IS IT ENGLISH?

A PREPRINT

Joshua Ip
Allen School
University of Washington
Seattle, WA 98195
joshuaip@cs.washington.edu

December 18, 2020

ABSTRACT

Frequently, natural language is corrupted by typos, deleted words, and malformed grammar. Identifying corrupted text is an area of interest in natural language processing. In this project, I use a transformer-based language model to identify corrupted sentences in the "Is It English" challenge problem, achieving near 90% accuracy on the validation set.

1 Introduction

Though recent developments in deep learning have allowed for great strides in understanding natural language, understanding corrupted text remains an difficult problem. Natural language, especially on the internet, frequently includes typos, flawed grammar, and bizarre word choice. Interpreting and correcting corrupted text is a field of interest for everything from search engines to speech recognition.

The first step to correcting corrupted text is identifying whether text is corrupted at all. Here, I will create an approach to the problem "Is It English" by Dr. Noah Smith at the University of Washington. In this problem, the model is presented with two different sentences. One sentence is taken from the wild and the other sentence is a corrupted version of that same sentence. The training dataset is a series of pairs of texts where the second sentence is a corrupted version of the first sentence. In the test set, the order of the original text and its corrupted version are randomized, and the model outputs the label "A" or "B" depending on whether the original text was first or second. The "Is it English" dataset includes 1,000,000 sentence pairs and many different types of corruption. Corruptions include typos, deleted words, inserted words, incorrectly encoded symbols, extra punctuation marks, malformed grammar, and swapped proper nouns.

2 Related Work

2.1 Corrupted Text

One of the key applications of correcting corrupted text is query revision. When a user searches using a natural language query, a query may be revised to show more relevant results for what the user is actually looking for. Though the detail of current query revision algorithms are difficult to acquire, Google's Bailey et al. filed a patent in 2006 for an early version of query revision. The algorithm they propose calculates a Query Rank (QR) for each query that a user enters.[4] The QR is based on the frequency that query is searched and user satisfaction with that query, which is measured by how often a user refines their query. The lower the QR, the more likely that the user will refine their query. If the QR is below a certain threshold, Google will suggest a correction to that query immediately below the search results.

2.2 Language Modeling

Query revision is a related problem to language modeling. In language modeling, a model predicts the probability of a given sequence of words occurring in a sentence. [3] If we simplify the problem with a few assumptions, language

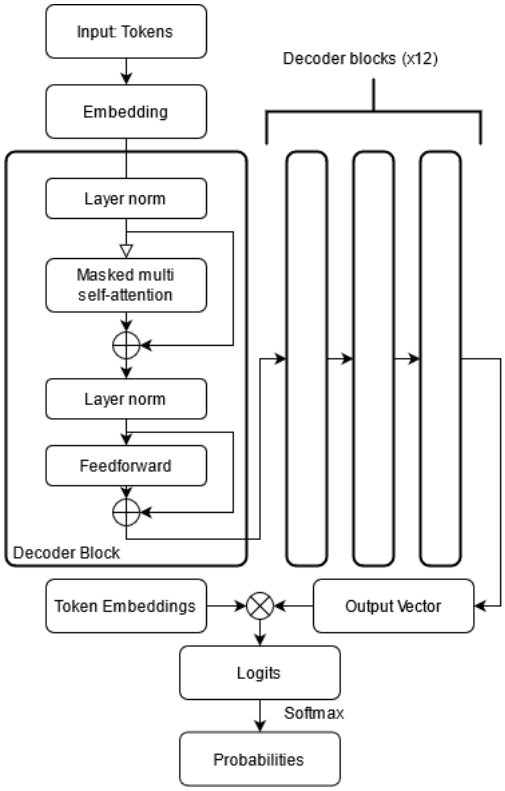

Figure 1: A schematic of the model architecture.

models offer a clear and compelling method of identifying corrupted text. The first assumption is that the sentences taken from the wild are valid English. An inspection of the dataset suggests that the sentences taken from the wild are in fact valid English, though the sentences may be truncated when included in the dataset.

The second assumption is that the corruptions turn valid English sentences to invalid sentences, or sentences that are extremely unlikely to appear in natural English. This is also a reasonable assumption. The wide array of corruptions in the dataset are likely to create invalid English.

With these assumptions in mind, if we train a language model on examples of valid English, the model should predict an extremely low probability for a corrupted version of that same sentence. Since the dataset includes both a corrupted and uncorrupted version of the same sentence, a language model could predict how likely each sentence is to appear in English. Whichever sentence has a lower probability should be the corrupted version.

2.3 Transformers

The Transformer, proposed by Google's Vaswani et al. (2017), uses self-attention to relate parts of a sentence. Architectures based on transformers have achieved state-of-the-art performance on language modelling tasks. As the dataset contains pairs which are dozens to hundreds of words long, it is suitable to use the longer-term memory of transformers for this task rather than less deep but shorter-term memory models like LSTMs or RNNs.

3 Methodology

3.1 Model Architecture

Following the example of Radford et al. (2019), I designed a model with 12 stacked transformer decoders.[1] The tokenizer is a byte-pair encoder with special tokens for the beginning/end of a sentence, unknown tokens, padding, and masks. I heavily used the library *Transformers* by Huggingface, which provides an API for creating custom tokenizers, transformers, and transformer models.

The model outputs a tensor with the logits for the next word. When softmax is applied, the output tensor now contains a prediction of the probability of each possible word in the vocabulary. I find the probability of the next word, take the natural log, then add the log probability to the log likelihood of the sentence. I do this for each sentence in the pair, then compare the log likelihood of each sentence. Whichever sentence has a more positive log likelihood is predicted to be the original sentence, while the other sentence is predicted to be the corrupted version.

3.2 Data

For the dataset, I used the "Is It English" dataset provided by Dr. Noah Smith. Because this is a challenge problem, I do not have the true labels of the test data. I therefore split the data into a training and validation set, with 20% of the data in the validation set. I will evaluate my model based on its accuracy on the validation set, or how many corrupted sentences the model correctly identifies.

3.3 Training

I trained the tokenizer's vocabulary on all of the original sentences in the dataset, discarding the corrupted version. I trained the model on text-token prediction, with cross-entropy loss. I trained the model with a batch size of 16, a block size of 200, a feature size of 768, and five epochs. Training took 27 hours on a GTX Titan X Maxwell.

3.4 Testing

When testing the model, I passed each pair of strings into the model one at a time. Each sentence in the pair is tokenized, then each word is converted to a tensor using one-hot encoding with the vocabulary learned during training. Each sentence's tensor representation is fed into the model independently.

4 Results

Table 1. Performance

Dataset	Correct	Incorrect	Accuracy
Training set	734,561	65,439	91.82%
Validation set	177,877	22,123	88.94%

In general, I am pleased with the performance of this model. In the description of this problem, Noah Smith says that the median accuracy of models on the test set in 2017 was 83%.

4.1 Failure modes

A brief inspection of the incorrectly classified pairs demonstrates a few failure modes of this approach. Of a sample of 50 incorrectly classified pairs of sentences in the validation set, 21 of them featured a word being deleted during corruption. This causes the corrupted version to have fewer tokens than the original version. Because the English language contains so many words, the log likelihood of any particular word appearing is highly negative. The more words appear, the more negative the log likelihood of that sentence.

A higher performance on the training set suggests overfitting. Many of the incorrectly classified pairs included proper nouns. The presence of proper nouns, especially names, may have contributed to the model's poor performance on these pairs. First, the vocabulary of the training data may not include all words in the validation set. When presented with an unknown word, the model may find it difficult to compare sentences when both sentences include unknown words.

4.2 Future Work

There are many different directions for future work. To address the failure mode of word deletion, I could augment the training set by adding padding so that all sentences are the same length. I would also be interested in refining the language model by adding more depth to the model or altering hyperparameters. The severely limited computer power I had for this task made it difficult to train models with different sets of hyperparameters. Alternatively, I could explore how different transformer-based models perform on this task.

References

[1] Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I. *Language Models are Unsupervised Multitask Learners.* :24.

[2] Vaswani A, Shazeer N, Parmar N, et al. Attention Is All You Need. *arXiv:170603762 [cs].* Published online December 5, 2017. Accessed December 18, 2020. http://arxiv.org/abs/1706.03762

[3] Jozefowicz R, Vinyals O, Schuster M, Shazeer N, Wu Y. *Exploring the Limits of Language Modeling. arXiv:160202410 [cs].* Published online February 11, 2016. Accessed December 18, 2020. http://arxiv.org/abs/1602.02410

[4] Bailey DR, Battle AJ, Cohn DA, Engelhardt B, Nayak PP. *United States Patent Application: 0060224554 - Query revision using known highly-ranked queries.* Published online A1. Accessed December 18, 2020.