CS584: FROM CLAIM TO QUESTION: FACT-VERIFICATION USING LANGUAGE MODELS

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

ABSTRACT

Many have shown that pre-trained Language Models (LMs) may store common-sense and factual information in their weights. Indeed, researchers proved that an unsupervised LM can answer "closed-book" questions with performance that rivals complex supervised systems. Given that background, some have explored whether LMs are able to replace existing fact-verification systems. Early results suggest that it may be possible to use LMs as fact-checkers; however, the LM pipeline struggled with claims that have little context and it returned biased results due to its training data. In this project, I suggest two potential improvements to overcome these bottlenecks. First, I suggest adding context to the claim using a smartly trained autoregressive LM, which helps overcome claims with little context. Then, I propose translating the claim into a question to combat an LM's bias from training. Although not as performant, I believe that further research can push these pipelines to be the state of the art.

1 Introduction

Pre-trained Language Models (LMs) are central to many downstream Natural Language Processing (NLP) tasks. State-of-the-art models such as BERT [2], RoBERTa [9], and T5 [15] have created the possibility for LM-only question answering, summarization, and commonsense reasoning systems. In fact, researchers have designed LMs capable of answering natural and trivia questions without additional context [15]. However, until recently, it was not thought that LMs 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 a given statement is supported, refuted or that there is not enough information (NEI). In recent years, it has become increasingly important to solve this problem due to the ability to quickly generate convincing misinformation using these pre-trained LMs [21]. In fact, misinformation is a top national security threat for the United States [11].

To help mitigate this problem, researchers have developed automated fact-verification systems [4]. However, these systems are often complex, computationally expensive and rely on manual updates to external knowledge bases [7]. For example, a typical system performs the following procedures: (1) gather relevant documents from an external knowledge base, (2) extract evidences from the documents, and (3) classify the statement using the evidences [7]. Given the complexities of this process, researchers have been looking for a simpler approach to fact-verification. To this end, recent research has suggested that an LM-only pipeline may be possible.

For example, Lee et al. [7] built a promising LM-only fact-verification pipeline that is simpler to implement and not reliant on an external knowledge base [7]. They proposed a three (3) layer pipeline to predict whether a given claim is supported, refuted or NEI. This pipeline was able to achieve standard benchmarks on the FEVER [19] dataset after fine-tuning. However, the system struggled to fill-in claims with little context, could not avoid bias from its training data, and floundered with classifying statements as refuted or NEI.

This project aims to extend this research by exploring the effectiveness of other LMs in this pipeline, and by suggesting novel improvements to mitigate the aforementioned bottlenecks. To be more specific, I recreate Lee et al's [7] pipeline and measure the performance of off-the-shelf BERT [2], BART [8], RoBERTa [9] and ALBERT [6] LMs. Then, I suggest three modifications to their pipeline that help solve key problems posed in the original paper:

1. ***Context Generation***: First, I suggest adding an autoregressive LM as a "context generation" layer. Drawing inspiration from Petroni et al. [13], the goal is to provide more context about the claim presented to the fact-verification system. As was shown with question answering (Q&A) systems, giving more context to an LM may improve its ability to accurately answer a question [13]. I explore whether this same principal holds true with fact-verification.
2. ***From Claim to Question***: Second, I suggest translating the "claim" into a "question" using a pre-selected "answer". Q&A models have proven to be quite effective at answering natural and trivia questions [18]. Moreover, a targeted question inherently adds more context to the desired data, which may help avoid biases from the LM's training data. As such, I suggest generating a question from the claim with the "answer" being the masked token. Then, a model fine-tuned to answer closed-book questions can be used for fact-verification.
3. ***Question-Context Generation***: Lastly, I suggest a combination of the two above approaches. First, a question should be generated using the same approach above. Then, as in the first suggestion, an autoregressive LM should generate context based on the claim. Finally, an extractive Q&A model can "select" the answer from the generated text.

Most importantly, none of the suggested approaches leverage an external knowledge base; that is, they retain the key improvements proposed by Lee et al [7]. Furthermore, the latter solutions solve the following three problems.

First, the Q&A-based pipelines inherently add more context to a claim. Second, they are better at avoiding the semantic bias in their training data, and (3) they produce much more balanced results. By translating the claim into a question, the Q&A approaches better understand the desired data, thus adding context and avoiding semantics in the training bias. Intuitively, answering the question "What year was Tim Roth born in?" has more context than "Tim Roth was born in [MASK]". Further, the question avoids the semantic bias of "[PERSON] was born in [PLACE]", which is commonly found in training data [7]. Lastly, the Q&A systems achieve more balanced results across the classes, and, thus, can better classify claims as refuted or NEI. Hence, I hypothesize that these methods may allow LM-only pipelines to better perform fact-verification tasks with further research.

In the following sections, I will provide background on both question answering and fact-verification systems. Then, I will present the design of the experiments, including the data and metrics used to measure the outcomes. Lastly, I describe the outcomes of those experiments, and my thoughts for future work.

2 Background

2.1 Question Answering

Question answering (Q&A) is the task of answering questions using LMs. One of the most popular variants is "extractive Q&A", where an LM extracts the answer from a given context. Research shows that LMs are quite effective at such tasks, and there are many open-source models available to accomplish these tasks [9, 2, 20]. As proof, off-the-shelf models, like BERT [2] and RoBERTa [9], exhibit state-of-the-art results in extractive Q&A tasks on the SQuAD 2.0 dataset [20, 17].

A separate yet related task is open-domain Q&A, where a model must answer an open-domain question without being given explicit context. Although context is not given, it is often assumed that the model has access to an external knowledge base so that it can "find" relevant context and the answer. [18]. Cleverly, Roberts et al. [18] likened this to an "open-book" exam, where the model is able to extract the answer from a "book" of knowledge [18]. Indeed, DrQA, an RNN-based system, achieved state-of-the-art results on the original SQuaD dataset using this approach [1, 16].

Until recently, LMs with access to external knowledge sources severely out-performed those that did not have such access. However, it is known that pre-trained Language Models (LMs) hold large quantities of relational data in their weights [12]. Today, an LM-only pipeline can perform as well as complex supervised systems in "closed-book" style Q&A tasks [18]. 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 questions; they answer solely on the knowledge retained in their weights.

A different, yet related field of study is fact-verification.

2.2 Fact-Verification

Fact-verification is the task of validating the truthfulness of a given claim. Given a factual claim, the system will classify it as supported, refuted, or not enough information (NEI). Like open-domain Q&A, these systems rely on document retrieval, sentence extraction and textual entailment [4]:

1. ***Document Retrieval***: An LM extracts relevant documents from a pre-defined knowledge base (e.g. Wikipedia).[1]
2. ***Document Extraction***: From the extracted documents, a separate LM will select sentences that are relevant to the given claim ("evidence").
3. ***Textual Entailment***: Finally, the system will present the collected "evidence" and the given claim to a textual entailment model[2], which outputs a label of "Supported", "Refuted" or "Not Enough Info".

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

Moreover, traditional open-domain Q&A and fact-verification systems have two key disadvantages. First, they rely on an external knowledge base which must be manually updated to ensure accuracy. Secondly, they are computationally expensive to implement and run due to their need to search large amounts of data. As such, researchers have been searching for a simpler approach to fact-verification. Fortunately, there has been promising research on an intuitive, 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. Lee et al [7] proposed the following design: (1) mask a named entity in a given statement, (2) use an LM to fill the mask, and (3) use textual entailment to classify the claim. The main idea is to exploit knowledge in an LM so that it fills the mask with the "factual" named entity. If the filled-in mask entails the original claim, then the claim is considered supported.

Although not as performant as traditional systems, a fine-tuned BERT [2] fact-checking system was able to pass standard benchmarks on the FEVER dataset [7, 19]. Compared to traditional systems, this LM approach to fact-checking is overall simpler, does not rely on external databases, and has great potential to improve as LM research progresses.

However, there were two key problems in the original system. First, it struggled to correctly classify claims that did not have much context [7]. For example, the FEVER dataset [19] is rich with short claims in the structure of "[OBJECT] is a [ENTITY]". These claims do not have much context about the desired output, so the LM struggled to correctly predict the masked token.

Second, the system struggled to overcome bias from the LM's training data. In Lee et al's [7] experiment, they used a BERT [2] model trained on Wikipedia. Note that Wikipedia often structures biographical information as "[PERSON] (born [YEAR])" and, later, "[PERSON] was born in [PLACE]" [7]. Thus, when presented with a claim of "[PERSON] was born in [MASK]", the LM often outputs a location instead of a date. This and other biases influenced the model's ability to correctly predict short claims. Likely, these two problems led to the models' inability discern refuted and NEI claims from supported ones.

So, the proposed pipelines aim to overcome these bottlenecks. The following section presents the structure of the experiment.

3 Experiment Design

This experiment will be broken into three phases: (1) reproduction and expansion of Lee et al's [7] experiment; (2) addition of a context generation layer, and (3) introduction of novel Q&A-based pipelines. Below I describe each proposition.

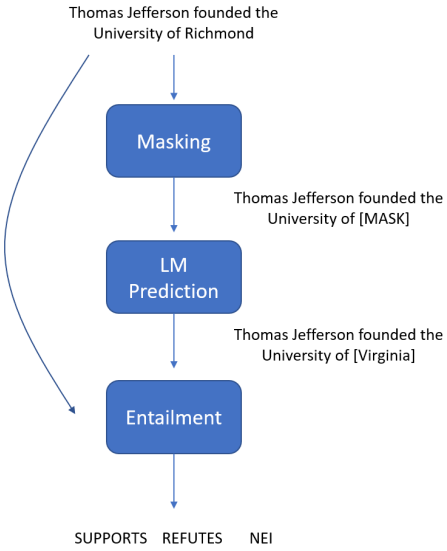
3.1 Reproduction and Expansion

The first phase of this project is to recreate the results from Lee et al's [7] experiment.[3] The original pipeline has three layers: (1) a masking layer, (2) a prediction layer, and (3) an entailment layer. For example, given the claim "Thomas Jefferson founded the University of Virginia", the following steps would occur:

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

[2] A commonly used textual entailment model provided by AllenNLP [3].

[3] Unfortunately, the code for their experiment was not released, so I will recreate it from scratch.



1. ***Named Entity Masking***: A Named Entity Recognizer (NER) parses the claim and extracts the named entities.[4]The last named entity is replaced with the model's mask token.[5]A potential result is "Thomas Jefferson founded the University of [MASK]".
2. ***Mask Filling***: The masked sentence is given to an LM to predict the factual token. Specifically, it uses the transformer as a "masked language model" (MLM) to predict the masked token, which is the same technique used to train many transformers [2, 8]. The result is "evidence": "Thomas Jefferson founded the University of [Virginia]".
3. ***Textual Entailment***: Lastly, a textual entailment model will be presented with both the original claim and the "evidence".[6]The output is 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 chosen LMs during testing.[7] The original experiment only used BERT [2], but I will expand this to BART [8], RoBERTa [9], and ALBERT [6]. Denote the models as $BERT_{FREEZE}$, $BART_{FREEZE}$, $RoBERTa_{FREEZE}$, and $ALBERT_{FREEZE}$.

After recreation, I add context generation layer to the pipeline.

3.2 Contextual Fact-Checking

I propose adding a "context generation" layer to the pipeline. As demonstrated by Massarelli et al [10], training an autoregressive LM on a factual dataset leads to more factual text generation. Moreover, a problem with the above pipeline was that many claims had little context. So, I will add a GPT-2 model fine-tuned on the CC-NEWS dataset to the pipeline so that short claims have more context.[8] For this system, I tested both $BART_{FREEZE}$ [8] and a partially fine-tuned $BART$ [8] model, denoted as $BART_{FINETUNE}$.

Concretely, this system performs the following tasks.

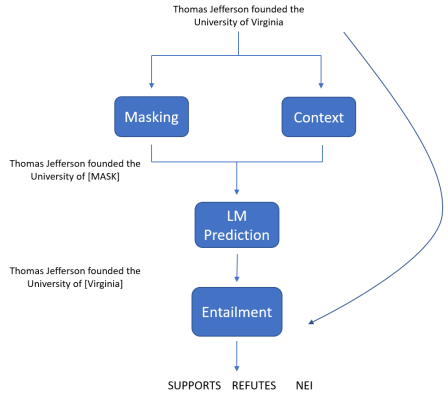
[4]For this task I use spaCy, an open-source NER model [5]

[5]Only a single-token named entity can be masked in this pipeline. Also, as with the original experiment, any named entity not found in the vocabulary will not be included in testing

[6]For the textual entailment layer, Lee et al [7] used a custom MLP for final verification. I will not use a custom Multi-Layer Perception (MLP); I will directly use the model provided by AllenNLP [3]. Although not used, I do provide a method to create a custom MLP for the textual entailment layer in my GitHub repository `https://github.com/jmcrey/cs584-final-project`.

[7]This is due to a resource limitation; I do not have the time or computational ability to fine-tune each model. However, I do fine-tune BART [8] and GPT-2 [14] for a later pipeline.

[8]In this case, GPT-2 was only fine-tuned on a partial dataset - 150,000 samples from the CC News dataset. For detail on loss and perplexity, please see my GitHub `https://github.com/jmcrey/cs584-final-project/tree/master/models`.



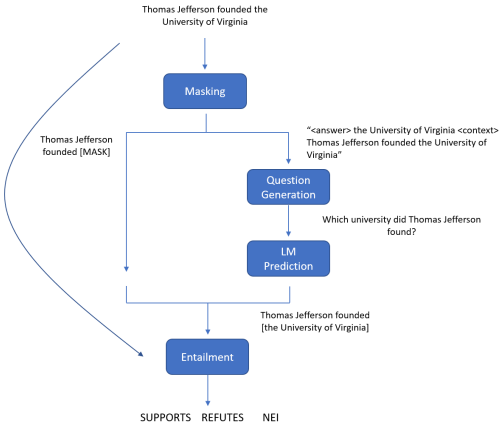
1. ***Named Entity Masking***: See above.
2. ***Context Generation***: A fine-tuned autoregressive LM, such as GPT-2 [14], generates context on the given claim. The masked sentence and the generated context are concatenated together for prediction.
3. ***Mask Filling***: The LM predicts the masked token based on the given sentence and context. Using that prediction, the masked sentence is filled and the "evidence" is extracted.
4. ***Textual Entailment***: Finally, the original claim and the extracted "evidence" is presented to a textual entailment model for final classification. Note that "evidence" is only the filled-in claim; it is not the claim with context.[9]

The goal of this pipeline is to provide the LM with more context about the given claim. Ideally, this would influence it to more accurately predict the masked token. Unfortunately, as will be discussed, the LM does not take advantage of the generated context. However, this lack of success inspired a new pipeline - one that transitions from pure claim verification to a Q&A-based pipeline.

3.3 Q&A-based Fact Verification

Again, Q&A tasks and claim-verification tasks are quite similar. However, questions have the advantage of being targeted - they ask a model for a specific piece of information. Moreover, the most recent Q&A models do not rely on external databases; they leverage the knowledge retained in their weights. Given this background, I propose the following new pipelines.

Closed-Book Fact-Verification



1. ***Named Entity Extraction***: Using an NER, extract a named entity and mark it as the "answer". Then, mask the chosen entity in the original claim. For example, it may extract "University of Virginia", then mask the claim like, "Thomas Jefferson founded the [MASK]".[10]
2. ***Question Generation***: Using a fine-tuned LM, generate a question with the "answer" being the named entity.[11]For example, the LM will take an input of "answer: University of Virginia context: Thomas Jefferson founded the University of Virginia" and it will output a question like "Thomas Jefferson founded which university?".
3. ***Closed-Book Q&A***: Input the generated question into a model trained to answer "closed-book" questions. Then use the output of the model ("answer") to fill-in the masked claim ("evidence").
4. ***Textual Entailment***: As before, input the claim and "evidence" into an entailment model for final verification.

[9]Although this may be worth exploring, in my experiments it negatively impacted the results.

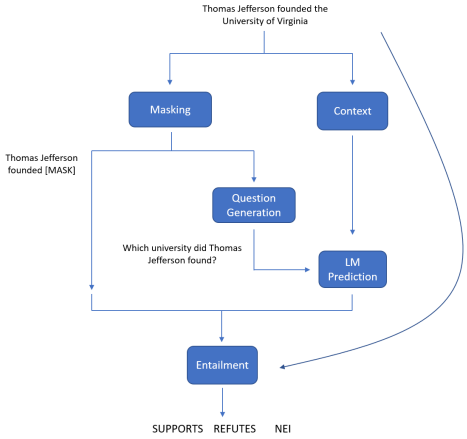
[10]This pipeline supports multi-token claims.

[11]I used the following model, which is freely available on Hugging Face `https://huggingface.co/mrm8488/t5-base-finetuned-question-generation-ap`.

In my experiment, I use a T5 model fine-tuned on natural question answering [18].[12] Denote this model as $T5_{SSM-NQ}$. Note that this model is not the best performing model; it only achieves an average of 29% accuracy on answering natural and trivia questions.

Open-Book Fact Verification

In general, this pipeline is the same as above, with only two differences.



1. ***Named Entity Extraction***: See above.
2. ***Question Generation***: See above.
3. ***Context Generation***: As with the Contextual Fact-Checking pipeline, I propose adding an autoregressive LM to generate context on the given claim. This will be used as the "context" in an off-the-shelf extractive Q&A system.
4. ***Open-Book Q&A*** Using an extractive Q&A model, input the question and the generated "context" such that the model extracts an "answer."[13]As before, the pipeline will fill in the masked claim using the extracted answer ("evidence").
5. ***Textual Entailment*** Input the claim and "evidence" into an entailment model for final verification.

For context generation, I use the same GPT-2 [14] fine-tuned on CC-NEWS. For open-book Q&A, I use a distilled BERT model fine-tuned on SQuAD [16]. Denote the distilled BERT model as $DISTILBERT_{SQUAD}$.

The key difference between the two pipelines is that the "knowledge" is coming from different places. The first system directly uses knowledge in an LM to answer closed-book questions, while the second leverages the factual generation of an autoregressive LM along with the reading comprehension of a transfomer. As discussed later, this difference plays a key role in the model's ability to answer questions.

Still, these hold three key advantages over the original pipeline. Foremost, question generation better avoids the bias in the training data and the written form of the claim. Note that most of the frozen models are trained on Wikipedia; so, if a model is presented with "Tim Roth was born in [MASK]", the model would likely guess London rather than 1961. On the other hand, if presented with a targeted question like "What year was Tim Roth born in?", the model will better understand the desired data and it is more likely to predict the original token of 1961. Hence, this model is better suited to avoid bias from its training data.

Similarly, these models naturally provide more context about the desired data. FEVER [19], in particular, holds many short claims which have a structure of "ENTITY is a ..." [7]. Models struggle to predict the correct token because (1) there is little context about the desired data and (2) this type of phrase is common, especially in Wikipedia [7]. Contrarily, asking a direct question like "What year was Tim Roth born in?" gives much needed context about the desired data. So, even in very short claims, the model will have more context about a claim, and, thus, be able to better predict the next token.

Lastly, these approaches exploit the rich research into LM-only Q&A. LMs accurately answering questions is a much more active research field than LM-only fact-verification. Consequently, this pipeline is likely to improve much faster because it is using the same techniques applied to Q&A.

3.4 Data & Metrics

The FEVER shared task [19] is the standard metric for fact-verification, and it is the metric I use in this experiment.[14] 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" [19].

[12]The model I used can be found here `https://huggingface.co/google/t5-small-ssm-nq`. I would have used a larger model, but my system could not handle the load.

[13]The model I used can be found here `https://huggingface.co/distilbert-base-cased-distilled-squad`.

[14]Specifically, I used the paper_test segment of the FEVER dataset [19]

Each statement has one of three labels: (1) Supported, (2) Refuted, and (3) Not Enough Info [19]. I will use this data to test how well the LM-only pipeline classifies each statement correctly.

Lee et al [7] reported precision, recall, F1 and accuracy, so I replicate these metrics. Notably, the FEVER shared task [19] has a stricter scoring mechanism where they measure not only accuracy, but the evidence captured by the document retrieval system. Since this system does not leverage document retrieval, I cannot use traditional FEVER metrics. Also note that FEVER [19] is a balanced dataset, so a random baseline would yield an accuracy of 33%.

The following section will introduce the results of my experiments.

4 Results

4.1 Reproduction and Expansion

In short, I was able to successfully reproduce the results of the original experiment using off-the-shelf LMs. Below I show the metrics of each model, $BERT_{FREEZE}$, $BART_{FREEZE}$, $RoBERTa_{FREEZE}$, and $ALBERT_{FREEZE}$.

Table 1: Frozen Models

Model	Label	prec	recall	f1	accuracy	macro prec	macro recall	macro f1
$BERT_{FB}$	SUPPORTS	0.43	0.09	0.15	0.38	0.39	0.38	0.33
	REFUTES	0.36	0.69	0.47				
	NEI	0.39	0.35	0.37				
$BERT_{FREEZE}$	SUPPORTS	0.40	0.85	0.54	0.38	0.36	0.35	0.27
	REFUTES	0.43	0.05	0.10				
	NEI	0.25	0.13	0.17				
$BART_{FREEZE}$	SUPPORTS	0.39	0.91	0.55	0.38	0.33	0.34	0.23
	REFUTES	0.40	0.03	0.06				
	NEI	0.20	0.06	0.09				
$RoBERTa_{FREEZE}$	SUPPORTS	0.39	0.86	0.54	0.38	0.33	0.34	0.26
	REFUTES	0.38	0.07	0.12				
	NEI	0.21	0.08	0.12				
$ALBERT_{FREEZE}$	SUPPORTS	0.36	0.85	0.50	0.35	0.35	0.34	0.25
	REFUTES	0.49	0.07	0.13				
	NEI	0.20	0.10	0.13				
Shared-Task Baseline [19]	-	-	-	-	**0.49**	-	-	-

As shown, the results are quite similar to the original experiment. Interestingly, almost all of the frozen models receive an accuracy of 38%, which is above a random baseline. Note that all of these models were trained on different data: $BERT_{FREEZE}$ and $BART_{FREEZE}$ were trained on Wikipedia; $RoBERTa_{FREEZE}$ on Bookcorpus, Wikipedia and CC-NEWS; and $ALBERT_{FREEZE}$ on Bookcorpus and Wikipedia. Despite these differences, they all score approximately the same on the test set, with $ALBERT_{FREEZE}$ being the lowest at 35%.

In the following sections, I only use $BART_{FREEZE}$ because it has the closest absolute scores to $BERT_{FREEZE}$, and $BERT_{FREEZE}$ had already been explored by Lee et al [7].

4.2 Contextual Fact-Checking

In this experiment, I fine-tuned GPT-2 [14] on 150,000 samples of the CC-NEWS database, and I fine-tuned BART [8] ($BART_{FINETUNE}$) on 150,000 samples of the FEVER train dataset [19]. After fine-tuning, I used GPT-2 to generate the context on each claim, then used $BART_{FINETUNE}$ to predict the masked token. I also used $BART_{FREEZE}$ to predict the masked token for comparison purposes.

The results indicate that generated context from an autoregressive LM do not help nor hurt the outcomes. Indeed, even after fine-tuning both GPT-2 and BART there was a meager performance boost. On closer analysis of the generated context and predictions, it appears that context did not have much influence on the prediction of the LM.

For example, Table 3 displays the text generated about the United Nations. Even though the context was decent, the prediction of the LM was not influenced. Indeed, the prediction was "Olympics", the same as the original prediction. However, this was not the case with the Question-based pipeline.

Table 2: Models with Generated Context

Model	Label	prec	recall	f1	accuracy	macro prec	macro recall	macro f1
$BART_{FREEZE}$	SUPPORTS	0.40	0.83	0.53	0.37	0.31	0.33	0.27
	REFUTES	0.34	0.08	0.13				
	NEI	0.21	0.10	0.13				
$BART_{FINETUNE}$	SUPPORTS	0.40	0.80	0.53	0.37	0.33	0.34	0.28
	REFUTES	0.37	0.11	0.17				
	NEI	0.22	0.11	0.15				
Shared-Task Baseline [19]	-	-	-	-	**0.49**	-	-	-

Table 3: Context Generation Examples

Claim	Context	Target	Prediction
"Ukrainian Soviet Socialist Republic was a founding participant of the UN."	*"The UN has been a major source of funding for the UN and has been instrumental in the creation of the UN Security Council. The UN has been a major source of funding"*	UN	Olympics

4.3 Question-based Fact-Checking

Table 4: Q&A Based Fact-Checking Models

Model	Label	prec	recall	f1	accuracy	macro prec	macro recall	macro f1
$T5_{SSMNQ}$	SUPPORTS	0.39	0.56	0.46	0.34	0.32	0.33	0.31
	REFUTES	0.35	0.19	0.24				
	NEI	0.22	0.23	0.23				
$DISTILBERT_{SQUAD}$	SUPPORTS	0.38	0.52	0.44	0.35	0.32	0.32	0.31
	REFUTES	0.35	0.34	0.34				
	NEI	0.22	0.11	0.15				
Shared-Task Baseline [19]	-	-	-	-	**0.49**	-	-	-

Although the metrics show that the model did not achieve overall improved accuracy, they hold some key insights. To start, the precision, recall and f1 scores across the classes are much more balanced here than with the "traditional" pipelines. In particular, the T5 model displays the most balanced performance, with the best recall and f1 score on the refutes and NEI classes.

Furthermore, these pipelines achieved almost the same accuracy with less than state-of-the-art models. For example, $T5_{SSM-NQ}$ achieved only a 25.7% accuracy on the original Natural Question (NQ) dataset [18], but it is achieving 34% accuracy on the FEVER dataset [19] without any fine-tuning. This begs the question on whether the state-of-the-art $T5.1.1 - XXL + SSM$ would be able to achieve greater accuracy.[15]

Finally, analysis of the pipelines shows that they produce potentially better answers to the question. For example, when GPT-2 [14] produces decent context, Table 5 shows that the $BART_{FINETUNE}$ model was unable to predict the correct token. However, $DISTILBERT_{SQUAD}$ model is able to extract the "correct" answer to the proposed question using GPT-2 [14] generated context.

Hence, these two pipelines seem to have increased potential to tackle the problem of LM-only fact-verification.

5 Conclusion and Future Work

This project set out to explore off-the-shelf models' ability to fact-check, and to create a pipeline to help overcome the bottlenecks of the Lee et al's experiment [7]. First, I recreated the pipeline to show that off-the-shelf LMs consistently achieve better scoring than a random baseline. Indeed, every model achieves almost 38% accuracy which is above a 33% random baseline.

[15] I would have used this model in this project, but my system could not handle the computational load.

Table 5: Q&A Examples

Pipeline	Claim	Question	Target	Prediction
Closed-Book	*"Ukrainian Soviet Socialist Republic was a founding participant of the UN."*	*"What organization was the Ukrainian Soviet Socialist Republic a founding member of?"*	UN	the Ukrainian SSR
Open-book	*"Ukrainian Soviet Socialist Republic was a founding participant of the UN."*	*"What organization was the Ukrainian Soviet Socialist Republic a founding member of?"*	UN	UN

Second, I proposed three possible LM-only pipelines to solve problems encountered in the original expeirment.

- First, to add context generation using a smartly trained autoregressive LM so that the model can better understand the claim and generate more factual predictions.
- Next, to translate the claim into a question so that the model inherently receives more context about the desired token and help it overcome bias in its training data.
- Finally, a combination of both where, (1) the claim is translated to a question, and (2) context about the claim is generated using a LM. While the former approach relies on a single LM, this leverages knowledge from both context generation and reading comprehension. Indeed, research has shown that targeted training can increase an LM's ability to generate factual information [10], and, paired with an extractive model, can be used to increase open-domain Q&A results [13].

Notably, each approach holds advantages over traditional fact-verification systems because they are overall simpler, less computationally expensive, and do not rely on an external knowledge base. Moreover, they have the ability to improve as research into LMs continues. In particular, the Q&A-based pipelines can exploit the rich research into question answering systems. So, if LMs can achieve state-of-the-art results on a closed book Q&A test, then these can achieve the same with fact-verification.

Lastly, I believe that further research may be able to improve the performance of these pipelines. At first glance, the results of this experiment seem to indicate that LMs are unable to perform fact-verification. However, all the models in the first experiment achieved a better-than-random score in the FEVER shared task without fine-tuning. Similarly, the $T5_{SSMNQ}$ Q&A system did not use a state-of-the-art model, yet it was able to perform better on this task than its original [18]. Finally, none of these pipelines used fully tailored models; the models were pulled off-the-shelf with little to no fine-tuning. Hence, I believe that with further research and fine-tuning, these pipelines may be able to achieve the state-of-the-art.

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. Ukpathene: 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.

[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] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, February 2019. https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf.

[15] 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.

[16] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text, October 2016.

[17] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad, June 2018. https://arxiv.org/abs/1806.03822.

[18] 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.

[19] 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.

[20] 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.

[21] 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.