**Determining Factual Accuracy of AI-generated Text**

CS 697R - GRADUATE SPECIAL PROJECTS

Brigham Young University

Trevor Andrus, Quinn Snell

Spring / Summer 2023

**Abstract**

With the advent of large language models such as Open Ai’s chatGPT, AI-generated text is becoming more common in both academic and professional spheres. However, while the grammatical accuracy of the models is readily apparent, it seems the factual accuracy of the information from these models is often left unchecked. This paper seeks to remedy that apparent shortcoming. We propose a framework for testing the factual accuracy of AI text, demonstrate the shortcomings of current sources to satisfy the framework, and in the meantime present an alternative solution.

We show that our proposed framework falls short because the consistent and accurate extraction of subject-object pairs from sentences has not yet been solved. We further show that even with the creation of knowledge graphs based on these subject object pairs, current knowledge bases are structured to identify only a finite number of relations between entities. This may be useful for identifying specific relationships but cannot currently be generalized to an arbitrary number of relations – which will be required to establish a comprehensive method for determining factual accuracy.

Despite this, we also propose a method of zero-shot learning that may give insight into the current state of LLM self-reflection. The problem of determining factual accuracy requires a massive amount of generalization, and the current open-source-free-to-use leader of general AI (at least in the natural language space) is ChatGPT. Using ChatGPT, we show a method of self-analysis on its own factual accuracy, and comment on potential pitfalls of this method.

**Introduction**

In an era dominated by the rapid evolution of technology, the surge in popularity of ChatGPT stands as a testament to the world’s growing fascination with advanced artificial intelligence. This large language model carries immense potential to revolutionize communication, research, and creativity. Sophisticated language models such as these have shown their ability to mimic human-like text with remarkable accuracy, blurring the line between authentic and fabricated information. However, they also come with the inadvertent potential of disseminating misinformation. In this paper we explore methods to mitigate the spread of this misinformation. We propose a framework with which one might programmatically determine the accuracy of AI-generated text. However, we also show the shortcomings of current technology that make our initially proposed framework infeasible. We state our findings and identify areas of further study to be pursued.

**Proposed Framework**

Our original framework (while not currently feasible with available resources) is as follows:

1. Split input text into individual sentences
2. Using a part of speech (POS) tagger, identify the parts of speech in each sentence
3. Using these POS-tagged sentences, extract the subject and object
4. Using dependency parsing, infer the relationships between these subject object pairs
5. Create knowledge graphs from the subject-object-relationship triples
6. Compare knowledge graphs against established knowledge base repositories (such as DBPedia)

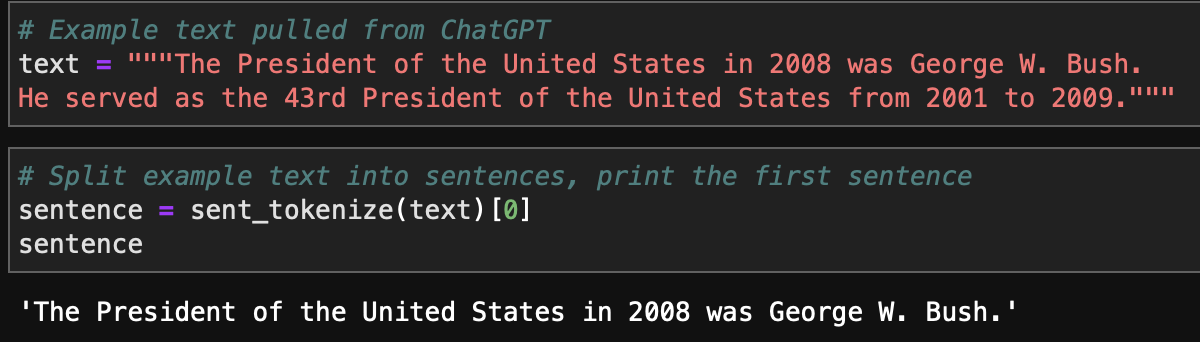
However, we’ve identified issues with steps 3, 4, and 6 of this process. Details of these issues will be described below.

**Framework exploration**

To illustrate our findings, this next section will be a commentary on our step-by-step process in attempting to implement our framework. The paper will address the steps that worked according to plan, and the steps that presented shortcomings, as well as the attempts we made to overcome them.

1. **Split input text into individual sentences**

* This step is straightforward, and solutions to the sentence-splitting problem are abundant and well demonstrated.
* For our tests, we used the NLTK sent\_tokenize function as shown below. (The sentence from the below example will serve as our test sentence for all steps in the framework).



1. **Using a part of speech (POS) tagger, identify the parts of speech in a given input text**.

* This step of the framework was also straightforward. Part of speech (POS) tagging and Named entity recognition (NER) have common and widely available programmatic solutions in the open source.
* We continued using the Python package NLTK to perform part of speech tagging and used the Python package Spacy to perform named entity recognition. (A full list of packages used can be found in the code provided with the paper submission). The result of the POS and NER tagging can be found below.

A screen shot of a computer program

Description automatically generated

* At this point, we had defined a simple programmatic pipeline to extract the linguistic elements from sentences

1. **Using these POS-tagged sentences, extract the subject and object of each sentence**

* Up until this step, our framework operated as planned. However, when looking deeper into the issue of solving for subject-object pairs within sentences, we found that the solution is not trivial, and in fact, there doesn’t seem to be a readily available solution that can generalize to solve for all subject-object pairs.
* Within the English language, each subject-object relation is defined by a specific set of rules. Using the part of speech tagger, we can hypothetically extract ‘nsubj’ tagged words as noun subjects, and ‘nobj’ tagged words as noun objects. However, the subjects and objects of sentences are not always classified in this black and white manner (some sentences have no objects, and some have multiple). This is not to mention the difference between active and passive voice, where subjects and objects can appear in different orders. As such, a generalized program to simply extract subjects and objects from a sentence of any format is not currently available. However, researchers at Stanford have sought to identify these relationships in specific cases with their CoreNLP tool. Using this tool, one can extract subjects and objects related through “Live\_In, Located\_In, OrgBased\_In, and Work\_For” subject relations. <https://nlp.stanford.edu/software/relationExtractor.html>
* Because of the complexity of the subject object extraction, we elected to verify the rest of our framework, with the possibility of overcoming the obstacle in the future.

1. **Using dependency parsing, infer the relationships between these subjects and objects**

* In this step, our problem in step 3 continued to cause issues. While hypothetically one may be able to determine a relationship between subject and object through dependency parsing (for example looking a dependency tree), the possible part of speech subject-object pairs (in varying orders and quantities) cannot feasibly be accounted for in the scope of this project. Rule based code would need to be written to analyze sentences for all possible combinations of subjects and objects, then implement dependency parsing to determine the relationships between the identified pairs. Currently there is no generalized solution to this problem, and we therefore found these two steps to be an obstacle in our proposed framework. We believe, however, that given advances in natural language processing to derive these subject-object pairs, our framework may be feasible.

1. **Create knowledge graphs from the subject-object-relationship pairs**

* Given the difficulties encountered in the previous two steps of the framework, we did not explore this step in detail. The task of creating knowledge graphs, however, is well documented in open-source spaces. Variation comes in the storage of knowledge graphs, but creating the data structure itself has been well proven. In our use case, we would construct a simple graph – each node being an entity and each edge being a relation, with simple string labels for each. After the construction of the initial knowledge graph, we would only need to convert open-source knowledge graphs from databases to our format for comparison. Examples of knowledge graph generation in the open source can be found [here](https://towardsdatascience.com/nlp-with-python-knowledge-graph-12b93146a458#:~:text=In%20order%20to%20build%20a,Cython%20(C%2BPython).).

1. **Compare knowledge graphs against established knowledge base repositories (such as DBPedia)**

* This step served as the second major setback in the realization of our framework – not only from a theoretical perspective, but also from an efficiency perspective. The structure of DBPedia (a leading knowledge graph repository) is refined to only a finite set of relations between entities. This means that only a specified number of subject-object relations can be identified and verified using this database. Take for example, our test sentence: “George Bush was the president of the United States in 2008.”. DBPedia features an entity for George Bush, but the relations associated with this entity are ‘occupation’, ‘title’, ‘birthdate’ and so on. The generic use of the ‘is / are’ verb in the English language makes the comparison against these relations difficult. In order to compare this fairly simple sentence against DBPedia, we would need to derive a way to translate ‘was’ in our test sentence to ‘occupation’ somehow. Again, this calls back to the subject-object relation issue we’ve discussed earlier. The REAL relation between George Bush and the President of the United States is ‘occupation’ or ‘office’, which cannot easily be determined from the example sentence, and furthermore cannot be generalized to all sentence types. We created a programmatic way to query DBPedia for all listed relations and attempted to do fuzzy matching between example knowledge graphs and the returned relations, but this solution turned out to be inaccurate and non-performant. Some example relation labels and values queried from DBPedia are shown below – more details can be found in the attached code.

**A computer screen with text and numbers

Description automatically generated**

**A computer screen with white text

Description automatically generated**

**Exploring alternatives**

Feeling unsatisfied from the unsuccessful attempt to employ our framework, we turned to alternative methods for determining subject object relations and determining the truthfulness of text in general.

We first looked at the possibility of using other language models to extract information for us. Providing our test sentence as reference, we experimented with GPT2 and a version of Distil-BERT to extract potential truth determinations. GPT2 seemed to lack in coherency – while it performed well at text generation, it lacked the ability to answer questions in a coherent manner. A version of Distil-BERT that was trained on the squad dataset was used to test the potential of a question-answer model for deriving truth. In these cases, we asked the model to compare our test sentence with the abstract of each entity available on Wikipedia.

An example question asked to the model was “Given this input text, was George Bush the president in 2008?”. We then provided the model with an abstract of the. Wikipedia article for George Bush.

However, when making accuracy determinations, this model presented challenges. Question and answer models only perform well when the answer to questions are found directly in reference text. They are incapable of any sort of logic or reasoning. Because of this, DistilBert cannot answer yes or no questions, and can only responds with snippits from reference text. For example, the range of 2001-2009 was mentioned in our provided abstract, but no mention of the specific year 2008 appeared. As a result, answers to questions about the year 2008 are erratic and inconsistent. (We will see later that this issue persists through later versions of GPT as well). Some examples are shown below.

A screenshot of a computer

Description automatically generated

We see that specific questions mentioned in the text are answered accurately, but any reasoning questions (including. Yes or no questions) result in an unpredictable answer.

We then experimented with the idea of using chatGPT to analyze its own text and saw some success. Using this zero-shot method, we supply the model with its own text and ask it to look at it objectively. However, a common issue that permeated the GPT tests was inconsistent reporting of time ranges. See the example below.

A screenshot of a phone

Description automatically generated

We see that ChatGPT makes a better attempt at reasoning than DistilBert, but despite its more coherent answer, it still reports incorrect information. We see it immediately contradict itself – confusing the range of years 2001-2009 with the year 2008. This problem persisted even when supplying wiki abstracts as reference – it seems the algorithm cannot extract that the year 2008 falls between 2001 and 2009.

Looking at solving the previously raised subject-object issue, however, we have been able to see potential for zero-shot efficacy. Like was previously stated, there is no algorithm currently fit to extract the subject-object pairs from an English sentence (without that sentence being of a very specific structure). ChatGPT, however, has shown sophisticated responses that indicate the potential of solving the issue. See below for examples.

A screenshot of a computer

Description automatically generated

These successes have shown that there may be other methodologies that could prove more successful than our originally proposed framework. We intend to pursue this course of action going forward. However, there are still complicated issues that could result from zero shot methods such as these. When asking the model to check its own accuracy directly, for example, we can stack the inaccuracies.

Using the ChatGPT API, we plan to continue to experiment with the zero-shot framework to find ways to consistently determine the factual accuracy of AI generated text.

**Conclusion / Next Steps**

As we have shown, our originally proposed framework is not currently feasible. The problem of extracting subject-object pairs and relations from text, and the structure of knowledge bases for comparing knowledge graphs currently inhibits our original plan. However, we have also seen success with using LLM’s to self-analyze. We are currently looking to find breakthroughs with this zero-shot method. If we find a way to reliably extract subject-object pairs, we may return to the original framework and reassess its feasibility. Additionally, we plan to experiment with skipping directly to using LLM’s to assess the accuracy of text as a whole. We’ve seen some potential concerns with this method but will continue to study ways to subvert these concerns.