}$$

$$= \frac{|p \cap g|}{|p|} \times \delta(l_p, l_g) \tag{6.2}$$

$$= \frac{|p \cap g|}{|p|} \times [l_p = l_g] \tag{6.3}$$

If p and g are singleton spans, this boils down to

$$= [p = g] \times [l_p = l_g] \tag{6.4}$$
$$= [p = g \wedge l_p = l_g] \tag{6.5}$$

Thus, we have, for any singleton span s and any label l:

$$[(s, \{l\}) \in G] \tag{6.6}$$
$$= [\exists (s', \{l'\}) \in G : s' = s \wedge l' = l] \tag{6.7}$$
$$= [\exists (s', \{l'\}) \in G : C(s, l, s', l', |s|) = 1] \tag{6.8}$$
$$= \max_{(s', \{l'\}) \in G} C(s, l, s', l', |s|) \tag{6.9}$$

This entails that the number of true positives (TP) is

$$|\{(s, l) \in P \mid (s, \{l\}) \in G\}| \tag{6.10}$$
$$= \sum_{(s,l) \in P} [(s, \{l\}) \in G] \tag{6.11}$$
$$= \sum_{(s,l) \in P} \max_{(s', \{l'\}) \in G} C(s, l, s', l', |s|) \tag{6.12}$$

The standard precision is the ratio of true positives (TP) out of the sum of true positives and false positives (FP):

$$\text{Standard Precision} \tag{6.13}$$
$$= \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{6.14}$$

With $|P| = \text{TP} + \text{FP}$ and Equation 6.12, this is equivalent to

$$= \frac{\sum_{(s,l) \in P} \max_{(s', \{l'\}) \in G} C(s, l, s', l', |s|)}{|P|}$$

□

Proposition 6.4.2. *Given a gold standard G, where each span comprises only a single sentence, and where each fallacy set contains only one element, and given a prediction P, where each span comprises only a single sentence, our recall coincides with the standard recall.*

Proof: As previously, for any p, g that are singleton spans, we have:

$$C(p, l_p, g, l_g, |g|) \tag{6.15}$$
$$= [p = g \wedge l_p = l_g] \tag{6.16}$$

Thus, we have, for any singleton span s and any label l:

$$
\begin{aligned}
& [(s', l') \in P] && (6.17)\\
= \; & [\exists (s, l) \in P : s = s' \wedge l = l'] && (6.18)\\
= \; & [\exists (s, l) \in P : C(s, l, s', l', |s'|) = 1] && (6.19)\\
= \; & \max_{(s,l) \in P} C(s, l, s', l', |s'|) && (6.20)
\end{aligned}
$$

This entails that the number of true positives (TP) is

$$
\begin{aligned}
& |\{(s', \{l'\}) \in G^- \mid (s', l') \in P\}| && (6.21)\\
= \; & \sum_{(s', \{l'\}) \in G^-} [(s', l') \in P] && (6.22)\\
= \; & \sum_{(s', \{l'\}) \in G^-} \max_{(s,l) \in P} C(s, l, s', l', |s'|) && (6.23)
\end{aligned}
$$

The standard recall is the ratio of true positives (TP) out of the sum of true positives and false negatives (FN):

$$
\begin{aligned}
& \text{Standard Recall} && (6.24)\\
= \; & \frac{\text{TP}}{\text{TP} + \text{FN}} && (6.25)
\end{aligned}
$$

With $|G^-| = \text{TP} + \text{FN}$ and Equation 6.23, this is equivalent to

$$= \frac{\sum_{(s, \{l\}) \in G^-} \max_{(s', l') \in P} C(s, l, s', l', |s|)}{|G^-|}$$

□

If there are no alternatives, our measures are also identical to the ones in Martino et al. [118], with one difference: we use the $\max$ instead of a sum in the definitions to select the best matching span. In this way, two neighboring spans with the same label do not achieve full precision or recall if the gold standard requires one contiguous span with that label. Here is an example:

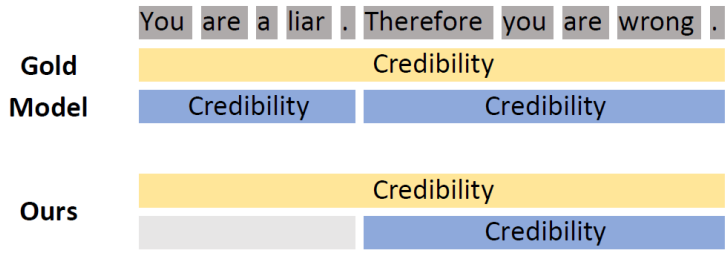


Figure 6.5: Illustration of the difference between our metric and the one from Martino et al. [118].

In this example, both annotated spans are counted by Martino's metric and we get a recall of 1. Our metric, however, counts only the largest overlapping span (light blue), resulting in a recall of 0.6.

Using $\max$ instead of a sum also avoids the case where Martino et al. [118]'s metric yields precision or recall scores exceeding one at Levels 0 and 1. This occurs when the gold standard contains overlapping spans with identical labels, affecting precision, or when the prediction includes such overlaps, affecting recall. Here is an example:

Example 6.3

You are a liar. Therefore, you are wrong.

In this example, there is only one *abusive ad hominem*, which is a *Fallacy of Credibility* on Level 1. Now assume that the model outputs: (You are a liar, *abusive ad hominem*), (You are a liar, therefore you are wrong, *tu quoque*). Using the recall from [118], and $G = \{([0, 40], \text{CREDIBILITY})\}$, $P = \{([0, 10], \text{CREDIBILITY}), ([0, 40], \text{CREDIBILITY})\}$ we get the following recall:

$$\begin{aligned} Recall_m(P, G) &= \frac{1}{|G|} \sum_{p \in P, g \in G} C_m(p, g, |g|) \\ &= \frac{1}{1} * (\frac{10}{40} * 1 + \frac{40}{40} * 1) \\ &= 1.25 \end{aligned}$$

Instead of computing one score for each element of G as it would be expected for the recall, the metrics is computing all scores between all spans with the same label. We thus get a score larger than one.

Hence, we propose to sum only the best match for each element of G.

$$
\begin{aligned}
\textit{Recall}(P, G) &= \frac{\sum_{(g,l_g) \in G^-} \max_{(p,l_p) \in P} C(p, l_p, g, l_g, |g|)}{|G^-|} \\
&= \frac{1}{1} * (max(\frac{10}{40}, \frac{40}{40})) \\
&= 1
\end{aligned}
$$

However, their metric is equivalent to ours as long as (1) there are no alternatives in the gold standard, (2) neither the gold standard nor the prediction contains overlapping spans with the same label and (3) each span from the gold standard overlaps with at most one span with the same label from the predictions and vice versa.

6.5 MAFALDA Dataset

6.5.1 Source Datasets

We used four publicly available fallacy datasets to construct our benchmark for fallacy detection and classification:

We imported all 8,576 texts from Sahai et al. [160]. These are online discussions from Reddit, as in Example (d) of Figure 6.2. We reconstructed the texts by concatenating the post of interest, the previous post (if available), and the title. The title was considered as a citation and was thus not annotated. This dataset contains sentences that were labeled as negative examples.

We imported all 336 texts from Martino et al. [118], which are from news outlets. We imported all 583 texts from Jin et al. [92], which are either toy examples gathered from online quizzes (as in Example (a)), or longer climate-related texts originating from news outlets. Finally, we imported all 250 texts from Goffredo et al. [60], which are American political debates (Example (b)). We split these longer texts into shorter texts by concatenating the previous and following sentences of allegedly fallacious texts.

This gives us an English-language corpus of 9,745 texts, which is diverse in terms of linguistic terms and text length. We removed URLs, emails, and phone numbers globally.

6.5.2 Annotation

The existing annotation schemes on our corpus varied a lot among papers: for example, only Sahai et al. [160] approached the annotation task as a binary classification, where annotators determine if a given text contains a specified type of fallacy. The annotations process also varied greatly w.r.t. how consensus was obtained, as explained in Section 6.2.2.

Therefore, we removed all annotations, and manually re-annotated, from scratch, 200 randomly selected texts from our merged corpus. Our sample mirrors the dis-

tribution of sources and the original labels in our corpus: it contains 124 texts from Sahai et al. [160], 59 texts from Jin et al. [92], and 17 political debate texts from [60]. We did not use the texts from Martino et al. [118] we initially planned to use because they were more than 5,000 characters long. Thus, annotating a single text would considerably bias the work towards Martino's. However, the texts are part of our cleaned and homogenized dataset, and our goal is to include the annotations of these texts as we enlarge our manual annotation.

LLMs were not involved in the annotation process. We did not involve crowd workers either, because 33%-46% of Mturk workers are estimated to use ChatGPT [180]. Hence, five annotators annotated the texts. The task was (*i*) identifying each argument in a text, (*ii*) determining whether it is fallacious, (*iii*) determining the span of the fallacy (as defined in Definition 6.4.2), and (*iv*) choosing the fallacy type(s). The annotators discussed each fallacious span together, and either converged on one annotation or permitted several alternative annotations for the same span. They provide a completed template for each annotation, as defined in Section 6.3.2, along with an explanation for each annotation.

The process took around 40 hours. This corresponds to an average of 12 minutes per example, ranging from less than a minute for toy examples to half an hour when disagreement raised a debate. The total number of person-hours was 130.

6.5.3 Annotation Edge Cases

Here, we discuss annotation edge cases that are useful for annotators. In our definition (as in Gensler [54]), a fallacy is always an argument in the sense of Definition 6.3.1, i.e., **a fallacy is always of the form "*A, B, C*, ... therefore *X*" or of the form "*X* because *A, B, C*, ...", or it can be rephrased into these forms**. Hence, false assertions are not, *per se*, fallacies. For example, "*Paris is the capital of England*" is a false claim. However, it is not a fallacy because it is not an argument. The same goes for generalizations: "*All Americans love Trump*" is false, but not a fallacy. An insult (such as "*You are too stupid*"), likewise, is not a fallacy[2]. Slogans (such as "*America first!*"), likewise, are not fallacies in our definition, even if other works classify them as propaganda [118]. An *appeal to emotion* (such as "*Think of the poor children!*"), likewise, is not a fallacy by itself. It becomes a fallacy only when used as the premise of a fallacious argument, as in: "*Think of the poor children, and [therefore] vote for me!*". But not every argument that appeals to emotion is automatically fallacious. For instance, the argument "*During a Covid-19 pandemic, you should wear a mask in public transport because otherwise you could get infected*" appeals to fear. However, it is still a valid argument because the premise does entail the conclusion. Even if the premises of an argument are factually false, the argument is not necessarily fallacious. For example, "*All Americans love Trump, and therefore Biden loves Trump*" is an argument that rests on a false premise – but it is not fallacious because the premise indeed entails the conclusion in the sense of [73]: if the premise were true, the conclusion would be

[2]It becomes a fallacy when it is used as the premise of an argument, as in "*You are stupid, therefore what you say can't be true.*"

true as well. **The fallaciousness of an argument is thus largely independent of the truth values of its components.**

Finally, **the description of fallacious reasoning is not automatically fallacious.** For example, "*You should wear a tin foil hat because it protects you against mind control*" is a fallacy (because the tin foil hat does not protect against mind control of any known form). However, the following is a factual assertion, not a fallacy: "*Some people wear tin foil hats because they are afraid of mind control*".

6.5.4 Annotation Guidelines for Identifying Fallacious Arguments

The task of annotating a text with fallacies can be decomposed into several steps: First, determine if the text contains an argument and what the premises and conclusion are. Then, the span must be delimited. Finally, an adequate label must be chosen. For the construction of our gold standard, annotators used Doccano[3], and followed these guidelines:

1. **Consensus Requirement:** Before finalizing annotations for any given text, annotators should try to reach a consensus. This collaborative approach ensures consistency and accuracy in the identification of fallacious arguments. In instances where consensus is unattainable, the differing viewpoints regarding potential fallacies should be noted as alternative interpretations, as detailed in Section 6.4.2.

2. **Resource Utilization:** Annotators are encouraged to consult various resources, including Google Search, Wikipedia, and books on argumentation. However, using Large Language Models, such as ChatGPT, is prohibited to prevent potential bias or contamination in the annotations.

3. **Reference Material:** For definitions and clarifications:

   - Refer to the definitions of an argument and a fallacy as outlined in Section 6.3.1 and Section 6.5.3.
   - Consult the Appendix B.1 for detailed formal and informal definitions of individual fallacies.
   - Follow the definition of spans detailed in Section 6.3.1

4. **Annotation Protocol:**

   - Upon reaching a consensus, annotators must document their rationale, aligning their reasoning with the formal definitions provided.
   - Annotators are encouraged to add useful comments. This includes identifying text segments that may require special post-processing or additional review.

Our annotation guidelines are also available on the Web page of our project, https://github.com/ChadiHelwe/MAFALDA/.

[3] https://github.com/doccano/doccano

6.5.5 Gold Standard Annotators

The gold standard was produced by five annotators, who have the following characteristics:

- Nationality: Lebanese, Gender: Male, Native language: Arabic, Education: Master's degree, Occupation: Ph.D. student in computer science.
- Nationality: French, Gender: Male, Native language: French, Education: Master's degree, Occupation: Ph.D. student in computer science.
- Nationality: French, Gender: Male, Native language: French, Education: Ph.D. degree, Occupation: Post-doctoral researcher in computer science.
- Nationality: French, Gender: Female, Native language: French, Education: Ph.D. degree, Occupation: Professor in computer science.
- Nationality: German, Gender: Male, Native language: German, Education: Ph.D. degree, Occupation: Professor in computer science.

6.5.6 Statistics

Our dataset comprises 9,745 texts, of which 200 texts have been annotated manually, with a total of 203 spans. Among these, 137 texts contained at least one span identified as fallacious, while the remaining texts did not contain any fallacious spans. The mean number of spans per text is 1.34.

Among the 200 texts, 71 were initially labeled as non-fallacious. However, our annotation found fallacies in some of these texts. This can be explained by the methodology from Sahai et al. [160], where crowd workers check only one specific type of fallacy. If that fallacy is not present, the text is annotated as non-fallacious. Our annotation, however, spotted other fallacies in the text, and labeled them. In the end, we have 63 non-fallacious texts. Table 6.2 showcases the source datasets of the MAFALDA examples.

Source Dataset	Non-annotated	Annotated
Sahai et al. [160]	640 (7,812)	71 (63)
Jin et al. [92]	524	59
Martino et al. [118]	336	0
Goffredo et al. [60]	233	17
TOTAL	1733 (7,812)	137 (63)

Table 6.2: Distribution of text from the initial source and from the final re-annotated dataset. Numbers in parenthesis are for non-fallacious texts.

The dataset contains all the fallacies presented in Section 6.3.2. The three most frequent fallacies represent 1/4 of the dataset, while the least frequent fallacies appear

less than three times (as shown in Table 6.3). 71.5% of the texts were annotated with a similar fallacy as the original one (at least one fallacy of the source annotation was in the new annotation, or we agreed on a non-fallacious text). The difference is mainly because our taxonomy introduced new fallacies, such as *appeal to ridicule*, and removed fallacies that we considered too vague or broad, such as *intentional fallacy*. In some cases, we used a different granularity than in the source: while the source might say *appeal to emotion*, we annotated, e.g., with *appeal to fear*. We also permit several alternative annotations per span, which entails that the new annotations have, on average, more annotations per text than the source annotations (Original: 0.665, Ours: 1.34).

The dataset contains 203 spans, of which 65 (i.e., 28%) contain at least two different (alternative) labels (as shown in Table 6.4). Example 6.2c shows an example of alternative labels. We computed the co-occurrence matrix of the fallacies (as shown in Figure 6.6). Most fallacies do not co-occur too frequently (less than 30% of the time) with another particular fallacy, which indicates that our definitions of fallacies are broadly orthogonal. However, there are two fallacies with high co-occurrence frequency: *appeal to pity* has a 100% co-occurrence with *strawman* and *appeal to worse problem*. However, this is because there is only one occurrence of *appeal to pity* in our dataset. The second one is *guilt by association*, which is 38% of the time associated with *abusive ad hominem*. This is explained by the fact that *guilt by association* and *abusive ad hominem* are two types of the *ad hominem* fallacy.

	annotations	sources
non-fallacious	63	71
hasty generalization	28	33
causal oversimplification	23	0
Appeal to Ridicule	20	0
false dilemma	18	7
ad hominem	16	8
nothing	14	0
ad populum	14	13
straw man	13	0
false causality	13	8
false analogy	12	0
slippery slope	11	6
appeal to fear	11	0
appeal to nature	11	10
circular reasoning	11	10
appeal to (false) authority	9	10
appeal to worse problems	8	8
guilt by association	8	0
equivocation	7	1
appeal to tradition	6	6
appeal to anger	6	0
appeal to positive emotion	3	0
tu quoque	3	0
fallacy of division	2	0
appeal to pity	1	0
fallacy of relevance *	0	2
intentional *	0	1
appeal to emotion *	0	10

* Fallacies not included in MAFALDA.

Table 6.3: Number of spans for each fallacy: this table presents the distribution of fallacies in our dataset, comparing MAFALDA annotations with source annotations.

Number of Spans	0	1	2	3	4	5	6
w/ counting alternatives	63	70	32	18	8	6	3
w/o counting alternatives	63	95	23	15	3	1	0

Table 6.4: Number of text with N spans. The first line considers alternatives, i.e., a disjunction of two labels for a span will count as two annotations. Conversely, in the second line, an alternative will count as one annotation. This allows for comparing the usage of alternatives in our annotations.

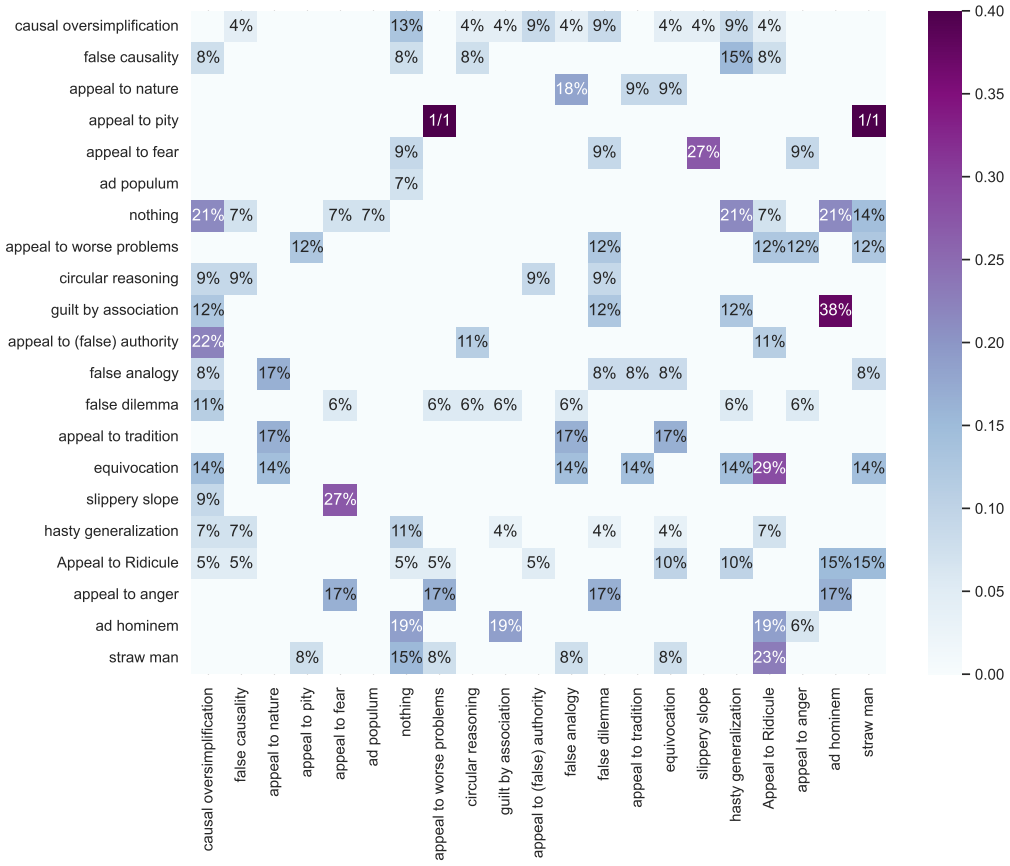


Figure 6.6: Co-occurrence of labels (frequency)

6.6 Experiments

We now evaluate the ability of state-of-the-art LMs (SLMs and LLMs) to detect the fallacies in our benchmark. Our benchmark is not intended for training or fine-tuning, and hence, we study a zero-shot setting with basic prompts. We are interested in the task of fallacy detection and classification of a given text, i.e., the input is a text, and the output is a list of annotated spans.

6.6.1 Settings

We study GPT-3.5 as well as 12 open-source models, covering different model sizes (Table 6.5). We use a bottom-up approach to evaluate our models starting at Level 2 granularity and extrapolate labels for Levels 1 and 0 based on these predictions, as our dataset includes three levels of granularity.

We employ a basic prompting approach that presents the model with our definition of a fallacy, the instruction to annotate the fallacies, the list of fallacies without their definitions, the corresponding text example, and the sentence to be labeled. The following is the detailed prompt used in our experiments:

Definitions:

- *An argument consists of an assertion called the conclusion and one or more assertions called premises, where the premises are intended to establish the truth of the conclusion. Premises or conclusions can be implicit in an argument.*
- *A fallacious argument is an argument where the premises do not entail the conclusion.*

Text: "{complete_example_input}"
Based on the above text, determine whether the following sentence is part of a fallacious argument or not. If it is, indicate the type(s) of fallacy without providing explanations. The potential types of fallacy include:

- *appeal to positive emotion*
- *...*
- *tu quoque*

Sentence: "{sentence_input}"
Output:

Our experiments are conducted at the sentence level; spans are formed by grouping consecutive sentences with the same label. A significant challenge with generative models is their inconsistent format output. Thus, we deem an output correct if it includes the name of the correct fallacy (or a part of it). Descriptions of the models can be found in Chapter 2.

6.6.2 LMs Results

Table 6.5 shows the results across different granularity levels in a zero-shot setting, as evaluated using our metric (see Section 6.4.2). We added *Baseline random*, a dummy model that predicts labels randomly following a uniform distribution.

	MAFALDA		
Model	F1 Level 0 *	F1 Level 1 *	F1 Level 2
Baseline random	0.435	0.061	0.010
Falcon 7B	0.397	0.130	0.022
LLAMA2 Chat 7B	0.572	0.114	0.068
LLAMA2 7B	0.492	0.148	0.038
Mistral Instruct 7B	0.536	0.144	0.069
Mistral 7B	0.450	0.127	0.044
Vicuna 7B	0.494	0.134	0.051
WizardLM 7B	0.490	0.087	0.036
Zephyr 7B	0.524	0.192	0.098
LLaMA2 Chat 13B	0.549	0.160	0.096
LLaMA2 13B	0.458	0.129	0.039
Vicuna 13B	0.557	0.173	0.100
WizardLM 13B	0.520	0.177	0.093
GPT 3.5 175B	**0.627**	**0.201**	**0.138**
Avg. Human on Sample	**0.749**	**0.352**	**0.186**

* Labels were extrapolated from Level 2.

Table 6.5: Performance results of different models across different granularity levels in a zero-shot setting. Avg. human on sample concerns only the 20 subsamples of MAFALDA for the user study. Metrics are explained in Section 6.4.2.

At all levels of granularity, all models surpass the performance of the baseline model (except for Falcon on Level 0), indicating that they are successfully identifying certain patterns or features. GPT 3.5 outperforms all other models at all levels. At Level 1, Zephyr 7B achieves comparable results to GPT 3.5, possibly thanks to the quality of its training dataset and/or engineering tricks, challenging the assumption that larger models are always more effective. More surprisingly, LLaMA2 performs better in its 7B version than in its 13B version for Levels 0 and 1. This phenomenon is in line with findings from Wei et al. [189]. For more in-depth analysis, The detailed results, including recall, precision, and F1 score for Levels 0, 1, and 2, are shown in Tables 6.6, 6.7, and 6.8, which shows that GPT-3.5 has better precision than all the evaluated SLMs at all Levels.

We also investigate whether it makes a difference to prompt the models directly on Level 1 (as opposed to extrapolating Level 1 from Level 2). For Mistral Instruct and

Zephyr, there is no significant difference: Mistral Instruct obtains an F1 score of 0.149, and Zephyr achieves an F1 score of 0.185.

Model	Precision Level 0	Recall Level 0	F1 Level 0
Falcon 7B	0.427	0.655	0.397
LLAMA2 Chat 7B	0.506	0.837	0.572
LLAMA2 7B	0.456	0.758	0.492
Mistral Instruct 7B	0.570	0.651	0.536
Mistral 7B	0.444	0.691	0.450
Vicuna 7B	0.529	0.628	0.494
WizardLM 7B	0.565	0.567	0.490
Zephyr 7B	0.489	0.765	0.524
LLaMA2 Chat 13B	0.493	0.793	0.549
LLaMA2 13B	0.433	0.739	0.458
Vicuna 13B	0.591	0.670	0.557
WizardLM 13B	0.523	0.756	0.520
GPT 3.5 175B	0.701	0.669	0.627

Table 6.6: Performance results for Level 0 on MAFALDA

Model	Precision Level 1	Recall Level 1	F1 Level 1
Falcon 7B	0.134	0.164	0.130
LLAMA2 Chat 7B	0.134	0.136	0.114
LLAMA2 7B	0.158	0.185	0.148
Mistral Instruct 7B	0.176	0.152	0.144
Mistral 7B	0.136	0.159	0.127
Vicuna 7B	0.161	0.146	0.134
WizardLM 7B	0.121	0.093	0.087
Zephyr 7B	0.207	0.230	0.192
LLaMA2 Chat 13B	0.173	0.183	0.160
LLaMA2 13B	0.140	0.151	0.129
Vicuna 13B	0.200	0.191	0.173
WizardLM 13B	0.193	0.205	0.177
GPT 3.5 175B	0.233	0.203	0.201

Table 6.7: Performance results for Level 1 on MAFALDA

Model	Precision Level 2	Recall Level 2	F1 Level 2
Falcon 7B	0.016	0.078	0.022
LLAMA2 Chat 7B	0.070	0.095	0.068
LLAMA2 7B	0.038	0.073	0.038
Mistral Instruct 7B	0.086	0.076	0.069
Mistral 7B	0.046	0.072	0.044
Vicuna 7B	0.062	0.067	0.051
WizardLM 7B	0.056	0.041	0.036
Zephyr 7B	0.090	0.145	0.098
LLaMA2 Chat 13B	0.101	0.122	0.096
LLaMA2 13B	0.037	0.068	0.039
Vicuna 13B	0.115	0.118	0.100
WizardLM 13B	0.088	0.134	0.093
GPT 3.5 175B	0.162	0.138	0.138

Table 6.8: Performance results for Level 2 on MAFALDA

6.7 User Study

We now evaluate humans' ability to detect and classify the fallacies in our benchmark sample of 20 randomly chosen examples.

6.7.1 User Study Annotators

The following 4 annotators provided the user study annotations:

- Nationality: Lebanese, Gender: Male, Native Language: Arabic, Education: Master's degree in mechanical engineering, Occupation: Statistics Expert.
- Nationality: French, Gender: Male, Native Language: French, Education: Master's degree in big data and data science, Occupation: Ph.D. Student in computer science.
- Nationality: Morrocan, Gender: Female, Native Language: French, Education: Ph.D. degree, Occupation: Data scientist.
- Nationality: French, Gender: Male, Native Language: French, Education: Master's degree in machine learning, Occupation: Ph.D. Student in computer science.

Compensation: The annotators were volunteers and were not compensated for the annotations.

6.7.2 Insights from the User Study Annotators

The annotation process was very time-consuming, with some annotators taking up to four hours to complete their tasks for the 20 examples. One annotator humorously

questioned their normality, stating, "*I don't know if I'm a normal human, but I found it difficult! :)*" while another jokingly expressed regret over accepting the task. These comments reflect the general sentiment about the task's complexity. The annotators often struggled with specific examples, such as *"Reasonable regulations don't lead to the fed keeping lists and someday coming after all gun owners to suppress the working class"*, which has been annotated differently by each user such as an *ad populum* and *false causality* fallacy while it is not a fallacy. This is often due to over-complicated sentences.

6.7.3 User Results

We measure human performance on our dataset (which constitutes, to our knowledge, the first such study in the fallacy classification literature). We aim to establish (*i*) whether humans outperform language models for the task at hand and (*ii*) whether humans agree more with our gold standard than among themselves. As human effort is more costly than running a language model (and even more so since we need engaged annotators who do not resort to ChatGPT or other LMs), we asked four other annotators to annotate 20 randomly chosen examples on the same task as the systems. On these 20 samples, we compared the results of human annotators and LMs. The low scores of human annotators reported in Table 6.10 show that the task is difficult. Still, human participants outperform the language models as shown in Table 6.5: Contrary to what previous work has demonstrated [58], GPT-3.5 does not perform better than humans. For more in-depth analysis, The detailed results, including recall, precision, and F1 score for Levels 0, 1, and 2, are shown in Table 6.9. The table reveals that User 2 has the highest precision at Levels 0 and 1, while User 1 has the highest precision at Level 2.

Next, we study whether they agree more with our gold standard than among themselves. We treat each annotator's work as a gold standard and assess the precision, recall, and F1 scores of the other annotators. Our gold standard achieves an F1 score of 0.186 on average for humans (see Table 6.10), outperforming the best alternative, which scores 0.144.

	Model	Precision	Recall	F1
Level 0	User 1	0.732	0.847	0.760
	User 2	0.785	0.892	0.821
	User 3	0.728	0.809	0.728
	User 4	0.704	0.767	0.694
	Average	0.737	0.829	0.749
Level 1	User 1	0.326	0.342	0.322
	User 2	0.399	0.402	0.397
	User 3	0.311	0.364	0.319
	User 4	0.375	0.394	0.371
	Average	0.353	0.376	0.352
Level 2	User 1	0.192	0.248	0.204
	User 2	0.162	0.172	0.164
	User 3	0.186	0.239	0.194
	User 4	0.170	0.211	0.180
	Average	0.177	0.217	0.186

Table 6.9: Performance results for the user study

Gold Standard	F1 Level 0 *	F1 Level 1 *	F1 Level 2
User 1	0.616	0.310	0.119
User 2	0.649	0.304	0.098
User 3	0.696	0.253	0.093
User 4	0.649	0.277	0.144
MAFALDA	**0.749**	**0.352**	**0.186**

* Labels were extrapolated from Level 2.

Table 6.10: Cross-comparison of user annotations and the gold standard. Each annotation of the user study has been alternatively used as a gold standard to demonstrate the superiority of our own gold standard.

6.8 Error Analysis

We conduct an error analysis on two models, GPT-3.5 and Falcon, which exhibit the best and worst performance on Level 2. Our analysis also includes the annotations of the user study. **Our first goal in this analysis is to compare whether the best model has better controlled behavior than the worst model when generating outputs**. The Falcon model identifies 625 fallacious spans, with an average of 4.8 fallacies per span, while the GPT-3.5 model detects only 199 fallacious spans, with an average of 1.07 fallacies per span. However, we have 203 fallacious spans in the gold standard. The distribution of fallacies for the fallacious span at Level 2 for each model is presented in Table 6.11. Based on our analysis, we have observed that the Falcon model tends to predict multiple fallacies that are irrelevant to a fallacious span. In contrast, the GPT-3.5 model displays a more controlled behavior, which explains why Falcon has a low precision score. It is also worth noting that GPT-3.5 never predicted a span as *tu quoque*. We observe that both models produce nonsensical outputs, such as SQL code like "*select name color order from tag where the name*," or incomplete classification of fallacies such as "*the sentence it's a mistake being considered as part of a fallacious argument.*". Falcon has 115 spans labeled as unknown, while GPT-3.5 has only 5. **Our second goal is to analyze the exact matching performance in detecting fallacies, as well as the types of fallacies that both models and humans struggle with at Level 1**. Out of the 625 fallacious spans identified by Falcon, only 60 match the gold standard exactly, while out of 199 fallacious spans detected by GPT-3.5, only 55 match the gold standard exactly. Both models struggle mainly with fallacies categorized as fallacies of emotion, as shown in Figure 6.7.
For the annotators of the user study, we use the sample of 20 randomly chosen examples with 24 spans. User 2 performs the best with 17 exactly matched spans, while User 4 performs worst with only 8 exactly matched spans. Based on the exact matched results, the analysis of Figure 6.9 reveals that all the annotators struggle mainly with the fallacies of appeal to emotion. This difficulty can be partly attributed to these fallacies being less prevalent in our sample compared to the other types of fallacies. Interestingly, Users 1 and 3 correctly predict more fallacies of logic. Conversely, Users 2 and 4 correctly predict more fallacies of credibility than the others. It is worth noting that none of the users used all 23 fallacies of the taxonomy during the annotations, as shown in Table 6.13.

In conclusion, models and humans tend to struggle more with fallacies of appeal to emotion, which could be expected since not every expression of emotion is necessarily a fallacy. The difficulty of the task lies in distinguishing valid arguments accompanied by emotions from fallacious arguments. This is supported by Figures 6.7 and 6.9. Despite the underrepresentation of the fallacies of the appeal to emotion in our user study sample, our findings indicate that humans often fail to exactly identify the specific fallacious spans classified under appeal to emotion fallacies. Moreover, even when humans correctly identify such fallacious spans, they are frequently misclassified, as shown in Table 6.10. In contrast, models tend more to find these fallacious spans although they, too, frequently misclassify them. The only instances where the models

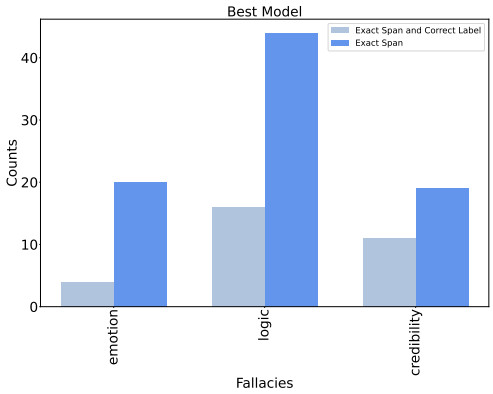
can correctly predict the fallacious spans and their labels are when they involve an *appeal to ridicule* or an *appeal to a worse problem*. These cases can be observed in Figures 6.8.

Fallacy Type	Best Model	Worst Model	Gold Standard
Appeal to Positive Emotion	3	128	3
Appeal to Anger	6	119	6
Appeal to Fear	5	132	11
Appeal to Pity	1	198	1
Appeal to Ridicule	10	121	20
Appeal to Worse Problems	21	157	8
Causal Oversimplification	6	81	23
Circular Reasoning	8	132	11
Equivocation	1	106	7
False Analogy	6	127	12
False Causality	9	57	13
False Dilemma	6	169	18
Hasty Generalization	41	123	28
Slippery Slope	10	77	11
Straw Man	6	135	13
Fallacy of Division	2	102	2
Ad Hominem	32	135	16
Ad Populum	4	75	14
Appeal to (False) Authority	10	211	9
Appeal to Nature	7	143	11
Appeal to Tradition	4	156	6
Guilt by Association	4	91	8
Tu Quoque	0	111	3
Unknown	5	115	-

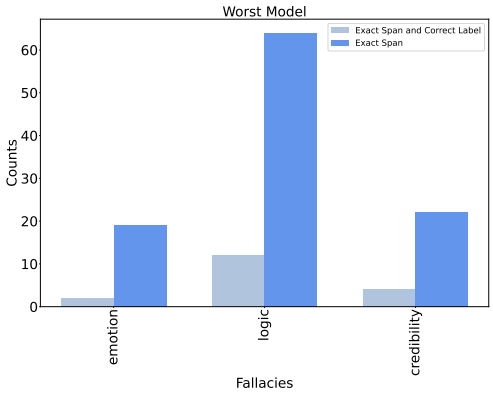
Table 6.11: Fallacy distribution at Level 2 of the Gold standard, Best model and Worst model

Fallacy Type	Best Model	Worst Model	Gold Standard
Emotion	46	855	49
Logic	95	1109	138
Credibility	61	922	67
Unknown	5	115	0

Table 6.12: Fallacy distribution at Level 1 of the Gold standard, Best model and Worst model

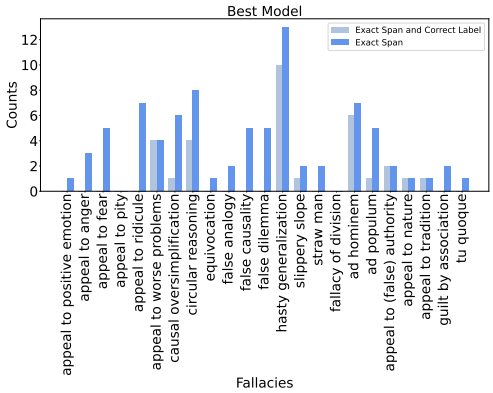


(a) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 1 for the **best model**

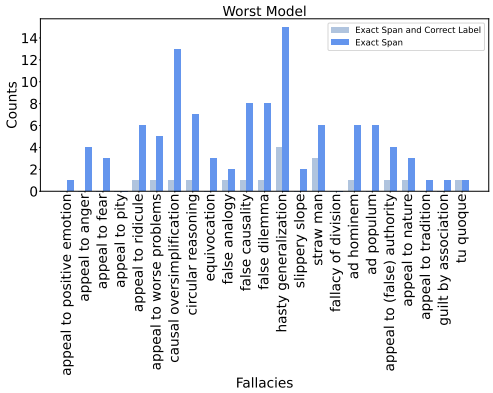


(b) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 1 for the **worst model**

Figure 6.7: Accuracy of fallacy labeling for spans that **exactly match** the gold standard at Level 1 for the best and worst models. *Exact Span* corresponds to the number of spans correctly identified by the model, *Exact Span and Correct Label* corresponds to the number of correctly labeled spans out of the correctly identified spans.



(a) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 2 for the **best model**

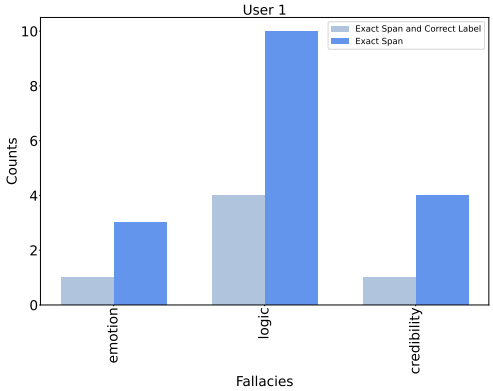


(b) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 2 for the **worst model**

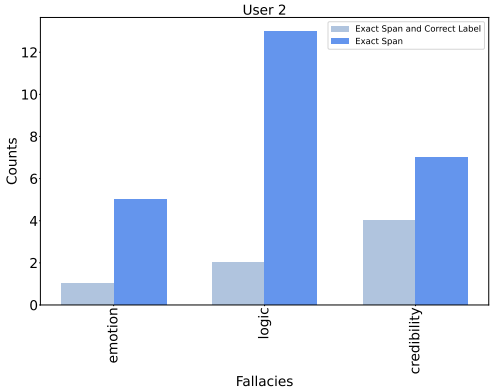
Figure 6.8: Accuracy of fallacy labeling for spans that **exactly match** the gold standard at Level 2 for the best and worst models. *Exact Span* corresponds to the number of spans correctly identified by the model, *Exact Span and Correct Label* corresponds to the number of correctly labeled spans out of the correctly identified spans.

Fallacy Type	User 1	User 2	User 3	User 4	Sample Gold Standard
Appeal to Positive Emotion	2	0	0	2	0
Appeal to Anger	1	0	0	0	0
Appeal to Fear	1	1	0	2	0
Appeal to Pity	0	0	0	0	0
Appeal to Ridicule	8	1	1	4	5
Appeal to Worse Problems	3	0	0	0	1
Causal Oversimplification	2	2	1	4	2
Circular Reasoning	2	0	2	0	1
Equivocation	1	0	0	5	1
False Analogy	1	1	1	0	0
False Causality	3	4	2	1	2
False Dilemma	1	1	0	1	2
Hasty Generalization	4	2	3	5	3
Slippery Slope	1	1	0	7	1
Straw Man	2	5	0	0	3
Fallacy of Division	3	0	0	0	0
Ad Hominem	4	1	3	2	4
Ad Populum	3	1	0	5	1
Appeal to (False) Authority	0	2	1	3	1
Appeal to Nature	0	1	0	0	0
Appeal to Tradition	1	1	1	2	2
Guilt by Association	1	3	1	1	1
Tu Quoque	0	1	2	0	0
Unknown	0	0	0	0	0

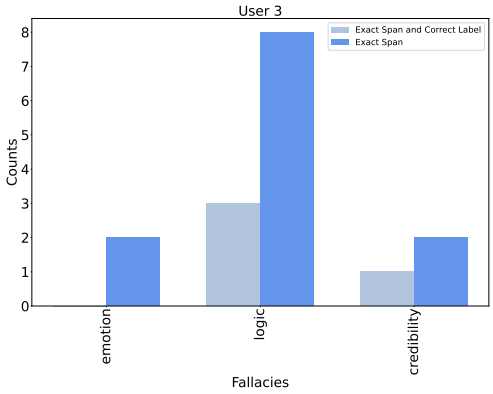
Table 6.13: Fallacies distribution at Level 2 of User 1, User 2, User 3, User 4, and the sample gold standard



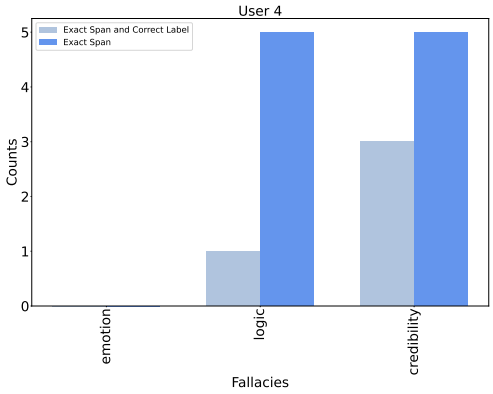
(a) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 1 for the **User 1**



(b) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 1 for the **User 2**

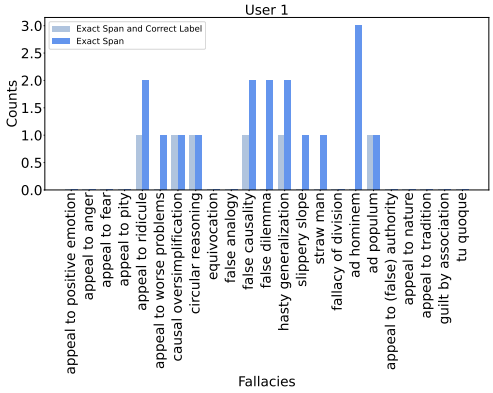


(c) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 1 for the **User 3**

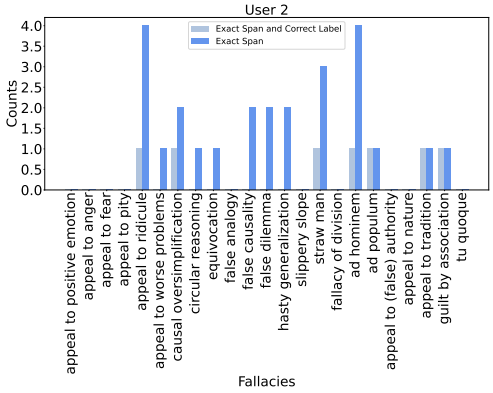


(d) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 1 for the **User 4**

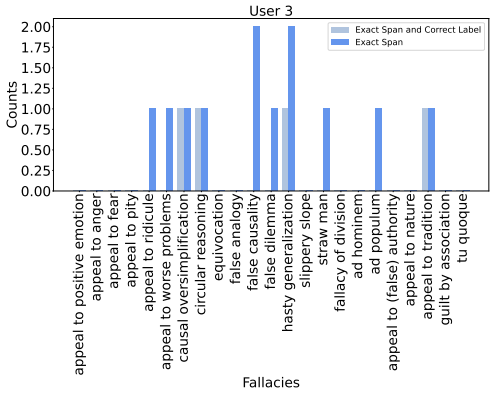
Figure 6.9: Accuracy of fallacy labeling for spans that **exactly match** the gold standard at Level 1 for the Users' annotations. *Exact Span* corresponds to the number of spans correctly identified by the user, *Exact Span and Correct Label* corresponds to the number of correctly labeled spans out of the correctly identified spans.



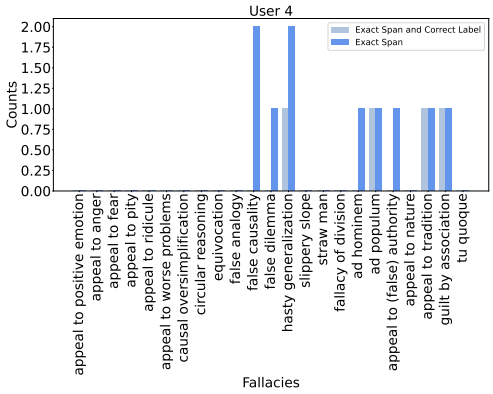
(a) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 2 for the **User 1**



(b) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 2 for the **User 2**



(c) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 2 for the **User 3**



(d) Accuracy of fallacy labeling for spans that exactly match the gold standard at Level 2 for the **User 4**

Figure 6.10: Accuracy of fallacy labeling for spans that **exactly match** the gold standard at Level 2 for the Users' annotations. *Exact Span* corresponds to the number of spans correctly identified by the user, *Exact Span and Correct Label* corresponds to the number of correctly labeled spans out of the correctly identified spans.

6.9 Conclusion

We have presented MAFALDA, a unified dataset designed for fallacy detection and classification. This dataset integrates four pre-existing datasets into a cohesive whole, achieved through developing a new, comprehensive taxonomy. This taxonomy aligns publicly available taxonomies dedicated to fallacy detection. We manually annotated 200 texts from our dataset and provided an explanation in the form of a completed template for each of them. The disjunctive annotation scheme we proposed embraces the subjectivity of the task and allows for several alternative annotations for the same span. We have further demonstrated the capabilities of various LMs in zero-shot fallacy detection and classification at the span level. While Level 0 classification shows good results, Levels 1 and 2 are largely out of reach of LMs (SLMs and LLMs) in zero-shot settings. We hope that our benchmark will enable researchers to improve the results of this challenging task.

Future work includes expanding into few-shot settings and exploring advanced prompting techniques, such as chain-of-thought, using the template-based definitions of fallacy and the taxonomy we provided. Furthermore, we believe that using a top-down approach, i.e., from Level 0 to Level 2 of our taxonomy, may provide better results than the bottom-up approach we used in our experiments. Regarding our disjunctive annotation scheme, we are interested in exploring its use in other NLP domains. Lastly, we plan to enrich the dataset with more annotated examples for model fine-tuning.

7 Conclusion

7.1 Summary

In this thesis, we have explored different reasoning capabilities of Language Models, how to evaluate them, and how we can improve them. Specifically, we have answered these research questions:

What types of reasoning can SLMs perform effectively? In Chapter 3, we have discussed the fundamental components required for an LM to perform reasoning. We have then explored various kinds of reasoning that SLMs can easily perform, such as Horn rule reasoning and simple commonsense reasoning. We have also examined the reasoning types that SLMs may find difficult to solve, such as Implicit Reasoning and Mathematical Proof Generation. Furthermore, we have highlighted the theoretical limitations of the Transformer architecture, as discovered by Hahn [66], which limits the Transformer's ability to model the Even Parity and Dyck-2 language. To validate these limitations, we have introduced two natural language tasks, namely the Light Switch Task and the Cake Task. These tasks highlight the concrete limitations of models based on the standard Transformer architecture in natural language reasoning. It is worth noting that even though this chapter primarily focuses on SLMs with millions of parameters, some of these limitations persist in SLMs with a few billions of parameters, such as Mistral, and LLMs, such as ChatGPT. In Chapter 4, we have presented LogiTorch, a Python library that enables reasoning on natural language. LogiTorch was built on top of the PyTorch framework, using the HuggingFace Transformers and PyTorch Lightning libraries. The library includes a vast collection of datasets that cover different reasoning tasks such as question answering, proof generation, and textual entailment. Among these datasets supported by LogiTorch, we have LogiQA, ConTROL, and ReCLOR. Also, the library includes various implemented models like PRover and BERTNOT. Additionally, there are several utility functions included in the library, such as coreference resolution and discourse delimitation, that can be utilized for feature engineering. The library is designed to simplify the use of reasoning datasets and enable the training of models with minimal coding. Furthermore, we have

evaluated the performance of our implemented models on several tasks and found that they achieved near-perfect accuracies comparable to the original implementations. We believe LogiTorch will simplify research, promote reusability, encourage evaluation, advance reproducibility, and foster open software and data for reasoning with LMs. In the future, more models and datasets will be included in the library.

How can we enhance the reasoning capabilities of SLMs? In Chapter 5, we have focused on improving the robustness of SLMs to negation, an essential component of reasoning. We have started by discussing the limitations of the current formal definition of textual entailment, which was used to evaluate models' reasoning, and have proposed a new probabilistic definition. This led us to develop TINA, a negated data augmentation technique that enhances the robustness of SLMs when negation is present in the premise, hypothesis, or both. Our experiments have shown that TINA, in combination with an unlikelihood loss function, effectively improves the robustness of SLMs to negation in textual entailment tasks. We have tested TINA with various models on different negated textual entailment datasets, including Negated SNLI and Negated MNLI. The results have shown that TINA significantly improves the performance of all the tested models on these datasets without affecting their performance on the non-negated textual entailment datasets, such as SNLI and MNLI.
TINA can be viewed as a Neuro-symbolic AI approach that combines neural networks and symbolic approaches. In TINA's Neuro-symbolic case, synthetic datasets are generated using data augmentation based on logic to train models.

How well can LLMs deal with logical fallacies? In Chapter 6, we have introduced MAFALDA, a benchmark for fallacy detection and classification. This benchmark requires a model to demonstrate a high level of reasoning to detect fallacious arguments. Additionally, we have proposed an annotation scheme and evaluation metric that considers subjectivity. The reason for incorporating subjectivity is that a single fallacious argument can have multiple types of fallacies, each of which can be defended. We have evaluated both LLMs, taking GPT-3.5 as our case study, and SLMs in a zero-shot setting on our benchmark using our metric, which have yielded preliminary results that can form the basis of future research. We also have evaluated human annotators on a benchmark sample and found they outperform LLMs and SLMs. In the future, we plan to evaluate few-shot settings and explore advanced prompting techniques, such as chain-of-thought, using the template-based definitions we have provided. Additionally, we intend to enhance the dataset with more annotated examples for model fine-tuning.

7.2 Future Work

In the context of assessing and improving the reasoning capabilities of Language Models, several promising research directions merit further investigation. These research directions include:

7.2.1 Neuro-Symbolic AI

In this thesis, we have proposed TINA, which can be viewed as a Neuro-symbolic AI technique to enhance a specific aspect of reasoning within LMs, namely, the understanding of negation. While various Neuro-symbolic approaches can enhance the overall reasoning capabilities of LMs, Henry Kautz proposed a taxonomy of neuro-symbolic architectures [164] that can be useful to explore their potential effectiveness in addressing different facets of reasoning. Despite the recent impressive performance of LLMs across diverse reasoning tasks, LLMs still encounter challenges such as hallucinations. Integrating symbolic approaches could potentially mitigate such issues. Moreover, incorporating symbolic AI can serve as a means to validate LLMs outputs. For instance, translating outputs into a logical form allows for verification by a prover, checking the reliability of LLMs-generated content.

7.2.2 Evaluating and Improving Reasoning of Low-resource LLMs

LLMs are predominantly trained and evaluated in English, primarily due to the abundant resources available in this language. Consequently, the reasoning capabilities of English LLMs are currently extensively studied. However, LLMs in languages such as Arabic, Portuguese, and Turkish have not been similarly explored due to these languages being considered less resourced. Consequently, few datasets are available for reasoning tasks in these languages, highlighting the urgent need to develop such resources. For example, a valuable future endeavor could be to develop a new version of the MAFALDA dataset that integrates multiple languages. This would establish the first multi-lingual fallacy detection and classification dataset.

7.2.3 Data Contamination and Trustworthiness of Reasoning Evaluation in Closed-Source LMs

Closed-source LMs such as ChatGPT do not allow for a comprehensive understanding of their pre-training data. This lack of transparency is also seen in some open-source LMs, which do not consistently reveal their pre-training corpus details. This lack of clarity presents a significant challenge in evaluating these models on standardized reasoning benchmarks, primarily because there is a risk that these benchmarks may have been unintentionally included in the pre-training data. Such accidental inclusion, known as "data contamination," makes it difficult to distinguish whether the model is genuinely reasoning or simply repeating memorized content from its dataset. To address this issue, researchers must continually develop new reasoning benchmarks while safeguarding against their leakage. Additionally, effective methods for detecting data contamination and techniques for unlearning LMs on contaminated datasets without compromising performance on other tasks are crucial areas for investigation.

At the end, our work in this thesis addresses critical challenges related to reasoning with LMs. While it is important to keep pushing the state-of-the-art models to achieve

improvements in reasoning benchmarks, we encourage researchers who read this thesis to focus on more foundational problems in NLP to achieve significant advancements in the field.

Beyond developing LMs that can understand, it is crucial to tackle the surrounding research problems of LMs that impact society at large. One such area is AI safety, specifically regarding the spread of misinformation. It is vital to develop methods to detect fake news or manipulated content. Additionally, AI ethics are essential. This includes ensuring LMs do not reinforce biases from their training data, respecting user privacy, and being developed responsibly. Another important aspect is the interpretability of LMs. As AI becomes more integrated into decision-making, it is important that users can trust and understand how AI comes to its conclusions, especially in critical applications like healthcare and justice. Finally, the environmental impact of developing and using LMs should be considered. Researchers should aim to create approaches that are more energy-efficient to minimize the ecological footprint of AI systems. Therefore, future research should also prioritize these dimensions, exploring innovative approaches to create robust, ethical, trustworthy, and eco-friendly AI systems.

Bibliography

[1] Malak Abdullah, Ola Altiti, and Rasha Obiedat. Detecting Propaganda Techniques in English News Articles using Pre-trained Transformers. In *2022 13th International Conference on Information and Communication Systems (ICICS)*, pages 301–308. IEEE, 2022.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] Baleegh Ahmad, Shailja Thakur, Benjamin Tan, Ramesh Karri, and Hammond Pearce. Fixing hardware security bugs with large language models. *arXiv preprint arXiv:2302.01215*, 2023.

[4] Hani Al-Omari, Malak Abdullah, Ola AlTiti, and Samira Shaikh. JUSTDeep at NLP4IF 2019 task 1: Propaganda detection using ensemble deep learning models. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 113–118. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-5016. URL https://aclanthology.org/D19-5016.

[5] Tariq Alhindi, Tuhin Chakrabarty, Elena Musi, and Smaranda Muresan. Multitask instruction-based prompting for fallacy recognition. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8172–8187, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.560.

[6] Saud Althabiti, Mohammad Ammar Alsalka, and Eric Atwell. Generative ai for explainable automated fact checking on the factex: A new benchmark dataset. In *Multidisciplinary International Symposium on Disinformation in Open Online Media*, pages 1–13. Springer, 2023.

[7] Amit Arora, Anshu Arora, and John McIntyre. Developing chatbots for cyber security: Assessing threats through sentiment analysis on social media. *Sustainability*, 15(17):13178, 2023.

[8] Isabelle Augenstein, Timothy Baldwin, Meeyoung Cha, Tanmoy Chakraborty, Giovanni Luca Ciampaglia, David Corney, Renee DiResta, Emilio Ferrara, Scott Hale, Alon Halevy, et al. Factuality challenges in the era of large language models. *arXiv preprint arXiv:2310.05189*, 2023.

[9] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[10] Oana Balalau and Roxana Horincar. From the Stage to the Audience: Propaganda on Reddit. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3540–3550. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.eacl-main.309. URL https://aclanthology.org/2021.eacl-main.309.

[11] Kshitij Bansal, Sarah Loos, Markus Rabe, Christian Szegedy, and Stewart Wilcox. Holist: An environment for machine learning of higher order logic theorem proving. In *International Conference on Machine Learning*, 2019.

[12] Qiming Bao. Pararule plus: A larger deep multi-step reasoning dataset over natural language. 2021.

[13] Emily M Bender and Alexander Koller. Climbing towards nlu: On meaning, form, and understanding in the age of data. In *Annual Meeting of the Association for Computational Linguistics*, 2020.

[14] Bo Bennett. *Logically Fallacious: The Ultimate Collection of over 300 Logical Fallacies (Academic Edition)*. EBookIt. com, 2012.

[15] Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. The fifth pascal recognizing textual entailment challenge. In *TAC*, 2009.

[16] Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. The reversal curse: Llms trained on" a is b" fail to learn" b is a". *arXiv preprint arXiv:2309.12288*, 2023.

[17] Dor Bernsohn, Gil Semo, Yaron Vazana, Gila Hayat, Ben Hagag, Joel Niklaus, Rohit Saha, and Kyryl Truskovskyi. Legallens: Leveraging llms for legal violation identification in unstructured text. *arXiv preprint arXiv:2402.04335*, 2024.

[18] Gregor Betz, Christian Voigt, and Kyle Richardson. Critical thinking for language models. In *Proceedings of the 14th International Conference on Computational Semantics (IWCS)*, pages 63–75, 2021.

[19] Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. On the ability of self-attention networks to recognize counter languages. In *Conference on Empirical Methods in Natural Language Processing*, 2020.

[20] Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. On the practical ability of recurrent neural networks to recognize hierarchical languages. In *International Conference on Computational Linguistics*, 2020.

[21] Yonatan Bisk, Rowan Zellers, Ronan LeBras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *AAAI Conference on Artificial Intelligence*, 2020.

[22] Michael Boratko, Xiang Li, Tim O'Gorman, Rajarshi Das, Dan Le, and Andrew McCallum. Protoqa: A question answering dataset for prototypical common-sense reasoning. In *Conference on Empirical Methods in Natural Language Processing*, 2020.

[23] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. Comet: Commonsense transformers for automatic knowledge graph construction. In *Annual Meeting of the Association for Computational Linguistics*, 2019.

[24] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. In *Conference on Empirical Methods in Natural Language Processing*, 2015.

[25] Nick Braisby and Angus Gellatly. *Cognitive Psychology*. Oxford University Press, 2005.

[26] Britannica. *Fallacy*. Encyclopaedia Britannica, Inc, 2023.

[27] Nathanael Chambers, Daniel Cer, Trond Grenager, David Hall, Chloe Kiddon, Bill MacCartney, Marie-Catherine De Marneffe, Daniel Ramage, Eric Yeh, and Christopher D Manning. Learning alignments and leveraging natural logic. In *ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, 2007.

[28] Michael Chen, Mike D'Arcy, Alisa Liu, Jared Fernandez, and Doug Downey. Codah: An adversarially-authored question answering dataset for common sense. In *Workshop on Evaluating Vector Space Representations for NLP*, 2019.

[29] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

[30] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing*, 2014.

[31] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

[32] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

[33] Peter Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3882–3890, 2021.

[34] Irving M Copi, Carl Cohen, and Victor Rodych. *Introduction to logic*. Routledge, 2018.

[35] Jiaxi Cui, Zongjian Li, Yang Yan, Bohua Chen, and Li Yuan. Chatlaw: Open-source legal large language model with integrated external knowledge bases. *arXiv preprint arXiv:2306.16092*, 2023.

[36] Leyang Cui, Sijie Cheng, Yu Wu, and Yue Zhang. Does bert solve commonsense task via commonsense knowledge? *arXiv preprint arXiv:2008.03945*, 2020.

[37] Gary N. Curtis. Logical Fallacies: The Fallacy Files, 2003. URL https://www.fallacyfiles.org/index.html.

[38] Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, 2005.

[39] Victor Danciu et al. Manipulative marketing: persuasion and manipulation of the consumer through advertising. *Theoretical and Applied Economics*, 21(2):591, 2014.

[40] Kahneman Daniel. *Thinking, fast and slow*. 2017.

[41] Joe Davison, Joshua Feldman, and Alexander Rush. Commonsense knowledge mining from pretrained models. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, 2019.

[42] Pieter Delobelle, Murilo Cunha, Eric Massip Cano, Jeroen Peperkamp, and Bettina Berendt. Computational Ad Hominem Detection. In *Proceedings of the 57th*

Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, pages 203–209. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-2028. URL https://aclanthology.org/P19-2028.

[43] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

[44] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

[45] Bradley Dowden. Fallacies — Internet Encyclopedia of Philosophy, 2010. URL https://iep.utm.edu/fallacy/.

[46] Stephen Downes. Stephen's Guide to the Logical Fallacies, 2003. URL http://people.uncw.edu/kozloffm/logicalfallacies.html.

[47] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357, 1995. ISSN 0004-3702. doi: 10.1016/0004-3702(94)00041-X. URL http://www.sciencedirect.com/science/article/pii/000437029400041X.

[48] Shadia Abdelhameed Elsayed, Osama Abu-Hammad, Albraa B Alolayan, Yasmin Salah Eldeen, and Najla Dar-Odeh. Fallacies and facts around covid-19: the multifaceted infection. *The Journal of craniofacial surgery*, 2020.

[49] Allyson Ettinger. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48, 2020.

[50] Logical Fallacies. Logical Fallacies - List of Logical Fallacies with Examples, 2008. URL https://www.logicalfallacies.org/.

[51] Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for nlp. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP*, 2021.

[52] Maxwell Forbes, Ari Holtzman, and Yejin Choi. Do neural language representations learn physical commonsense? *arXiv preprint arXiv:1908.02899*, 2019.

[53] Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. Specializing smaller language models towards multi-step reasoning. In *International Conference on Machine Learning*, pages 10421–10430. PMLR, 2023.

[54] Harry J. Gensler. *The A to Z of Logic*. The A to Z Guide Series. Scarecrow Press, 2010. ISBN 0810875969,9780810875968.

[55] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 2021.

[56] Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. The third pascal recognizing textual entailment challenge. In *ACL-PASCAL workshop on textual entailment and paraphrasing*, 2007.

[57] Danilo Giampiccolo, Hoa Trang Dang, Bernardo Magnini, Ido Dagan, Elena Cabrio, and Bill Dolan. The fourth pascal recognizing textual entailment challenge. In *TAC*, 2008.

[58] Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. ChatGPT Outperforms Crowd-Workers for Text-Annotation Tasks, 2023. URL http://arxiv.org/abs/2303.15056.

[59] Oren Glickman, Ido Dagan, and Moshe Koppel. Web based probabilistic textual entailment. In *Proceedings of the 1st Pascal Challenge Workshop*, pages 33–36, 2005.

[60] Pierpaolo Goffredo, Shohreh Haddadan, Vorakit Vorakitphan, Elena Cabrio, and Serena Villata. Fallacious Argument Classification in Political Debates. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, pages 4143–4149. International Joint Conferences on Artificial Intelligence Organization, 2022. ISBN 978-1-956792-00-3. doi: 10.24963/ijcai.2022/575. URL https://www.ijcai.org/proceedings/2022/575.

[61] Nicolas Gontier, Koustuv Sinha, Siva Reddy, and Chris Pal. Measuring systematic generalization in neural proof generation with transformers. *Advances in Neural Information Processing Systems*, 2020.

[62] Ashim Gupta, Giorgi Kvernadze, and Vivek Srikumar. Bert & family eat word salad: Experiments with text understanding. In *AAAI Conference on Artificial Intelligence*, 2021.

[63] Ivan Habernal, Raffael Hannemann, Christian Pollak, Christopher Klamm, Patrick Pauli, and Iryna Gurevych. Argotario: Computational argumentation meets serious games. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 7–12, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-2002. URL https://aclanthology.org/D17-2002.

[64] Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. The argument reasoning comprehension task: Identification and reconstruction of

implicit warrants. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2018.

[65] Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. Before Name-Calling: Dynamics and Triggers of Ad Hominem Fallacies in Web Argumentation. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 386–396. Association for Computational Linguistics, 2018. doi: 10.18653/v1/n18-1036.

[66] Michael Hahn. Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*, 2020.

[67] R Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. The second pascal recognising textual entailment challenge. In *The Second PASCAL Challenges Workshop on Recognising Textual Entailment*, 2006.

[68] Hans Hansen. Fallacies. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Summer 2020 edition, 2020.

[69] Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *IEEE International Conference on Computer Vision*, 2017.

[70] Chadi Helwe, Chloé Clavel, and Fabian M Suchanek. Reasoning with transformer-based models: Deep learning, but shallow reasoning. In *3rd Conference on Automated Knowledge Base Construction*, 2021.

[71] Chadi Helwe, Chloé Clavel, and Fabian Suchanek. Logitorch: A pytorch-based library for logical reasoning on natural language. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 250–257, 2022.

[72] Chadi Helwe, Simon Coumes, Chloé Clavel, and Fabian Suchanek. Tina: Textual inference with negation augmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4086–4099, 2022.

[73] Chadi Helwe, Simon Coumes, Chloé Clavel, and Fabian M. Suchanek. TINA: Textual Inference with Negation Augmentation. In *EMNLP Find.*, 2022.

[74] Chadi Helwe, Tom Calamai, Pierre-Henri Paris, Chloé Clavel, and Fabian Suchanek. Mafalda: A benchmark and comprehensive study of fallacy detection and classification. *arXiv preprint arXiv:2311.09761*, 2023.

[75] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[76] Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*, 2020.

[77] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[78] Emma Hoes, Sacha Altay, and Juan Bermeo. Leveraging chatgpt for efficient fact-checking. *PsyArXiv. April*, 3, 2023.

[79] Vincent Homer, Lisa Matthewson, Cécile Meier, Hotze Rullman, and Thomas Ede Zimmermann. Negative polarity. *Blackwell companion to semantics, Wiley (forthcoming)*, 2019.

[80] Ruixin Hong, Hongming Zhang, Xinyu Pang, Dong Yu, and Changshui Zhang. A closer look at the self-verification abilities of large language models in logical reasoning. *arXiv preprint arXiv:2311.07954*, 2023.

[81] Laurence R Horn. A natural history of negation. 1989.

[82] Md Mosharaf Hossain, Venelin Kovatchev, Pranoy Dutta, Tiffany Kao, Elizabeth Wei, and Eduardo Blanco. An analysis of natural language inference benchmarks through the lens of negation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9106–9118, 2020.

[83] Md Mosharaf Hossain, Dhivya Chinnappa, and Eduardo Blanco. An analysis of negation in natural language understanding corpora. In *Annual Meeting of the Association for Computational Linguistics*, 2022.

[84] Arian Hosseini, Siva Reddy, Dzmitry Bahdanau, R Devon Hjelm, Alessandro Sordoni, and Aaron Courville. Understanding by understanding not: Modeling negation in language models. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2021.

[85] Beizhe Hu, Qiang Sheng, Juan Cao, Yuhui Shi, Yang Li, Danding Wang, and Peng Qi. Bad actor, good advisor: Exploring the role of large language models in fake news detection. *arXiv preprint arXiv:2309.12247*, 2023.

[86] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[87] Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, 2019.

[88] Shanshan Huang and Kenny Zhu. Statistically profiling biases in natural language reasoning datasets and models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4521–4530, 2023.

[89] Yinya Huang, Meng Fang, Yu Cao, Liwei Wang, and Xiaodan Liang. Dagn: Discourse-aware graph network for logical reasoning. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2021.

[90] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[91] Liming Jiang. Detecting scams using large language models. *arXiv preprint arXiv:2402.03147*, 2024.

[92] Zhijing Jin, Abhinav Lalwani, Tejas Vaidhya, Xiaoyu Shen, Yiwen Ding, Zhiheng Lyu, Mrinmaya Sachan, Rada Mihalcea, and Bernhard Schölkopf. Logical Fallacy Detection, 2022. URL http://arxiv.org/abs/2202.13758.

[93] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020.

[94] Waleed Kareem and Noorhan Abbas. Fighting lies with intelligence: Using large language models and chain of thoughts technique to combat fake news. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 253–258. Springer, 2023.

[95] Nora Kassner and Hinrich Schütze. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Annual Meeting of the Association for Computational Linguistics*, 2020.

[96] Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. Unifiedqa: Crossing format boundaries with a single qa system. In *Conference on Empirical Methods in Natural Language Processing*, 2020.

[97] Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. Mawps: A math word problem repository. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016.

[98] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, 2017.

[99] Jinqi Lai, Wensheng Gan, Jiayang Wu, Zhenlian Qi, and Philip S Yu. Large language models in law: A survey. *arXiv preprint arXiv:2312.03718*, 2023.

[100] Guillaume Lample and François Charton. Deep learning for symbolic mathematics. In *International Conference on Learning Representations*, 2019.

[101] João A Leite, Olesya Razuvayevskaya, Kalina Bontcheva, and Carolina Scarton. Detecting misinformation with llm-predicted credibility signals and weak supervision. *arXiv preprint arXiv:2309.07601*, 2023.

[102] Hector Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *Conference on the Principles of Knowledge Representation and Reasoning*, 2012.

[103] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Annual Meeting of the Association for Computational Linguistics*, 2019.

[104] Wenda Li, Lei Yu, Yuhuai Wu, and Lawrence Paulson. Isarstep: a benchmark for high-level mathematical reasoning. In *International Conference on Learning Representations*, 2021.

[105] Xiao Li, Gong Cheng, Ziheng Chen, Yawei Sun, and Yuzhong Qu. Adalogn: Adaptive logic graph network for reasoning-based machine reading comprehension. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7147–7161, 2022.

[106] Tatiana Likhomanenko, Qiantong Xu, Gabriel Synnaeve, Ronan Collobert, and Alex Rogozhnikov. Cape: Encoding relative positions with continuous augmented positional embeddings. *Advances in Neural Information Processing Systems*, 34: 16079–16092, 2021.

[107] Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. Birds have four legs?! numersense: Probing numerical commonsense knowledge of pre-trained language models. In *Conference on Empirical Methods in Natural Language Processing*, 2020.

[108] Jieyu Lin, Jiajie Zou, and Nai Ding. Using adversarial attacks to reveal the statistical bias in machine reading comprehension models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the*

11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 333–342, 2021.

[109] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, 2022.

[110] Hanmeng Liu, Leyang Cui, Jian Liu, and Yue Zhang. Natural language inference in context–investigating contextual reasoning over long texts. In *AAAI Conference on Artificial Intelligence*, 2020.

[111] Hanmeng Liu, Ruoxi Ning, Zhiyang Teng, Jian Liu, Qiji Zhou, and Yue Zhang. Evaluating the logical reasoning ability of chatgpt and gpt-4. *arXiv preprint arXiv:2304.03439*, 2023.

[112] Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. Logiqa: a challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3622–3628, 2021.

[113] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[114] Sebastian Löbner. Polarity in natural language: Predication, quantification and negation in particular and characterizing sentences. *Linguistics and Philosophy*, 2000.

[115] Nicholas Lourie, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Unicorn on rainbow: A universal commonsense reasoning model on a new multitask benchmark. In *AAAI Conference on Artificial Intelligence*, 2021.

[116] Fabrizio Macagno. Argumentation profiles and the manipulation of common ground. the arguments of populist leaders on twitter. *Journal of Pragmatics*, 191: 67–82, 2022.

[117] Robin Manhaeve, Sebastijan Dumančić, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Deepproblog: Neural probabilistic logic programming. *Advances in Neural Information Processing Systems*, 2018.

[118] Giovanni Da San Martino, Seunghak Yu, Alberto Barrón-Cedeño, Rostislav Petrov, and Preslav Nakov. Fine-Grained Analysis of Propaganda in News Article. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing,*

EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 5635–5645. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1565.

[119] John McCarthy. Applications of circumscription to formalizing common-sense knowledge. *Artificial intelligence*, 28(1):89–116, 1986.

[120] R Thomas McCoy, Junghyun Min, and Tal Linzen. Berts of a feather do not generalize together: Large variability in generalization across models with similar test set performance. In *Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, 2019.

[121] Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Annual Meeting of the Association for Computational Linguistics*, 2019.

[122] William Merrill, Yoav Goldberg, Roy Schwartz, and Noah A Smith. On the power of saturated transformers: A view from circuit complexity. *arXiv preprint arXiv:2106.16213*, 2021.

[123] Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing english math word problem solvers. In *Annual Meeting of the Association for Computational Linguistics*, 2020.

[124] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[125] Pasquale Minervini, Sebastian Riedel, Pontus Stenetorp, Edward Grefenstette, and Tim Rocktäschel. Learning reasoning strategies in end-to-end differentiable proving. In *International Conference on Machine Learning*, 2020.

[126] Shujaat Mirza, Bruno Coelho, Yuyuan Cui, Christina Pöpper, and Damon McCoy. Global-liar: Factuality of llms over time and geographic regions. *arXiv preprint arXiv:2401.17839*, 2024.

[127] Kanishka Misra, Allyson Ettinger, and Julia Rayz. Exploring bert's sensitivity to lexical cues using tests from semantic priming. In *Conference on Empirical Methods in Natural Language Processing*, 2020.

[128] Elena Musi, Myrto Aloumpi, Elinor Carmi, Simeon Yates, and O'Halloran Kay. Developing fake news immunity: Fallacies as misinformation triggers during the pandemic. In *Online Journal of Communication and Media Technologies*, pages 12(3), e202217, 2022. doi: https://doi.org/10.30935/ojcmt/12083.

[129] Ha-Thanh Nguyen. A brief report on lawgpt 1.0: A virtual legal assistant based on gpt-3. *arXiv preprint arXiv:2302.05729*, 2023.

[130] Qiang Ning, Hao Wu, Rujun Han, Nanyun Peng, Matt Gardner, and Dan Roth. Torque: A reading comprehension dataset of temporal ordering questions. In *Conference on Empirical Methods in Natural Language Processing*, 2020.

[131] Timothy Niven and Hung-Yu Kao. Probing neural network comprehension of natural language arguments. In *Annual Meeting of the Association for Computational Linguistics*, 2019.

[132] Hiroshi Noji and Hiroya Takamura. An analysis of the utility of explicit negative examples to improve the syntactic abilities of neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3375–3385, 2020.

[133] Yu Nong, Mohammed Aldeen, Long Cheng, Hongxin Hu, Feng Chen, and Haipeng Cai. Chain-of-thought prompting of large language models for discovering and fixing software vulnerabilities. *arXiv preprint arXiv:2402.17230*, 2024.

[134] Marc Gallofré Ocaña and Andreas L Opdahl. A software reference architecture for journalistic knowledge platforms. *Knowledge-Based Systems*, 276:110750, 2023.

[135] Siru Ouyang, Zhuosheng Zhang, and Hai Zhao. Fact-driven logical reasoning for machine reading comprehension. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18851–18859, 2024.

[136] Magdalini Paschali, Walter Simson, Abhijit Guha Roy, Muhammad Ferjad Naeem, Rüdiger Göbl, Christian Wachinger, and Nassir Navab. Data augmentation with manifold exploring geometric transformations for increased performance and robustness. *arXiv preprint arXiv:1901.04420*, 2019.

[137] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[138] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems? In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2021.

[139] Amirreza Payandeh, Dan Pluth, Jordan Hosier, Xuesu Xiao, and Vijay K Gurbani. How susceptible are llms to logical fallacies? *arXiv preprint arXiv:2308.09853*, 2023.

[140] Hammond Pearce, Benjamin Tan, Baleegh Ahmad, Ramesh Karri, and Brendan Dolan-Gavitt. Examining zero-shot vulnerability repair with large language models. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 2339–2356. IEEE, 2023.

[141] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*, 2023.

[142] Doris Penka. Negation and polarity. In *The Routledge handbook of semantics*. 2015.

[143] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, 2019.

[144] Thang Pham, Trung Bui, Long Mai, and Anh Nguyen. Out of order: How important is the sequential order of words in a sentence in natural language understanding tasks? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1145–1160, 2021.

[145] Xinyu Pi, Wanjun Zhong, Yan Gao, Nan Duan, and Jian-Guang Lou. Logigan: Learning logical reasoning via adversarial pre-training. *Advances in Neural Information Processing Systems*, 35:16290–16304, 2022.

[146] Barbara Plank. The "problem" of human label variation: On ground truth in data, modeling and evaluation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10671–10682, 2022.

[147] Adam Poliak. A survey on recognizing textual entailment as an NLP evaluation. In *First Workshop on Evaluation and Comparison of NLP Systems*, 2020.

[148] Stanislas Polu and Ilya Sutskever. Generative language modeling for automated theorem proving. *arXiv preprint arXiv:2009.03393*, 2020.

[149] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, 2020.

[150] Lianhui Qin, Aditya Gupta, Shyam Upadhyay, Luheng He, Yejin Choi, and Manaal Faruqui. Timedial: Temporal commonsense reasoning in dialog. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7066–7076, 2021.

[151] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 2020.

[152] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[153] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits

of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 2020.

[154] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 2020.

[155] Leonardo Ranaldi and André Freitas. Aligning large and small language models via chain-of-thought reasoning. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1812–1827, 2024.

[156] Paul Reisert, Benjamin Heinzerling, Naoya Inoue, Shun Kiyono, and Kentaro Inui. Riposte! a large corpus of counter-arguments. *arXiv preprint arXiv:1910.03246*, 2019.

[157] Anna Rogers, Olga Kovaleva, Matthew Downey, and Anna Rumshisky. Getting closer to ai complete question answering: A set of prerequisite real tasks. In *AAAI Conference on Artificial Intelligence*, 2020.

[158] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 2020.

[159] Swarnadeep Saha, Sayan Ghosh, Shashank Srivastava, and Mohit Bansal. Prover: Proof generation for interpretable reasoning over rules. In *Conference on Empirical Methods in Natural Language Processing*, 2020.

[160] Saumya Sahai, Oana Balalau, and Roxana Horincar. Breaking Down the Invisible Wall of Informal Fallacies in Online Discussions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 644–657. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-long.53. URL https://aclanthology.org/2021.acl-long.53.

[161] Gözde Gül Şahin and Mark Steedman. Data augmentation via dependency tree morphing for low-resource languages. In *Conference on Empirical Methods in Natural Language Processing*, 2018.

[162] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS*, 2019.

[163] Chinnadhurai Sankar, Sandeep Subramanian, Christopher Pal, Sarath Chandar, and Yoshua Bengio. Do neural dialog systems use the conversation history effectively? an empirical study. In *Annual Meeting of the Association for Computational Linguistics*, 2019.

[164] Md Kamruzzaman Sarker, Lu Zhou, Aaron Eberhart, and Pascal Hitzler. Neuro-symbolic artificial intelligence. *AI Communications*, 34(3):197–209, 2021.

[165] David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations*, 2019.

[166] Timo Schick and Hinrich Schütze. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, 2021.

[167] Viktor Schlegel, Marco Valentino, André Freitas, Goran Nenadic, and Riza Theresa Batista-Navarro. A framework for evaluation of machine reading comprehension gold standards. In *Language Resources and Evaluation Conference*, 2020.

[168] Eli Schwartz, Leonid Karlinsky, Joseph Shtok, Sivan Harary, Mattias Marder, Abhishek Kumar, Rogerio Feris, Raja Giryes, and Alex Bronstein. Delta-encoder: an effective sample synthesis method for few-shot object recognition. *Advances in Neural Information Processing Systems*, 2018.

[169] Emre Sezgin, Joseph Sirrianni, and Simon L Linwood. Operationalizing and implementing pretrained, large artificial intelligence linguistic models in the us health care system: outlook of generative pretrained transformer 3 (gpt-3) as a service model. *JMIR medical informatics*, 10(2):e32875, 2022.

[170] Samaneh Shafee, Alysson Bessani, and Pedro M Ferreira. Evaluation of llm chatbots for osint-based cyberthreat awareness. *arXiv preprint arXiv:2401.15127*, 2024.

[171] Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306, 2020.

[172] Shikhar Singh, Nuan Wen, Yu Hou, Pegah Alipoormolabashi, Te-Lin Wu, Xuezhe Ma, and Nanyun Peng. Com2sense: A commonsense reasoning benchmark with complementary sentences. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 883–898, 2021.

[173] Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L Hamilton. Clutrr: A diagnostic benchmark for inductive reasoning from text. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, 2019.

[174] Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. Proofwriter: Generating implications, proofs, and abductive statements over natural language. In *Findings*

of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 3621–3634, 2021.

[175] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

[176] Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. Leap-of-thought: Teaching pre-trained models to systematically reason over implicit knowledge. *Advances in Neural Information Processing Systems*, 33: 20227–20237, 2020.

[177] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[178] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*, 2023.

[179] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

[180] Veniamin Veselovsky, Manoel Horta Ribeiro, and Robert West. Artificial artificial artificial intelligence: Crowd workers widely use large language models for text production tasks. *arXiv preprint arXiv:2306.07899*, 2023.

[181] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2018.

[182] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in Neural Information Processing Systems*, 2019.

[183] Cunxiang Wang, Shuailong Liang, Yue Zhang, Xiaonan Li, and Tian Gao. Does it make sense? and why? a pilot study for sense making and explanation. In *Annual Meeting of the Association for Computational Linguistics*, 2019.

[184] Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. Entailment as few-shot learner. *arXiv preprint arXiv:2104.14690*, 2021.

[185] Siyuan Wang, Wanjun Zhong, Duyu Tang, Zhongyu Wei, Zhihao Fan, Daxin Jiang, Ming Zhou, and Nan Duan. Logic-driven context extension and data augmentation for logical reasoning of text. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1619–1629, 2022.

[186] Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 2019.

[187] Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 2019.

[188] Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 2019.

[189] Jason Wei, Najoung Kim, Yi Tay, and Quoc V. Le. Inverse scaling can become u-shaped. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 15580–15591. Association for Computational Linguistics, 2023. URL https://aclanthology.org/2023.emnlp-main.963.

[190] Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*, 2020.

[191] Hannes Westermann and Karim Benyekhlef. Justicebot: A methodology for building augmented intelligence tools for laypeople to increase access to justice. In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law*, pages 351–360, 2023.

[192] Hannes Westermann, Sébastien Meeùs, Mia Godet, Aurore Troussel, Jinzhe Tan, Jaromir Savelka, and Karim Benyekhlef. Bridging the gap: Mapping layperson narratives to legal issues with language models. In *Proceedings of the 6th Workshop on Automated Semantic Analysis of Information in Legal Text co-located with the 19th International Conference on Artificial Intelligence and Law (ICAIL 2023), Braga, Portugal*, volume 3441, pages 37–48, 2023.

[193] Richard Whately. *Elements of Logic*. Longman, Green, Longman, Roberts and Green, 1826. URL https://books.google.com/books?hl=fr&lr=&id=89FCAQAAMAAJ&oi=fnd&pg=PR7&dq=Richard+Whately%27s+%22Elements+of+Logic%22&ots=MCzPfy7h7J&sig=2iXrrzUz79MS03Sv1tT-xe93Q88.

[194] Wikipedia. List of fallacies, 2023. URL https://en.wikipedia.org/w/index.php?title=List_of_fallacies&oldid=1178241902.

[195] Wikipedia contributors. Sophistical refutations — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Sophistical_Refutations&oldid=1185233224, 2023. [Online; accessed 4-February-2024].

[196] Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018.

[197] Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2018.

[198] J. Wisse. *Ethos and Pathos: From Aristotle to Cicero*. Hakkert, 1989. ISBN 9789025609634. URL https://books.google.fr/books?id=xsswAAAAIAAJ.

[199] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45, 2020.

[200] Patrick Xia, Shijie Wu, and Benjamin Van Durme. Which* bert? a survey organizing contextualized encoders. In *Conference on Empirical Methods in Natural Language Processing*, 2020.

[201] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 2020.

[202] Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*, 2023.

[203] Fangzhi Xu, Jun Liu, Qika Lin, Yudai Pan, and Lingling Zhang. Logiformer: A two-branch graph transformer network for interpretable logical reasoning. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1055–1065, 2022.

[204] Yichong Xu, Chenguang Zhu, Ruochen Xu, Yang Liu, Michael Zeng, and Xuedong Huang. Fusing context into knowledge graph for commonsense reasoning. In *Annual Meeting of the Association for Computational Linguistics*, 2021.

[205] Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Bing Yin, and Xia Hu. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *arXiv preprint arXiv:2304.13712*, 2023.

[206] Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. G-daug: Generative data augmentation for commonsense reasoning. In *Conference on Empirical Methods in Natural Language Processing*, 2020.

[207] Andrew Yates, Rodrigo Nogueira, and Jimmy Lin. Pretrained transformers for text ranking: Bert and beyond. In *ACM International Conference on Web Search and Data Mining*, 2021.

[208] Nathan Young, Qiming Bao, Joshua Bensemann, and Michael J Witbrock. Abductionrules: Training transformers to explain unexpected inputs. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 218–227, 2022.

[209] Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. Reclor: A reading comprehension dataset requiring logical reasoning. In *International Conference on Learning Representations*, 2020.

[210] Shengbin Yue, Wei Chen, Siyuan Wang, Bingxuan Li, Chenchen Shen, Shujun Liu, Yuxuan Zhou, Yao Xiao, Song Yun, Wei Lin, et al. Disc-lawllm: Fine-tuning large language models for intelligent legal services. *arXiv preprint arXiv:2309.11325*, 2023.

[211] Franco Zappettini. The brexit referendum: How trade and immigration in the discourses of the official campaigns have legitimised a toxic (inter) national logic. *Critical Discourse Studies*, 16(4):403–419, 2019.

[212] Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *Conference on Empirical Methods in Natural Language Processing*, 2018.

[213] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Annual Meeting of the Association for Computational Linguistics*, 2019.

[214] Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*, 2018.

[215] Xuan Zhang and Wei Gao. Towards llm-based fact verification on news claims with a hierarchical step-by-step prompting method. *arXiv preprint arXiv:2310.00305*, 2023.

[216] Wanjun Zhong, Siyuan Wang, Duyu Tang, Zenan Xu, Daya Guo, Jiahai Wang, Jian Yin, Ming Zhou, and Nan Duan. Ar-lsat: Investigating analytical reasoning of text. *arXiv preprint arXiv:2104.06598*, 2021.

[217] Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, 2019.

[218] Ben Zhou, Kyle Richardson, Qiang Ning, Tushar Khot, Ashish Sabharwal, and Dan Roth. Temporal reasoning on implicit events from distant supervision. In *North American Chapter of the Association for Computational Linguistics*, 2021.

[219] Pei Zhou, Rahul Khanna, Bill Yuchen Lin, Daniel Ho, Xiang Ren, and Jay Pujara. Can bert reason? logically equivalent probes for evaluating the inference capabilities of language models. *arXiv preprint arXiv:2005.00782*, 2020.

[220] Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. Evaluating commonsense in pre-trained language models. In *AAAI Conference on Artificial Intelligence*, 2020.

[221] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *IEEE international conference on computer vision*, 2015.

A

Appendix for Chapter 3

A.1 Model Performances on Selected Datasets

Type of Reasoning	Dataset	Model	Performance	Metric
Horn Reasoning (Section: 3.3.1)	ParaRules	RoBERTa	98.8%	Accuracy
	ParaRules	PROVER (Based on RoBERTa)	98.4%	Accuracy
	ParaRules	PRoofWriter (Based on T5)	99.1%	Accuracy
Commonsense Reasoning (Section: 3.3.2)	ProtoQA	GPT-2	71.2%	Accuracy
	CODAH	BERT	69.5%	Accuracy
	CATS	RoBERTa	67%	Accuracy
	PIQA	RoBERTa	77%	Accuracy
Event-based Commonsense Reasoning (Section: 3.3.3)	MCTACO	BERT	69.9%	F1-Score
	TRACIE	SymTime (Based on T5)	80.6%	F1-Score
	TORQUE	RoBERTa	75.2%	F1-Score
Implicit Reasoning (Section: 3.3.4)	LogiQA	RoBERTa	35.31%	Accuracy
	SNLI	EFL (Based on RoBERTa)	93.1%	F1-Score
	LogiQA	DAGN	39.32%	Accuracy
	LogiQA	FOCAL REASONER (Based on RoBERTa)	40.25%	Accuracy
	ReClor EASY	RoBERTa	75.50%	Accuracy
	ReClor HARD	RoBERTa	54.3%	Accuracy
	ReCLor EASY	DAGN (Based on RoBERTa)	75.91%	Accuracy
	ReClor HARD	DAGN (Based on RoBERTa)	44.46%	Accuracy
	ReCLor EASY	FOCAL REASONER (Based on RoBERTa)	77.05%	Accuracy
	ReClor HARD	FOCAL REASONER (Based onRoBERTa)	44.64%	Accuracy
	ReClor EASY	RoBERTa with Logical Data Augmentation	81.4%	Accuracy
	ReClor HARD	RoBERTa with Logical Data Augmentation	62.5%	Accuracy
	AR-LSAT	RoBERTa	23.1%	Accuracy
	QuAIL	BERT	55.9%	Accuracy
	StrategyQA	RoBERTa	66%	Accuracy
	ConTRoL	BART	60.95%	Accuracy
	CLUTTR	GAT	77%	Accuracy
	RICA	GPT-2	50%	Accuracy
	RICA	RoBERTa	50%	Accuracy
Mathematical Reasoning (Section: 3.3.5)	SVAMP	RoBERTa Embeddings + Graph2Tree	65%	Accuracy
	MATH	GPT-2	6.9%	Accuracy
	IsarStep	Hierarchical Transformer	22.8%	Accuracy

Table A.1: Model Performances on Selected Datasets

Appendix for Chapter 6

B.1 Definitions of the fallacies

In the following, we provide, for each fallacy, its informal definition, its formal definition, and a toy example.

We start by describing the variables/placeholders used in the formal templates.

- $A =$ attack
- $E =$ entity (persons, organizations) or group of entities
- $P, P_i =$ premises, properties, or possibilities
- $C =$ conclusion

The following definitions are inspired from Bennett [14] and have been adapted to be more generic.

B.1.1 Fallacies of Emotion

Appeal to Anger

Informal: This fallacy involves using anger or indignation as the main justification for an argument, rather than logical reasoning or evidence.

Formal: E claims P. E is outraged. Therefore, P. Or E_1 claims P. E_2 is outraged by P. Therefore, P (or $\neg P$ depending on the situation).

Example: The victim's family has been torn apart by this act of terror. Put yourselves in their terrible situation, you will see that he is guilty.

Annotation with Variables: E (the speaker) claims P (the accused is guilty) and expresses outrage. Therefore, P (guilt).

Appeal to Fear

Informal: This fallacy occurs when fear or threats are used as the main justification for an argument, rather than logical reasoning or evidence.

Formal: If $\neg P_1$, something terrible P_2 will happen. Therefore, P_1.

Example: If you don't support this politician, our country will be in ruins, so you must support them.

Annotation with Variables: If $\neg P_1$ (not supporting the politician), then P_2 (country in ruins) will happen. Therefore, P_1 (must support the politician).

Appeal to Pity

Informal: This fallacy involves using sympathy or compassion as the main justification for an argument, rather than logical reasoning or evidence.

Formal: P which is pitiful, therefore C, with only a superficial link between P and C

Example: He's really struggling, so he should get the job despite lacking qualifications.

Annotation with Variables: P (he's struggling) is presented as a pitiful situation, leading to C (he should get the job), despite a superficial link between P and C.

Appeal to Positive Emotion

Informal: This fallacy occurs when a positive emotion – like hope, optimism, happiness, or pleasure – is used as the main justification for an argument, rather than logical reasoning or evidence.

Formal: P is positive. Therefore, P.

Example: Smoking a cigarette will make you look cool, you should try it!

Annotation with Variables: P (smoking cigarettes looks cool) leads to P (try smoking).

Appeal to Ridicule

Informal: This fallacy occurs when an opponent's argument is portrayed as absurd or ridiculous with the intention of discrediting it.

Formal: E_1 claims P. E_2 makes P look ridiculous, by misrepresenting P (P'). Therefore, $\neg P$.

Example: There's a proposal to reduce carbon emissions by 50% in the next decade. What's next? Are we all going to stop breathing to reduce CO2?

Annotation with Variables: E_1 (unspecified entity) claims P (proposal to reduce the carbon emissions). E_2 (the speaker) made P looks ridiculous by suggesting an extreme scenario P' (stop beathing). Therefore, $\neg P$ (reducing carbon emission is unreasonable).

Appeal to Worse Problems

Informal: This fallacy involves dismissing an issue or problem by claiming that there are more important issues to deal with, instead of addressing the argument at hand. This fallacy is also known as the "relative privation" fallacy.

Formal: P_1 is presented. P_2 is presented as a best-case. Therefore, P_1 is not that good. OR P_1 is presented. P_2 is presented as a worst-case. Therefore, P_1 is very good.

Example: Why worry about littering when there are bigger problems like global warming?

Annotation with Variables: P_1 (littering) is compared to P_2 (global warming), which is a worse problem, leading to P_1 is not important.

B.1.2 Fallacies of Logic

Causal Oversimplification

Informal: This fallacy occurs when a complex issue is reduced to a single cause and effect, oversimplifying the actual relationships between events or factors.

Formal: P_1 caused C (although P_2, P_3, P_4, etc. also contributed to C.)

Example: There is an economic crisis in the country, the one to blame is the president.

Annotation with Variables: P_1 (the president) caused C (economic crisis), while ignoring other contributing factors (P_2 (worldwide economical context), P_3 (previous policies), etc.).

Circular Reasoning

Informal: This fallacy occurs when an argument assumes the very thing it is trying to prove, resulting in a circular and logically invalid argument.

Formal: C because of P. P because of C. OR C because C.

Example: The best smartphone is the iPhone because Apple creates the best products.

Annotation with Variables: C (iPhone is the best smartphone) because P (Apple creates the best products), which in turn is justified by the claim C.

Equivocation

Informal: This fallacy involves using ambiguous language or changing the meaning of a term within an argument, leading to confusion and false conclusions.

Formal: No logical form: P_1 uses a term T that has a meaning M_1. P_2 uses the term T with the meaning M_2 to mislead.

Example: The government admitted that many cases of credible UFOs (Unidentified flying objects) have been reported. Therefore, that means that Aliens have already visited Earth.

Annotation with Variables: P_1 (many cases of credible UFOs have been reported) uses the term UFO with the meaning M_1 (unidentified flying objects). P_2 (aliens have already visited Earth) uses UFO with a different meaning M_2 (implying that aliens = UFOs), misleading the conclusion.

Fallacy of Division

Informal: This fallacy involves assuming that if something is true for a whole, it must also be true of all or some of its parts.

Formal: E_1 is part of E, E has property P. Therefore, E_1 has property P.

Example: The team is great, so every player on the team must be great.

Annotation with Variables: E_1 (every player) is part of E (the team). E has the property P (great), then E_1 also has P.

False Analogy

Informal: This fallacy involves making an analogy between two elements based on superficial resemblance.

Formal: E_1 is like E_2. E_2 has property P. Therefore, E_1 has property P. (but E_1 really is not too much like E_2)

Example: We should not invest in Space Exploration. It's like saying that a person in debt should pay for fancy vacations.

Annotation with Variables: E_1 (a state in debt plans to explore space) is linked to E_2 (a family in debt plans fancy vacations). E_2 has property P (expensive and not advisable), implying E_1 should also have P.

False Causality

Informal: This fallacy involves incorrectly assuming that one event causes another, usually based on temporal order or correlation rather than a proven causal relationship.

Formal: P is associated with C (when the link is mostly temporal and not logical). Therefore, P causes C.

Example: After the rooster crows, the sun rises; therefore, the rooster causes the sunrise.

Annotation with Variables: P (rooster crows) is associated with C (sunrise), but the link is temporal, not causal, leading to the false conclusion that P causes C.

False Dilemma

Informal: This fallacy occurs when only two options are presented in an argument, even though more options may exist.

Formal: Either P_1 or P_2, while there are other possibilities. OR Either P_1, P_2, or P_3, while there are other possibilities.

Example: You're either with us, or against us.

Annotation with Variables: Presents a choice between P_1 (with us) and P_2 (against us), excluding other possibilities.

Hasty Generalization

Informal: This fallacy occurs when a conclusion is drawn based on insufficient or unrepresentative evidence.

Formal: Sample E_1 is taken from population E. (Sample E_1 is a very small part of population E.) Conclusion C is drawn from sample E_1.

Example: I met two aggressive dogs, so all dogs must be aggressive.

Annotation with Variables: A small sample E_1 (two aggressive dogs) is taken from a larger population E (all dogs). Therefore C (all dogs are aggressive).

Slippery Slope

Informal: This fallacy occurs when it is claimed that a small step will inevitably lead to a chain of events, resulting in a significant negative outcome.

Formal: P_1 implies P_2, then P_2 implies P_3,... then C which is negative. Therefore, $\neg P_1$.

Example: If we allow kids to play video games, they will see fights, guns, and violence, and then they'll become violent adults.

Annotation with Variables: P_1 (allowing kids to play video games) implies P_2 (seeing fights, guns, and violence), which in turns implies P_3 (to like violence, etc.) leading to C (kids becomes violent adults). Therefore, $\neg P_1$.

Strawman Fallacy

Informal: This fallacy involves misrepresenting an opponent's argument, making it easier to attack and discredit.

Formal: E_1 claims P. E_2 restates E_1's claim (in a distorted way P'). E_2 attacks (A) P'. Therefore, $\neg P$.

Example: He says we need better internet security, but I think his panic about hackers is overblown.

Annotation with Variables: E_1 (an unspecified person (He)) claims P (need for better internet security), E_2 (the speaker) distorts the claim as P' (panic about hackers). Therefore $\neg P$.

B.1.3 Fallacies of Credibility

Abusive Ad Hominem

Informal: This fallacy involves attacking a person's character or motives instead of addressing the substance of their argument.

Formal: E claims P. E's character is attacked (A). Therefore, $\neg P$.

Example: "John says the earth is round, but he's a convicted criminal, so he must be wrong."

Annotation with Variables: E (John) claims P (the earth is round). John's character is attacked (A) (being a criminal). Therefore, $\neg P$ (the earth is not round).

Ad Populum

Informal: This fallacy involves claiming that an idea or action is valid because it is popular or widely accepted.

Formal: A lot of people believe/do P. Therefore, P. OR Only a few people believe/do P. Therefore, $\neg P$.

Example: Millions of people believe in astrology, so it must be true.

Annotation with Variables: Many people believe in P (astrology). Therefore, P (astrology is true).

Appeal to Authority

Informal: This fallacy occurs when an argument relies on the opinion or endorsement of an authority figure who may not have relevant expertise or whose expertise is questionable. When applicable, a scientific consensus is not an appeal to authority.

Formal: E claims P (when E is seen as an authority on the facts relevant to P). Therefore, P.

Example: A famous actor says this health supplement works, so it must be effective.

Annotation with Variables: E (famous actor) claims P (the health supplement works). Therefore, P (it must be effective).

Appeal to Nature

Informal: This fallacy occurs when something is assumed to be good or desirable simply because it is natural, while its unnatural counterpart is assumed to be bad or undesirable.

Formal: P_1 is natural. P_2 is not natural. Therefore, P_1 is better than P_2. OR P_1 is natural, therefore P_1 is good.

Example: Herbs are natural, so they are better than synthetic medicines.

Annotation with Variables: P_1 (herbs are natural) and P_2 (synthetic medicines are not natural), leading to P_1 is better than P_2.

Appeal to Tradition

Informal: This fallacy involves arguing that something should continue to be done a certain way because it has always been done that way, rather than evaluating its merits.

Formal: We have been doing P for generations. Therefore, we should keep doing P. OR Our ancestors thought P. Therefore, P.

Example: We've always had a meat dish at Thanksgiving, so we should not change it.

Annotation with Variables: P (always had a meat dish at Thanksgiving) should continue. Therefore, continue P.

Guilt by Association

Informal: This fallacy involves discrediting an idea or person based on their association with another person, group, or idea that is viewed negatively.

Formal: E_1 claims P. Also E_2 claims P, and E_2's character is attacked (A). Therefore, $\neg P$. OR E_1 claims P. E_2's character is attacked (A) and is similar to E_1. Therefore $\neg P$.

Example: Alice believes in climate change, just like the discredited scientist Bob, so her belief must be false.

Annotation with Variables: E_1 (Alice) claims P (belief in climate change). E_2 (Bob) also claims P. However E_2's character (A) is attacked (being discredited). Therefore $\neg P$.

Tu Quoque

Informal: This fallacy occurs when someone's argument is dismissed because they are accused of acting inconsistently with their claim, rather than addressing the argument itself.

Formal: E claims P, but E is acting as if $\neg P$. Therefore $\neg P$.

Example: Laura advocates for healthy eating but was seen eating a burger, so her advice on diet is invalid.

Annotation with Variables: E (Laura) claims P (advocates for healthy eating), but E is acting as if $\neg P$ (eating a burger, which is unhealthy eating). Therefore $\neg P$ (advice on diet is invalid).

Our categorization of fallacies into logic, emotion, and credibility is based on the primary aspect of the fallacy that leads to an invalid or weak argument. In practice, some fallacies could be argued to fit into more than one category.

B.2 Metrics Edge Cases

In this section, we show how our metrics handle edge cases of our disjunctive annotation scheme.

Table B.1: The model predicts at least one correct label

	Spans	Labels		
Gold	Lorem ipsum dolor sit amet. Ut enim ad minim veniam. Sed do eiusmod tempor incididunt.	$l1, \perp$ $l2$ $l3$		
Case	Prediction	Label	Recall	Precision
0.1	Ut enim ad minim veniam.	$l2$	0.5	1
0.2	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	$l1$ $l2$	0.5	1
0.3	Ut enim ad minim veniam. Sed do eiusmod tempor incididunt.	$l2$ $l4$	0.5	0.5
0.4	Lorem ipsum dolor sit amet. Ut enim ad minim veniam. Sed do eiusmod tempor incididunt.	$l1$ $l2$ $l4$	0.5	0.666

Table B.2: The gold standard has only one span, which has a "no fallacy" as an alternative

	Spans	Labels	Recall	Precision
Gold	Lorem ipsum dolor sit amet.	$l1, \perp$		
Case	Prediction	Label		
1.1	Lorem ipsum dolor sit amet.	$l1$	1	1
1.2	Lorem ipsum dolor sit amet.	$l3$	1	0
1.3	-	-	1	1
1.4	Ut enim ad minim veniam.	$l1$	1	0
1.5	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	$l3$	1	0

Table B.3: The gold standard does not have a "no fallacy"

	Spans	Labels	Recall	Precision
Gold	Lorem ipsum dolor sit amet.	$l1$		
Case	Prediction	Label		
2.1	Lorem ipsum dolor sit amet.	$l1$	1	1
2.2	Lorem ipsum dolor sit amet.	$l3$	0	0
2.3	Ut enim ad minim veniam.	$l1$	0	0
2.4	-	-	0	1

Table B.4: The gold standard contains no fallacious spans

	Spans	Labels	Recall	Precision
Gold	-	-		
Case	Prediction	Label		
3.1	Lorem ipsum dolor sit amet.	$l1$	1	0
3.2	-	-	1	1

Table B.5: Two gold standard spans, one has a "no fallacy" alternative and the other one a required fallacy

	Spans	Labels	Recall	Precision
Gold	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	$l1, \perp$ $l2$		
Case	Prediction	Label		
4.1	Lorem ipsum dolor sit amet.	$l1$	0	1
4.2	Lorem ipsum dolor sit amet.	$l3$	0	0
4.3	Ut enim ad minim veniam.	$l2$	1	1
4.4	Ut enim ad minim veniam.	$l3$	0	0

Table B.6: The gold standard spans across 2 sentences

	Spans	Labels	Recall	Precision
Gold	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	$l1, \perp$		
Case	Prediction	Label		
5.1	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	$l1$	1	1
5.2	-	-	1	1
5.3	Lorem ipsum dolor sit amet.	$l1$	1	1
5.4	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	$l3$	1	0
5.5	Ut enim ad minim veniam.	$l3$	1	0

Table B.7: Two overlapping gold standard spans, but one span has a "no fallacy" as an alternative and the other one has a required fallacy

	Spans	Labels	Recall	Precision
Gold	Lorem ipsum dolor sit amet. Ut enim ad minim veniam. Ut enim ad minim veniam.	$l1, \perp$ $l2$		
Case	Prediction	Label		
6.1	Lorem ipsum dolor sit amet. Ut enim ad minim veniam. Ut enim ad minim veniam.	$l1$ $l2$	1	1
6.2	Ut enim ad minim veniam.	$l2$	1	1
6.3	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	$l1$	0	1
6.4	-	-	0	1
6.5	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	$l1$ $l2$	1	1
6.6	Lorem ipsum dolor sit amet.	$l1$	0	1
6.7	Ut enim ad minim veniam.	$l3$	0	0

Table B.8: Two overlapping gold standard spans labeled differently

	Spans	Labels	Recall	Precision
Gold	Lorem ipsum dolor sit amet. Ut enim ad minim veniam. Ut enim ad minim veniam.	$l1$ $l2$		
Case	Prediction	Label		
7.1	Lorem ipsum dolor sit amet. Ut enim ad minim veniam. Ut enim ad minim veniam.	$l1$ $l2$	1	1
7.2	Ut enim ad minim veniam.	$l2$	0.5	1
7.3	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	$l1$	0.5	1
7.4	-	-	0	1
7.5	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	$l1$ $l2$	0.75	1
7.6	Lorem ipsum dolor sit amet.	$l1$	0.25	1
7.7	Ut enim ad minim veniam.	$l3$	0	0

Table B.9: Two labels have the same Level 0 or Level 1 fallacy category

	Spans	Labels	Recall	Precision
Gold	Lorem ipsum dolor sit amet. Ut enim ad minim veniam. Ut enim ad minim veniam.	fallacy (l1) fallacy (l2)		
Case	Prediction	Label		
8.1	Lorem ipsum dolor sit amet. Ut enim ad minim veniam. Ut enim ad minim veniam.	fallacy fallacy	1	1
8.2	Ut enim ad minim veniam.	fallacy	0.75	1
8.3	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	fallacy	1	1
8.4	Ut enim ad minim veniam. Ut enim ad minim veniam.	fallacy fallacy (duplicate)	0.75	0.5
8.5	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	fallacy fallacy	0.75	1
8.6	Lorem ipsum dolor sit amet.	fallacy (l1)	0.25	1
8.7	Lorem ipsum dolor sit amet.	fallacy (l2)	0.25	1

Table B.10: Two labels have the same Level 0 or Level 1 fallacy category with an alternative "no fallacy"

	Spans	Labels	Recall	Precision
Gold	Ut enim ad minim veniam. Ut enim ad minim veniam.	fallacy (l1), $\perp$ fallacy (l2)		
Case	Prediction	Label		
9.1	Ut enim ad minim veniam. Ut enim ad minim veniam.	fallacy fallacy (duplicate)	1	1
9.2	Ut enim ad minim veniam.	fallacy (l1)	1	1
9.3	Ut enim ad minim veniam.	fallacy (l2)	1	1
9.4	Lorem ipsum dolor sit amet. Ut enim ad minim veniam.	fallacy	1	0.5
9.5	-	-	0	1
9.6	Lorem ipsum dolor sit amet.	fallacy	0	0

Table B.11: Two same fallacious gold standard spans labeled differently

	Spans	Labels	Recall	Precision
Gold	Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet.	$l1$ $l2$		
Case	Prediction	Label		
10.1	Lorem ipsum dolor sit amet.	$l1$	0.5	1
10.2	Lorem ipsum dolor sit amet.	$l3$	0	0
10.3	Ut enim ad minim veniam.	$l1$	0	0
10.4	-	-	0	1
10.5	Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet.	$l1$ $l2$	1	1

Table B.12: Two same fallacious gold standard spans, but one has a "no fallacy" alternative and the other one has a required fallacy

	Spans	Labels	Recall	Precision
Gold	Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet.	$l1, \perp$ $l2$		
Case	Prediction	Label		
11.1	Lorem ipsum dolor sit amet.	$l1$	1	1
11.2	Lorem ipsum dolor sit amet.	$l3$	0	0
11.3	Ut enim ad minim veniam.	$l1$	0	0
11.4	-	-	0	1
11.5	Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet.	$l1$ $l2$	1	1

ECOLE
DOCTORALE

Titre: Evaluation et Amélioration des Capacités de Raisonnement des Modèles de Langage

Mots clés: apprentissage en profondeur, modèles de langage, traitement automatique du langage, apprentissage automatique, IA neuro-symbolique

Résumé: Cette thèse examine les capacités de raisonnement des Petits Modèles de Langage (SLMs) et Grands Modèles de Langage (LLMs) et expose leurs limites. Elle présente LogiTorch, une bibliothèque Python facilitant l'entraînement de modèles sur diverses tâches de raisonnement. La thèse inclut également TINA, une technique d'augmentation de données qui renforce la robustesse des SLMs face à la négation dans les tâches d'implication textuelle. De plus, la thèse explore les capacités des LLMs avec MAFALDA, un nouveau benchmark pour la classification des sophismes, intégrant une métrique d'évaluation qui considère la subjectivité. Les résultats montrent que les humains surpassent les modèles dans cette tâche de raisonnement. Nous proposons plusieurs directions de recherche qui méritent une investigation plus approfondie, telles que l'exploration de l'IA Neuro-symbolique et l'amélioration des capacités de raisonnement des LLMs à faibles ressources.

Title: Evaluating and Improving the Reasoning Abilities of Language Models

Keywords: deep learning, language models, natural language processing, machine learning, neruo-symbolic AI

Abstract: This thesis focuses on evaluating and improving the reasoning abilities of Smaller Language Models (SLMs) and Large Language Models (LLMs). It explores SLMs' performance on complex tasks and their limitations with simpler ones. This thesis introduces LogiTorch, a Python library that facilitates the training of models on various reasoning tasks with minimal coding. It also presents TINA, a negated data augmentation technique that improves SLMs' robustness to negation in textual entailment tasks. Further, this thesis explores LLMs' capabilities through MAFALDA, a new benchmark for identifying and classifying reasoning fallacies, proposing a new annotation scheme and evaluation metric that considers subjectivity in reasoning. The findings indicate that humans outperform SLMs and LLMs in this reasoning task. We propose several research directions that merit further investigation, such as investigating Neuro-symbolic AI and improving the reasoning abilities of low-resource LLMs.

Institut Polytechnique de Paris
91120 Palaiseau, France