Dependency Parsing as an Upstream Task for Logical Reasoning

David Heineman, Drew Mayernik, Anoop Gangireddy, Nevin Aresh
Georgia Institute of Technology
{david.heineman, dmayernik3, anoopgangireddy542, naresh6}@gatech.edu

Abstract

Logical reasoning requires understanding and synthesizing sentence-level logic constructs. Pre-trained models are effective at identifying and performing common-sense reasoning over word-level semantics but lack the nuanced ability to parse logic. Past approaches have attempted to translate text to explicit first-order logical representations with a dictionary-based but are limited to basic, explicit relationships. We use dependency trees to describe the semantic structure of a sentence through the relationship between words, thus highlighting the logical framework of the sentence. We show how our method of extending RoBERTa achieves state-of-the-art performance on a baseline dataset and argue broadly how dependency trees can assist tasks which require logical understanding.[1]

1 Introduction

One of the largest fields of research in machine learning is natural language processing (NLP) as understanding language is key for advanced ML applications. In recent years, logical reasoning has become a focus of NLP research because it requires models to demonstrate a more complex understanding of the underlying meaning present in language (McCarthy, 1989; Nilsson, 1991). Even as large-scale models begin approaching a gold-standard ability to process and generate text, logical reasoning has been demonstrated to be a substantial challenge in the field (Williams et al., 2017; Habernal et al., 2017). Additionally, the ability to identify and perform operations on cardinal information has shown to be promising for aiding text generation and parsing tasks (Mehta et al., 2021; Dou et al., 2021). Large, pre-trained language models have proven effective at common-sense reasoning, but primarily rely on identifying individual words, rather than overarching sentence-level logic, and thus, lack the proper context to decipher true logic. (Liu et al., 2019; Devlin et al., 2018; Williams et al., 2017).

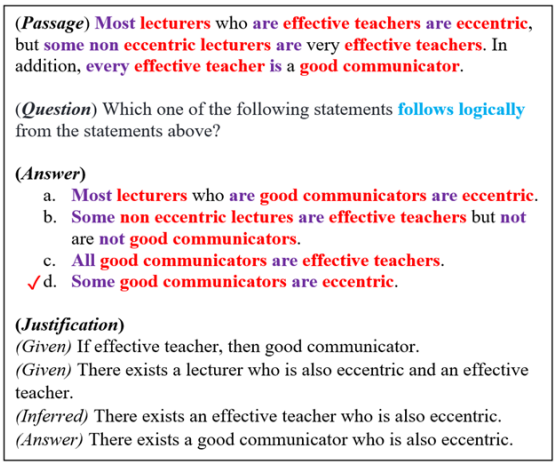

Figure 1: An example from ReClor dataset. Highlighted are entities, relations between entities and key words for solving the problem. ✓ indicates the correct answer.

As a stand-in for the task of logical reasoning, we use questions from logic-based exams (Yu et al., 2020) to measure performance. An example is shown in figure 1. Both a context and question is given and the answer is clearly a logical extension of the given text. To solve the problem, the explicit logical relationships between entities must be recognized and inferred.

Past approaches (Wang et al., 2021) attempt to convert the prompts and questions to an explicit, first order logical representation and compute the answer based on these generated representations. Although this is shown to significantly improve accuracy, a dictionary-based approach is limited in that it only captures basic, explicit relationships between entities (Ouyang et al., 2021). Instead, we attempt to use dependency trees, which describe

[1]Our code and data has been released at https://github.com/davidheineman/dep-logic.

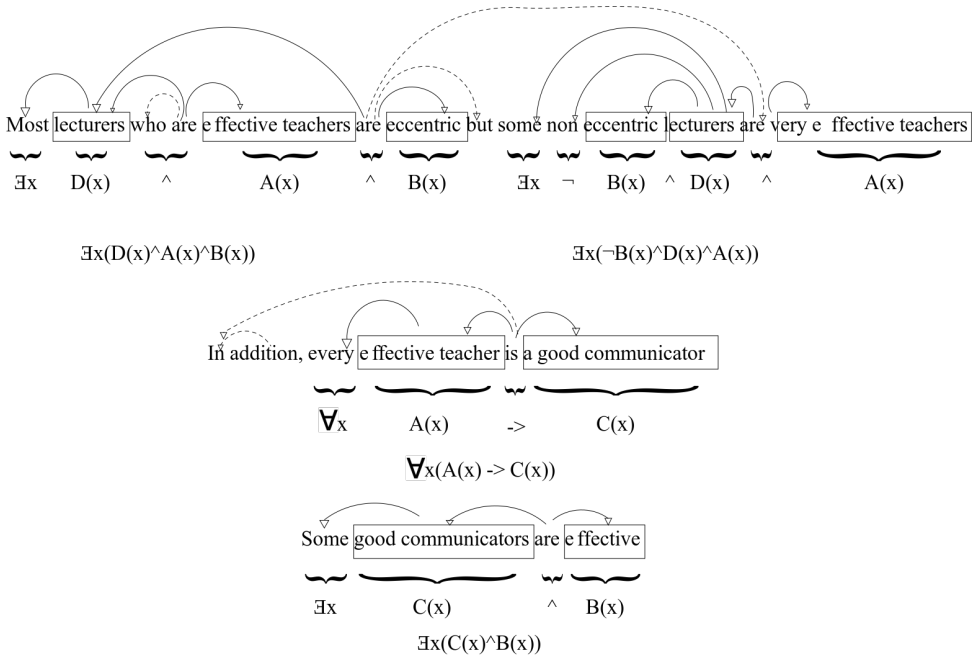

Figure 2: The syntax tree and logical representations of the context and correct answer from figure 1. The solid lines represent meaningful relationships between entities which represent logical relationships and the dotted lines are inconsequential relationships.

the semantic structure of a sentence by representing the relationships between words as a tree. This approach to encoding logical forms has shown to retain the framework of the sentence while stripping the semantics of the sentence (Reddy et al., 2016). Figure 2 shows the syntax and logical representations of the context, the correct answer and how the syntax representation of the sentence implicitly encodes the logical representation.

For our baseline, we use RoBERTa and BERT and measure performance using the logical reasoning benchmark dataset ReClor (Yu et al., 2020). We show that the inclusion of the explicit dependency representation of the input context, question, and answers achieves state-of-the-art performance, and that dependency trees can more broadly be used as an upstream step for increasing the efficacy of logic-reliant tasks.

2 Related Work

Our work focuses on the utilization of logical reasoning methods to analyze the ReClor dataset (Yu et al., 2020) and use RoBERTa to determine the most plausible answer from the given set of four multiple choice options.

2.1 Dataset Development

A multitude of datasets have been proposed in order to better represent the variety that can arise between multiple choice questions. One of the earliest datasets proposed for reading comprehension is the MCTest (Richardson et al., 2013) which is comprised of 2,000 multiple choice questions about fictional stories. The much more comprehensive SQuAD dataset (Rajpurkar et al., 2016) was proposed three years later with the intention of creating more variability in types of questions. Composing of over 100,000 multiple questions taken from Wikipedia articles, SQuAD (Rajpurkar et al., 2016) was believed to be robust and served as a baseline for NLP programmers. However, further examination of the aforementioned data sets found (Louis et al., 2016) that less than 2% of the questions in these datasets necessitated the use logical reasoning.

In 2020, the ReClor (Yu et al., 2020) and LogicQA (Liu et al., 2020) datasets were proposed to tackle the need for a robust and diverse dataset. The former (Yu et al., 2020) utilizes graduate admissions examinations to present a dataset of 6,138 multiple choice questions for **Re**ading **C**omprehension that utilizes **lo**gical **r**egression. LogicQA (Yu et al., 2020) follows a similar principle in that it presents 8,678 questions written by experts that test a neural model's ability of deductive reasoning.

2.2 Models to Tackle Multiple Choice QAs

Several prominent papers have attempted to use BERT models (Devlin et al., 2018) to train and analyze multiple choice problems. A precursor to many logic focused multiple choice questions are logic puzzles which are used to decipher simple logic connections. A 2019 study (Escamocher and O'Sullivan, 2019) developed an algorithm which recreates the methods a person may use to solve a puzzle as well as explain the basis used to arrive at that conclusion. From here, studies have attempted to correlate the research between algorithms that solve logic puzzles to those that can solve more complex logic-based multiple choice questions, such as those given on the LSAT. A paper written at Sun Yat-Sen University (Wang et al., 2021) focuses on parsing the questions semantically then determining which logical expressions can be derived from the parsed expression of the question. A recent Stanford paper (Xu et al., 2019) proposed a DCN based on BERT in order to analyze Passage/Question and Passage/Answer datasets and achieved an accuracy of 62.2%. A similar study at MIT (Tran et al., 2021) utilized multiple choice questions from their Intro to Machine Learning class and attempted to devise a model that would answer these questions achieving an accuracy of 97%.

3 Our Method

3.1 Dataset & Problem Definition

Our multiple choice question answering task is formalized as follows: for any problem p given a context, $p_{context}$, a question, $p_{question}$, and candidate answers, $\{a_1, a_2, a_3, a_4\}$, select the correct answer $\hat{a} \in \{a_1, a_2, a_3, a_4\}$. Our data comes from the widely-used and standardized ReClor dataset (Yu et al., 2020) consisting of exclusively logic-type reasoning-type questions from the Graduate Management Admission Test (GMAT) and Law School Admission Test (LSAT) and are popular for measuring performance on logical reasoning QA. Each question must be answered through some form of extending the logical reasoning verbalized within the context and question. We submit our model to a public leaderboard for evaluation on the test data for ReClor [2]

[2] eval.ai/web/challenges/challenge-page/503/leaderboard/1347

3.2 Baseline Models

We use out-of-the-box pretrained models as a benchmark to measure our performance. For each answer $a_n \in \{a_1, a_2, a_3, a_4\}$ we concatenate the context, question and corresponding BERT tokens into four separate representations: $\langle s \rangle\ c\ \langle /s \rangle\ q\ \langle d \rangle\ a_n\ \langle /s \rangle$, where $\langle s \rangle$ and $\langle /s \rangle$ are classification and separator tokens, respectively, and $\langle d \rangle$ is a custom token to highlight the separation between information given in the question and the current answer. The probability for each candidate answer, $P(o_n|c, q)$, is calculated using a softmax function, and the answer with the highest probability is selected. Therefore, we simply minimize the cross entropy loss as given by:

$$\mathcal{L} = -\sum \log P(\hat{a}|c, q)$$

We report scores using both RoBERTa (Liu et al., 2019) and BERT (Devlin et al., 2018), which have been shown to have broad application, despite limited effectiveness at logical reasoning.

Past research has shown that these baselines are mostly effective at capturing logic represented in word-level semantics (Wang et al., 2021) and fail to learn sentence-level semantics, making them effective baselines to measure the increase in logical understanding through our modifications.

3.3 Syntax Aware Models

We attempt two methods of passing in dependency trees.

All depedency trees are simply generated using the SpaCy[3] python library using the *en_core_web_lg* model with no modifications to the trees after their generation.

Figure 3 shows an example dependency tree visualized as a graph and its corresponding textual encoding. We initially report scores concatenating the textual representation to the base BERT model. We also adopt a similar approach to encoding the dependency representations of problems as (Bai et al., 2021), as illustrated in Figure 3.

The problem with directly encoding the dependency tree is that the self-attention layer in BERT merely calculates the relevance between pairs of adjacent tokens, rather than allowing for attention which spans between tokens in completely unrelated parts of the sentence. Although encoding the structure of a dependency tree is possible without

[3] spacy.io

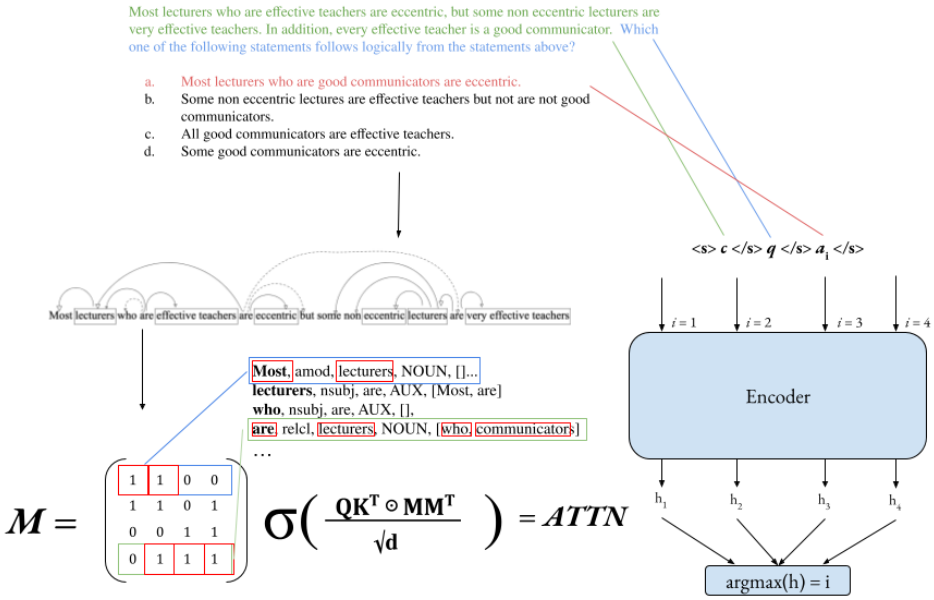

Figure 3: Visualization of our syntax-aware model.

modifying the attention layer, this can only capture the parent of each token, meaning the sibling and children relationships between tokens fail to be considered (Strubell et al., 2018).

We encode the dependency tree as a prior for the language model's self-attention layer through creating a sparse, two dimensional mask representing the relationships between tokens in the dependency tree. The sentence is embedded in BERT identically to our baseline method, relying on the mask to encode the relationships between words.

We represent our new Dependency Attention Mask as $D \in \{0,1\}^{nxn}$, where $D_{i,j} \in \{0,1\}$ specifies the existence of a connection from token i to token j, and n is the total number of tokens in the input. The dependency mask captures the interactions between the tokens in a single input. The masked-self attention is identical to that of standard self-attention, with the addition of sparse network connections defined by our Dependency Attention Mask. The dependency mask applied to the self-attention layer can be mathematically described as the Hadamard product between the dot-product attention and the Gramian matrix of the transpose of the corresponding mask:

$$ATTN = \sigma\left(\frac{QK^T \odot MM^T}{\sqrt{d}}\right)$$

where Q, K, and V represent the query, key, and value matrices respectively. d represents the number of attention heads on the self attention layer. We repeat this calculation throughout each self-attention layer.

4 Experiments

We compare our method with the baseline models listed in §3.2. All models were implemented in PyTorch using the Huggingface Transformers library [4]. Following Wang et al. (2021), we fine-tune using a batch size of 8 for 10 epochs each, using AdamW as an optimizer with a learning rate of 1^{-5} (Loshchilov and Hutter, 2017). The ReClor dataset (Yu et al., 2020) is split into 6,138 total question-answer pairs, which we split into 4638/500/1000 train, val and test sets respectively.

We also provide human performance over 100 randomly chosen questions as reported by Yu et al. (2020) to serve as a gold-standard of performance.

Models	Acc_{test}	$Accuracy_{val}$	$F1_{test}$	$F1_{val}$
BERT	23.4	26.1	15.6	16.4
RoBERTa	32.4	35.3	27.2	25.4
Syntax Trees[1]	40.1	42.7	30.5	32.1
Attention Mask[2]	48.2	49.6	35.1	36.2
Human	**63.0**	**60.0**	**50.2**	**48.5**

Table 1: Performance of various models on the ReClor dataset. Results in **bold** represent superior performance. [1] represents the RoBERTa model using the textual syntax trees and [2] represents the RoBERTa model with a modified attention mask.

[4] huggingface.co

5 Conclusion & Future Work

We demonstrate non-trivial improvement over baseline methods (Table 1). It is possible to use dependency trees to properly parse the semantics of a multiple choice logical reasoning question and using these dependency trees is a valid method of building upon existing work. Moving forward, future work could explore more applications of dependency trees in language processing: improving the efficiency of our method, using constituency trees, or applying dependency trees to text generation are a few examples. While our method worked and improved on current methods, our training times were slow and it may not be scale-able to larger sets of data thus using additional python libraries to maximize the efficiency of our code could allow for our method to be more widely applicable. Furthermore, it is possible that, instead of using dependency trees, using constituency trees which are based on context-free grammar could yield better results. Finally, it may be possible to apply our method to text generation. A common problem with text generation is the generation of contradicting logic in sentences. If dependency trees were used as an adversarial training method, there could be noticeable gains in the actual logic of sentences created through text generation. Overall, we have shown that it is possible to use upstream tasks like dependency parsing to improve on the results generated by pre-trained models such as RoBERTa allowing for the expansion of those models to more logic based applications.

References

Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu, and Yunhai Tong. 2021. Syntax-bert: Improving pre-trained transformers with syntax trees. *arXiv preprint arXiv:2103.04350*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yao Dou, Maxwell Forbes, Rik Koncel-Kedziorski, Noah A Smith, and Yejin Choi. 2021. Scarecrow: A framework for scrutinizing machine text. *arXiv preprint arXiv:2107.01294*.

Guillaume Escamocher and Barry O'Sullivan. 2019. Solving logic grid puzzles with an algorithm that imitates human behavior. *arXiv preprint arXiv:1910.06636*.

Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. 2017. The argument reasoning comprehension task: Identification and reconstruction of implicit warrants. *arXiv preprint arXiv:1708.01425*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*.

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

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Annie Louis, Michael Roth, Bonnie Webber, Michael White, and Luke Zettlemoyer. 2016. Proceedings of the workshop on uphill battles in language processing: Scaling early achievements to robust methods. In *Proceedings of the Workshop on Uphill Battles in Language Processing: Scaling Early Achievements to Robust Methods*.

John McCarthy. 1989. Artificial intelligence, logic and formalizing common sense. In *Philosophical logic and artificial intelligence*, pages 161–190. Springer.

Sanket Vaibhav Mehta, Jinfeng Rao, Yi Tay, Mihir Kale, Ankur Parikh, Hongtao Zhong, and Emma Strubell. 2021. Improving compositional generalization with self-training for data-to-text generation. *arXiv preprint arXiv:2110.08467*.

Nils J Nilsson. 1991. Logic and artificial intelligence. *Artificial intelligence*, 47(1-3):31–56.

Siru Ouyang, Zhuosheng Zhang, and Hai Zhao. 2021. Fact-driven logical reasoning. *arXiv preprint arXiv:2105.10334*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Siva Reddy, Oscar Täckström, Michael Collins, Tom Kwiatkowski, Dipanjan Das, Mark Steedman, and Mirella Lapata. 2016. Transforming dependency structures to logical forms for semantic parsing. *Transactions of the Association for Computational Linguistics*, 4:127–140.

Matthew Richardson, Christopher JC Burges, and Erin Renshaw. 2013. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 193–203.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. *arXiv preprint arXiv:1804.08199*.

Sunny Tran, Pranav Krishna, Ishan Pakuwal, Prabhakar Kafle, Nikhil Singh, Jayson Lynch, and Iddo Drori. 2021. Solving machine learning problems. In *Asian Conference on Machine Learning*, pages 470–485. PMLR.

Siyuan Wang, Wanjun Zhong, Duyu Tang, Zhongyu Wei, Zhihao Fan, Daxin Jiang, Ming Zhou, and Nan Duan. 2021. Logic-driven context extension and data augmentation for logical reasoning of text. *arXiv preprint arXiv:2105.03659*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Kegang Xu, Jingjie Tin, and Jungyoun Kim. 2019. A bert based model for multiple-choice reading comprehension. *Passages*, 6(368):362.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2020. Reclor: A reading comprehension dataset requiring logical reasoning. *arXiv preprint arXiv:2002.04326*.