CS584: YOUR PROJECT TITLE

Jeremiah McReynolds[1]
[1]Stevens Institute of Technology
jmcreyno@stevens.edu,

ABSTRACT

Recent research has shown that pre-trained Language Models (LMs) may store common-sense and factual information in their weights. Indeed, off-the-shelf LMs perform fairly well at answering zero-shot cloze-style questions. More recently, researchers discovered an unsupervised, LM-centric approach to answering cloze-style questions that rivals performance with supervised approaches. Given that background, some researchers have explored whether or not LMs are able to replace existing fact-checking systems. Early results suggest that it may be possible to use LMs as fact-checkers; however, the performance of the LMs pales in comparison to state-of-the-art fact checking systems. In this project, I explored (1) a variety of pre-trained language model's performance on fact-checking, (2) how context affects an LM's ability to fact-check, and (3) whether or not an off-the-shelf question and answering system can perform fact-checking.

1 Introduction

Pre-trained Language Models (LMs) have become central to many downstream Natural Language Processing (NLP) tasks. State-of-the-art models such as BERT [2], RoBERTa [9], and T5 [14] have created the possibility for LM-only question and answer, summarization, and commonsense reasoning pipelines. In fact, there is promising research with closed-book question and answering systems using only language models [14]. However, until recently, it was not thought that language models could be used as fact-verification systems.

Fact-verification is the task of validating the truthfulness of a given claim. At its core, fact checking is a classification problem that states whether or not a given statement is Supported, Refuted or that there is Not Enough Information (NEI). Systems that solve this problem have become increasingly important in recent years. Today, due to advancements in Natural Language Processing (NLP) tasks, convincing misinformation can be generated automatically through the use of pre-trained Language Models (LMs) [18]. Because of this, vast quantities of misinformation can be generated and dispersed quickly, which can be dangerous. In fact, misinformation is a top national security threat for the United States [11].

To help combat this problem, automated fact-verification has become a very important research topic [16]. However, systems that solve this problem are often complex, computationally expensive and rely on manual updates to external knowledge bases [7]. For example, a typical system may perform the following procedures: (1) gather relevant documents from an external knowledge base, (2) extract relevant evidences from the documents, and (3) classify the statement and return appropriate evidences [7]. Given these complex pipelines, researchers have been searching for a simpler approach to fact-verification. Fortunately, recent research suggests that there is a possibility of an LM-only fact-verification pipeline.

Facebook Research has recently built a promising LM-only fact-verification pipeline that is simpler to implement and not reliant on an external knowledge base [7]. In their experiment, they use a three (3) layer pipeline to predict whether a given claim is Supported, Refuted or NEI. This project aims to extend this research, first, by by measuring the effectiveness of this pipeline using other language models. In particular, I measure the performance of off-the-shelf BERT, BART, RoBERTa, and ALBERT models.

Second, I suggest three key modifications to their pipeline.

1. ***Generative Context***: First, I suggest adding an autoregressive text-generation layer to the fact-checking pipeline. Drawing inspiration from Petroni et al. 2020 [13], the goal of this layer is to provide more context

about the statement being presented to the model. As was shown with Question and Answering systems, giving more context to an LM can improve its performance for accurately answering a question [13]. I explore whether this same principal holds true with fact-verification.

2. ***From Claim to Question***: Second, I suggest modifying the ”claim” to become a ”question”. Question and Answer systems have been thoroughly researched and can be applied to many domains with fine-tuning. Moreover, closed-book question and answering systems have proven to be quite effective [15]. As such, I suggest generating a question from the claim with the ”answer” being the masked token. Then, the pipeline can use a closed-book question and answer model for fact-verification. As I will show, this approach solves a key problem posed in the original paper.
3. ***Question-Context Generation***: Finally, I suggest a combination of the two above approaches. First, the pipeline should generate context from the claim using a fine-tuned autoregressive model, then it should generate a question from the claim. Finally, I suggest using the generated context and the claim in an off-the-shelf question and answer LM to determine whether a given claim is factual.

Importantly, none of the suggested approaches leverage an external knowledge base, thus retaining the key improvements proposed by the original LM-only model. Furthermore, these approaches solve a key problem of only being able to fill a single masked token. The original pipeline only masks a single token, then fills that token for verification. Since a closed- or open-book question and answering system can generate multiple tokens in response to a question, using a QA-enabled pipeline enables masking multiple tokens which more completely verifies a given claim.

2 Background

2.1 Question Answering

Question and Answering (Q&A) is the task of providing answers to a proposed question. One of the most popular variants of this task is extracting the answer from a given relevant context. Research has shown that LM models are quite effective at such tasks, and there are many open-source models available to accomplish these tasks [9] [2]. As proof, off-the-shelf models, such as BERT [2] and RoBERTa [9], exhibit state-of-the-art results in extractive Q&A tasks on the SQuAD 2.0 dataset [17].

A separate yet related task is open-domain Q&A, whereby a model must answer an open-domain question without being given explicit context. Notably, it is often assumed that the model has access to external data sources to ”find” the answer [15]. Cleverly, Roberts et al. [15] likened this to an ”open-book” exam, where the model is able to extract the answer from a ”book” [15]. Indeed, RNN-based systems like DrQA [1] have proven effective at this task, achieving state-of-the-art results on the original SQuaD dataset.

More recently, however, research shows that pre-trained Language Models (LMs) hold large quantities of relational data in their weights [12]. Indeed, recent research has shown that autoregressive language models can generate factual information which may be used for Q&A tasks [10] [13]. Indeed, Petroni et al [13] demonstrated that the addition of an autoregressive layer to an open-domain Q&A system improved performance by 7% [13]. This begged whether models could answer questions ”closed-book” style.

Today, an LM-only pipeline can perform as well as complex supervised systems in closed-book style question and answer tasks [15]. Different from ”open-book” Q&A, these models do not require context to successfully answer questions. Moreover, they do not rely on external data sources to answer question. They are only answering questions solely on the knowledge retained in their weights.

This success begs the question on whether or not these off-the-shelf models can be used not only to answer questions, but to fact-check statements. As you will see, fact-verification is a similar task to open-domain Q&A.

2.2 Fact-Verification

Fact-verification is the task of validating the truthfulness of a given claim. Particularly, it is classifying a claim as supported, refuted, or not enough information (NEI). Often, these systems rely on document retrieval, sentence extraction and textual entailment [4].

1. ***Document Retrieval***: The system will use a module to extract relevant documents from a pre-defined knowledge base (e.g. Wikipedia).[1]

[1] You can see can example of a system here `https://github.com/UKPLab/fever-2018-team-athene/blob/master/README.md`

2. ***Document Extraction***: From the extracted documents, the system will select sentences that are relevant to the given claim (”facts”).
3. ***Textual Entailment***: Finally, the system will present the collected ”facts” and the presented claim to a textual entailment model[2], which outputs a label of ”Supported”, ”Refuted” or ”Not Enough Info”.

Notably, these tasks are quite similar to open-domain Q&A: they both must retrieve relevant documents and extract specific sentences. The key difference is that the Q&A system returns those results to answer the question, while the fact-verification system performs an additional textual entailment step to classify a claim.

These systems have two key disadvantages. Firstly, they rely on an external knowledge base which much be manually up-kept to ensure model accuracy. Secondly, they are quite computationally expensive to both implement and run. As such, researchers have been searching for a simpler approach to fact-verification. Fortunately, there has been promising research on a simpler, LM-forward approach to fact-verification.

As with closed-book Q&A, early research proved that designing an LM-only fact-checking system may be possible. Although not as performant as traditional systems, a BERT fact-checking system was able to pass standard benchmarks on the FEVER dataset [7, 16]. Notably, this LM approach to fact-checking is overall simpler than traditional systems, does not rely on external databases, and has great potential to improve as LM research progresses. Given these advantages, it seems important to continue exploring how to improve the performance of LM fact-checking system.

3 Data

The FEVER shared task is the standard metric for fact-verification, and it is the metric I will use in this experiment. Fact Extraction and VERification (FEVER) is a dataset that ”consists of 185,441 claims generated by altering sentences extracted from Wikipedia and subsequently verified without knowledge of the sentence they were derived from” [16]. Each statement has one of three labels: (1) Supported, (2) Refuted, and (3) Not Enough Info [16]. I will use this data to test how well the LM-only pipeline classifies each statement correctly.

4 Experiment Design

This experiment will be broken into three phases: (1) reproduction and expansion of the original experiment; (2) creation of a context-additive pipeline, and (3) introduction of a novel Q&A-based pipeline. Below I will describe each phase in turn.

4.1 Reproduction and Expansion

Unfortunately, the code for the original experiment was not released [7], so I will be recreating it from scratch. The original pipeline has three layers: (1) a masking layer, (2) a prediction layer, and (3) an entailment layer. The masking layer masks a named entity in a given claim, then feeds the masked claim to an LM. The LM then predicts the token using its embedded knowledge. Finally, the ”hypothesis” sentence and the original claim are fed into a textual entailment model for prediction - an entailed sentence would be considered ”Supported”.[3]

For example, given the claim ”Thomas Jefferson founded the University of Virginia”, the following steps would occur:

1. A Named Entity Recognizer (NER) would parse the sentence and extract the named entities.[4] The last Named Entity is masked with a given mask token. The result would be ”Thomas Jefferson founded the University of [MASK]”.[5]
2. The masked sentence is then fed to an off-the-shelf language model. The original experiment only used BERT [2], but I will expand this to BART [8], RoBERTa [9], and ALBERT [6]. The result is an ”evidence” sentence: ”Thomas Jefferson founded the University of Virginia”.

[2] A commonly used textual entailment model is the one from AllenNLP [3]. Indeed, this is the one I use in my own project.

[3] Note that, for the textual entailment layer, I will not use a custom Multi-Layer Perception (MLP); I will directly use the model provided by AllenNLP [3]. Although I do not use it, I do provide a method to create a custom MLP for the textual entailment layer. It is provided in my GitHub repository `https://github.com/jmcrey/cs584-final-project`.

[4] For this task I use the off-the-shelf spaCy model [5]

[5] Note that this pipeline only supports a single mask token. Furthermore, as with the original experiment, any named entity not found in the vocabulary will not be included in testing.

3. Finally, the system will present the original claim and the generated "evidence" to a textual entailment model[6], which outputs a label of "Supported", "Refuted" or "Not Enough Info".

Note that I will only use "frozen" models for this task. In other words, I will not fine-tune the off-the-shelf models during testing.[7] Denote the models as $BERT_{FREEZE}$, $BART_{FREEZE}$, $RoBERTa_{FREEZE}$, and $ALBERT_{FREEZE}$.

After this, I add a contextual layer to the pipeline.

4.2 Fact-Check with Context

1. A Named Entity Recognizer (NER) would parse the sentence and extract the named entities.[8] The last Named Entity is masked with a given mask token. The result would be "Thomas Jefferson founded the University of [MASK]".[9]
2. The masked sentence is then fed to an off-the-shelf language model. The original experiment only used BERT [2], but I will expand this to BART [8], RoBERTa [9], and ALBERT [6]. The result is an "evidence" sentence: "Thomas Jefferson founded the University of Virginia".
3. Finally, the system will present the original claim and the generated "evidence" to a textual entailment model[10], which outputs a label of "Supported", "Refuted" or "Not Enough Info".

5 Results

6 Conclusion and Future Work

References

[1] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions, April 2017. https://arxiv.org/abs/1704.00051.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, May 2019. https://arxiv.org/abs/1810.04805.

[3] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. Allennlp: A deep semantic natural language processing platform, May 2018. https://arxiv.org/abs/1803.07640.

[4] Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. Ukp-athene: Multi-sentence textual entailment for claim verification, May 2019. https://arxiv.org/abs/1809.01479.

[5] Montani Honnibal. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing, January 2017. https://sentometrics-research.com/publication/72/.

[6] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations, February 2020. https://arxiv.org/abs/1909.11942.

[7] Nayeon Lee, Belinda Z. Li, Sinong Wang, Wen tau Yih, Hao Ma, and Madian Khabsa. Language models as fact checkers?, June 2020. https://arxiv.org/abs/2006.04102.

[8] 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, October 2019. https://arxiv.org/abs/1910.13461.

[9] 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, July 2019. https://arxiv.org/abs/1907.11692.

[10] Luca Massarelli, Fabio Petroni, Aleksandra Piktus, Myle Ott, Tim Rocktasche, Vassilis Plachouras, Fabrizio Silvestri, and Sebastian Riedel. How decoding strategies affect the verifiability of generated text, September 2020. https://arxiv.org/abs/1911.03587.

[6] A commonly used textual entailment model is the one from AllenNLP [3]. Indeed, this is the one I use in my own project.

[7] Note that this is due to a resource limitation; I simply do not have the compute power or time to perform fine-tuning on each experiment model. However, I do fine-tune BART and GPT-2 for a later pipeline.

[8] For this task I use the off-the-shelf spaCy model [5]

[9] Note that this pipeline only supports a single mask token. Furthermore, as with the original experiment, any named entity not found in the vocabulary will not be included in testing.

[10] A commonly used textual entailment model is the one from AllenNLP [3]. Indeed, this is the one I use in my own project.

[11] U.S. Department of Homeland Security. Homeland threat assessment, October 2020. https://www.dhs.gov/sites/default/files/publications/2020_10_06_homeland-threat-assessment.pdf.

[12] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. Language models as knowledge bases?, August 2019. https://arxiv.org/abs/1909.01066.

[13] Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. How context affects language models' factual predictions, May 2020. https://arxiv.org/abs/2005.04611.

[14] 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, June 2020. https://arxiv.org/abs/1910.10683.

[15] Adam Robert, Colin Raffel, and Noam Shazeer. How much knowledge can you packinto the parameters of a language model?, October 2020. https://arxiv.org/abs/2002.08910.

[16] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Fever: a large-scale dataset for fact extraction and verification, December 2018. https://arxiv.org/abs/1803.05355.

[17] Yuwen Zhang and Zhaozhuo Xu. Bert for question answering on squad 2.0, March 2018. https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1194/reports/default/15848021.pdf.

[18] Zhixuan Zhou, Huankang Guan, Meghana Moorthy Bhat, and Justin Hsu. Fake news detection via nlp is vulnerable to adversarial attacks, January 2019. https://arxiv.org/abs/1901.09657.