Generative Word Embeddings with New Similarity Techniques for Legal Linking

Stanford CS224N Custom Project | Mentor Heidi Zhang

Jared Azevedo
Department of Computer Science
Stanford University
jaredssm@stanford.edu

Andrés Felipe Suárez
Law School
Stanford University
asuarezg@stanford.edu

Abstract

The field of law is regarded for its robustness and the amount of work required to be successful. This is especially true when it comes to legal writing. In legal writing, being able to refer to past precedents is crucial to creating a convincing position. This is often a time-consuming and draining process. Having a way to quickly search through those precedents and find which ones best correlate to the position an author is writing about would increase the quality and efficiency of legal writing. The goal of this project was to use different sentence similarity models, specifically pretrained Bidirectional Encoder Representations from Transformers-based models (BERT), and similarity techniques, to match amendments from the Constitution of the United States to snippets of text. We found that using neural network sentence embeddings was preferred over linearly generated embeddings (output from a multi-label classifier) and that distance-based similarity measurements performed best. However, these models still do not perform well enough to completely remove the need for manual searching of related texts.

1 Introduction

The writing of legal texts is a skill that takes many legal practitioners an extended period of time to master. Part of this is being aware of other texts that already exist and recognizing when these other texts can be referenced or reused in their own legal writing to strengthen their position. This can prove difficult for many due to the extensive number of legal texts already written and the challenges of finding similar texts in terms of meaning more than semantics. Resolving this problem could result in improvement in the textual works legal practitioners are able to produce.

While there is already some research and a few products focused on addressing this problem, their efficacy could be improved. One such successful product is Luminance, which offers assistance with a range of tasks, but the most interesting one for this project is their task that searches across legal documents for similar sentences and returns ranked results. The methods of how this ranked retrieval works are not public, but it is known that it only searches across sentences of documents and not larger chunks of text. This can lead to pitfalls where the sentence doesn't align with the context the author is going for. Another successful example is the research done by Shaffer and Mayhew (2019), which this project builds upon. Their application of legal linking between inputs and Constitutional Amendments is great at capturing meaning, but is not very generalized.

The main goal of this project is to try and generalize the work done by Shaffer and Mayhew (2019). We accomplish this by taking the same task and performing less preprocessing on the data. The task can be described as taking phrases (often longer than a single sentence) and attempting to match them with the amendment from the Constitution that best represents the argument in the phrase. Additionally, we applied a variety of models and similarity techniques to find the best-performing combination. The takeaway after performing these experiments was that the combination of methods we employed did not generalize well and that finding a better linking technique at this time is difficult due to the meaning of phrases and their length.

2 Related Work

The objective of our project is to evaluate the accuracy of natural processing models designed to link inputs to a limited number of categories that are provided to the model. This task will be evaluated using inputs, that could be related to a legal category or not, and a finite number of legal categories which in this case correspond to the amendments from the Constitution of the United States. The studies we use as a framework for our work are Shaffer and Mayhew (2019), Man and Tai (2019), Lervtittayakumjorn et al. (2021), Reimers and Gurevych (2019) and Devlin et al. (2019); where the first three are applications of neural networks to the legal field, and the fourth and fifth papers are methodological studies that focus on the development of NLP tools. Additionally, the data used to conduct our study and the baseline model we use to evaluate our implementation are taken from the study developed by Shaffer and Mayhew (2019).

Shaffer and Mayhew (2019) use a multi-label classifier and a neural network model to determine if the content of legal texts, regarded as the inputs, are related to the content of another document, regarded as the knowledge base or reference text. In terms of the paper, the input documents were opinions from cases decided by the Supreme Court of the United States and the reference text which the first documents would be compared was the Constitution of the United States, including its amendments. From the authors' perspective, their main contribution would be to provide robust baseline results for future work in this field [1]. The authors used three methods for this task: a ruled-based method, a linear model, and a neural network model [2]. They evaluated the three models using precision, recall, and F1 measures on two variations of the same dataset, the original and the stripped dataset. As can be seen in Table 1, included in the Appendix, the linear and neural models trained on the original data performed similarly in terms of recall to the rule-based approach, but performed worst in terms of precision. On the other hand, the linear and neural network models outperformed the rule-based approach in terms of recall when trained on the stripped dataset [3].

Man and Tai (2019) conducted a domain adaptation of the German version of the pre-trained language model BERT and evaluated the performance of its implementation on tasks such as classification, regression, and sentence similarity. The authors used as the baseline for comparison the models of Bag-of-Words complemented with term frequency-inverse document frequency and FastText. Concerning sentence similarity, the authors found that Multilingual BERT was not capable of accurately linking legal documents that were found to be related by participants. Regarding the German Bert, the authors highlighted that the algorithm would generally identify two cases as being similar to the input case and that German legal BERT's mean average precision was slightly higher when compared to the other methods. Lervtittayakumjorn et al. (2021) developed a decision support tool that aimed to help regulators analyze complaints in the field of health services. As part of the tool, the authors looked to identify similar past cases to facilitate the resolution of new cases. For the latter, the authors' strategy was to compute the term frequency-inverse document frequency cosine similarity of each of the past cases with respect to the input cases and output the ones that were most similar to the input case. However, according to the ratings provided by human evaluators, the tool did not perform appropriately in this regard, getting a score of 1.8 out of 5.0.

Regarding the methodological papers we follow for our study, Reimers and Gurevych (2019) developed Sentence-Bert (SBERT) to perform several NLP related tasks more efficiently than the existing BERT model. According to the authors, SBERT is a modification of BERT network using siamese and triplet networks which are capable of outputting semantically meaningful sentence embeddings. In contrast to BERT, which takes as input a pair of sentences separated by a special

[1]Additionally, they claimed that in the legal field there were only a few studies that tried to relate two different types of legal documents (Schwartz et al.(2015), Branting (2017) and Nomoto (2018))

[2]The linear model is a multi-label classifier based on the work of Boutell et al., (2004) and Nam et al., (2014). The neural network model is based on the work of Chang et al., (2008)

[3]From the authors' perspective, this shows the linear and neural network models are better suited to find more subtle relations between the sources in comparison to deterministic training set rules.

token and returns a final label; SBERT includes a pooling operation to the output of BERT Devlin et al. (2019)/RoBERTa Liu et al. (2019) to obtain a fixed-sized sentence embedding. The authors used three different pooling strategies: i) using the output of the CLS-token, ii) computing the max of all output vectors, and iii) computing a max-over-time of the output vectors. Figure 1, included in the Appendix, illustrates two of the three pooling strategies chosen by the authors. The architecture on the left of the figure corresponds to a classification objective function and the one to the right to a regression objective function. For the classification objective function, the authors concatenated the embeddings corresponding to the two sentences being compared and their difference, multiply it with a trainable weight matrix $W_t \in \mathbb{R}^{3n \times k}$, and pass it through a softmax function. The loss function for this approach is the cross-entropy loss. On the other hand, for the regression objective function, the authors used cosine-similarity between the embeddings of the two sentences to determine if they are related. In this case, the loss function is the squared-error loss. For our implementation of the SBERT model, we used the SentenceTransformers package developed by the authors SBERT [4].

3 Approach

We are using SBERT (introduced in the Related Works section) to identify the similarity between phrases obtained from decisions of the Supreme Court and the amendments from the Constitution of the United States. For example, given an input string $s \in S$, where S is the set of all input phrases, we want to identify which amendment $a \in A$, where A is the set of all amendments, best matches s. Note that we have added an extra entry into A that corresponds to "none" or not strongly matching any of the amendments. SBERT helps us do this by generating "sentence" embeddings e (same idea as word embeddings but for entire phrases) for each input s and amendment a and then we apply some similarity technique $f(e_s, e_a)$ to these embeddings to score them, as explained in the previous section when talking about the different pooling strategies used in SBERT.

We use as our baseline model the results obtained by Shaffer and Mayhew (2019) in which the authors used a multi-label classifier to study sentence similarity. However, instead of trying to rank s against a where a is the name of the amendment (e.g. Amendment XIV), we have mapped each amendment to their definitions obtained from `constitutioncenter.org`. This was to help the model learn a deeper meaning behind the sentences than just matching to some amendment name that by itself does not mean much (i.e. generalize the task).

3.1 Models

Within the SBERT Python package, we employed three different pretrained models accessed via the HuggingFace Model Hub. The first of these models is all-mpnet-base-v2, which was built by utilizing the already pretrained mpnet-base model by Microsoft and then finetuning it on an additional 1 billion data points. The next model is multi-qa-mpnet-base-cos-v1, which was built and finetuned the same way as the first model. The last model is all-MiniLM-L12-v2 which was built by utilizing the already pretrained MiniLM-L12-H384-uncased model by Microsoft and then finetuning it on an additional 1 billion data points (these are the same data points used to finetune the other two models as well). According to HuggingFace's documentation, all three models have been trained to analyze the relationship between sentences through semantic search.

In contrast to sentence similarity, which tries to link a query to a set of documents based on lexical matches, semantic search looks to improve the search process by understanding the context and meaning of the search query. To attain this goal, semantic search works by: first, breaking down the query into its constituent parts to identify key concepts and entities and understand the relationships between them; second, using techniques such as query expansion to identify related concepts and queries that many not be explicitly mentioned in the query; third, finding the closest embeddings to the query within the vector space, whether they be sentences, paragraphs, or documents (i.e. any phrases).

All of these models have been shared publicly, but do not expose the layers and connections that make them special beyond the base models they extend and Figure 1 seen in the Appendix. Additionally, all three models have a limit on how long input sequences can be which will be analyzed more in-depth in the Analysis.

[4]To learn more about SBERT's Python package, visit `https://www.sbert.net/`

3.2 Similarity Techniques

We tried four different techniques or measures of similarity for this project. The thinking behind each one was that some calculations might better capture the distance between the meaning of the phrases. The first technique we tried was cosine similarity. Cosine similarity is a popular choice for finding the similarity between two embeddings. The equation for cosine similarity is shown in Equation 1.

$$\frac{e_s \cdot e_a}{||e_s|| \cdot ||e_a||} \tag{1}$$

The next technique we tried was the Euclidean distance with n = length of embedding. Another popular choice for finding similarity, the equation for Euclidean distance is shown in Equation 2.

$$\sqrt{\sum_{i=1}^{n}(e_{s_i} - e_{a_i})^2} \tag{2}$$

The next technique we tried was the Manhattan distance with n = length of embedding. The equation for the Manhattan distance is shown in Equation 3.

$$\sum_{i=1}^{n} |e_{s_i} - e_{a_i}| \tag{3}$$

The last technique we tried was the Minkowski distance with n = length of embedding and $p = 2$. The equation for the Minkowski distance is shown in Equation 4.

$$(\sum_{i=1}^{n} |e_{s_i} - e_{a_i}|^p)^{1/p} \tag{4}$$

The code for this project was written entirely in Python and makes use of the SentenceTransformers package that grants access to SBERT implementations and SciPy distance calculations. The Github repository for this project can be accessed through the following link `https://github.com/JaredAzevedoSSM/cs224nlegallinking`.

4 Experiments

4.1 Data

We will be using the Linking Supreme Court Decisions to the US Constitution dataset. It contains over 36,000 paragraphs from US Supreme Court opinions and 41,000 links to the US Constitution. All the pairs of phrases are composed of sentences obtained from opinions of the Supreme Court and the text of the specific article or amendment of the Constitution of the United States to which the opinion is referring or to an empty string in cases where there is no match.

For training, Shaffer and Mayhew (2019) divided the data into two categories: original and stripped data. The decision to use two different datasets for training was made to address the fact that some of the input phrases in the original data included direct references or links (e.g. URLs) to the sections of the Constitution they were referring to. Given that including this information might allow the model to identify the relations between the input documents and the amendments/sections of the Constitution they refer to using a "trivial rule" (the presence of a direct reference to the amendment or section being referred), the authors used a second training dataset, the stripped dataset, in which they removed direct references and links to sections and amendments of the Constitution from half of their observations, limiting the model's ability to make predictions using trivial rules.

For this project, we replaced the name of specific amendments from the Constitution with its entire text. We believe this modification could give the model more information about the categories to which it has to map the inputs, thus improving its performance. Aside from this modification, each sample in our data consisted of three variables: first, the string to be matched; second, the text of the amendment for the inputs that matched any of them or the string 'This input does not strongly match with any of the amendments.' if it did not; and third, a float equal to one for inputs that are linked to an Amendment and zero in cases it did not.

4.2 Evaluation method

In line with the metrics used by Shaffer and Mayhew (2019) on the baseline model, we will use recall, precision, and F1 to evaluate the performance of our implementation. Recall refers to the proportion of true positives correctly identified by the model over the total of true positive instances. Precision is the proportion of true positive instances identified by the model over all the instances identified as positive. F1 is the harmonic mean of precision and recall.

Another motivation for using these metrics was the fact the data is not very well balanced. A large majority of the data points correspond to no amendment and thus using a metric like raw accuracy would prove very little about the performance of our models.

4.3 Experimental details

To get the final results reported below, we ran each model twice. We ran each model on the original dataset and then on the stripped dataset. We used a cosine similarity loss function, a batch size of 16, 100 warmup steps, and 1 epoch when finetuning. While we experimented with the number of epochs and suspect that running the model for longer could improve results, we found that altering the batch size and warmup steps had little to no effect on performance. That said, we did have to use a batch size of 8 for the second model (multi-qa-mpnet-base-cos-v1) due to memory limitations and were able to run the third model (all-MiniLM-L12-v2) for 3 epochs.

The training time varied depending on the model used. For the first two models used (all-mpnet-base-v2 and multi-qa-mpnet-base-cos-v1), we observed a training time of roughly 3 hours and 50 minutes per epoch. For the other model used (all-MiniLM-L12-v2), we observed a much shorter training time of roughly 1 hour and 10 minutes per epoch. This accounts for finetuning on 85% of the dataset and evaluating on the remaining 15% where the split is made randomly (i.e. 15% of the dataset is sampled for the evaluation set and the remainder is for the training set).

4.4 Results

Using the model configurations described above, the best results can be viewed in Table 1 and Table 2. For readability, all-mpnet-base-v2 is referred to as "bert", multi-qa-mpnet-base-cos-v1 is referred to as "search", and all-MiniLM-L12-v2 is referred to as "mini" within the tables.

Model	Similarity	Precision	Recall	F1
baseline	Cosine	79.0	45.8	58.0
bert	Cosine	2.3	0.2	0.3
bert	Euclidean	2.3	0.2	0.3
bert	Manhattan	2.4	0.2	0.4
bert	Minkowski	2.3	0.2	0.3
search	Cosine	6.6	0.3	0.6
search	Euclidean	6.6	0.3	0.6
search	Manhattan	6.7	0.3	0.6
search	Minkowski	6.6	0.3	0.6
mini	Cosine	14.5	0.5	1.0
mini	Euclidean	14.7	0.5	1.0
mini	Manhattan	14.8	0.5	1.0
mini	Minkowski	14.7	0.5	1.0

Table 1: Original dataset results

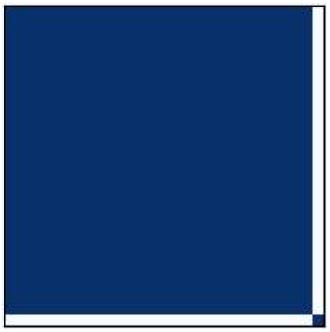
Model	Similarity	Precision	Recall	F1
baseline	Cosine	68.3	54.3	60.5
bert	Cosine	13.3	0.6	1.1
bert	Euclidean	13.4	0.6	1.1
bert	Manhattan	13.3	0.6	1.1
bert	Minkowski	13.4	0.6	1.1
search	Cosine	2.6	0.2	0.4
search	Euclidean	3.3	0.2	0.4
search	Manhattan	4.3	0.3	0.5
search	Minkowski	3.3	0.2	0.4
mini	Cosine	16.6	0.5	1.0
mini	Euclidean	12.6	0.5	0.9
mini	Manhattan	14.5	0.5	1.0
mini	Minkowski	12.6	0.5	0.9

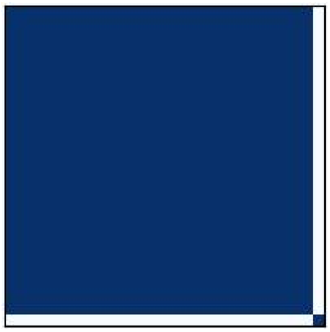
Table 2: Stripped dataset results

Quantitatively, these results are somewhat expected due to the generalization we tried to apply. Overall, they are still worse than we were hoping for in comparison to the benchmarks of the baseline and other results obtained by Shaffer and Mayhew (2019). We believe this indicates that trying to generalize the legal text linking problem with the current methods and similarity techniques is ineffective and ill-advised.

5 Analysis

As evidenced by the quantitative results above, the models did not have much success on this problem. There are a variety of facets that we looked at in order to do qualitative analysis. First, we decided to evaluate how similar or different the embeddings of the amendments of the Constitution are after the finetuning process. We conducted this analysis based on the understanding that if there are no meaningful differences among the embeddings of the amendments, the model will have a hard time distinguishing the amendments and linking the inputs fed to the model with each of them. Figure 1 shows the cosine similarity for the embeddings from all the amendments for all three models we implemented. Given that cosine similarity takes values between -1 and 1 (white to dark blue), where 1 implies that the two embeddings are proportional and -1 that they are opposite, we find that there seems to be significant overlap between the amendment embeddings. For instance, in the case of the all-mpnet-base-v2 and multi-qa-mpnet-base-cos-v1 models, subfigures **a** and **b** seems to illustrate an issue in terms of training, given that it returns cosine similarities equal or close to one for all combinations of amendments but the empty string or absence of a match. In like manner, subfigure **c** also shows that there is a high correlation, although much less than the previous models, for the amendments obtained using the all-MiniLM-L12-v2 model. Note that the only embedding completely unrelated to any of the others in all three models is the embeddings for "None" or not matching any of the amendments.


(a) all-mpnet-base-v2


(b) multi-qa-mpnet-base-cos-v1

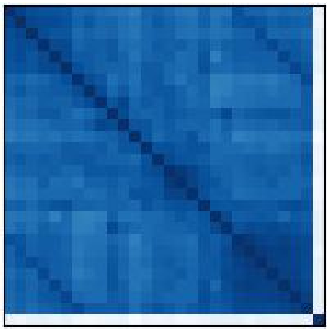

(c) all-MiniLM-L12-v2

Figure 1: Cosine similarity for amendments

Another aspect that we considered to impact our results is the length of the inputs and amendments. The models we used have a maximum text length of 384, 512, and 256, respectively, after which point the model just truncates the string. After looking at some of the later amendments and inputs, there are a significant number of strings that fall into this category and end up being truncated. This suggests a limitation in the models, but also a limitation in that it may be asking too much to compare larger phrases and achieve high evaluation metrics at the same time.

One aspect that we can confidently say mired our results is the fact we did not employ the same training tricks Shaffer and Mayhew (2019) used. In their paper, they discuss first balancing the number of direct and indirect references in the data they feed the model and second, adding a factor to decrease the influence of negative samples since there are so many. Additionally, as highlighted previously, we decided to train our models using the complete texts of the Amendments instead of their names based on the assumption that given more information to the model would allow it to increase its accuracy. However, given the analysis we conducted in relation to the similarity among the embeddings from the Amendments, it looks like this approach also added noise to the model given that despite some differences the Amendments also share common words such as connectors.

Additionally, they evaluated on a set of hand-annotated examples that we did not have access to and were not able to replicate for this project. This led to us evaluating on a random subset of the data which makes it difficult to truly compare our results to theirs. This could have been mitigated by using their linear model with the data as we were using it, but we also were unable to get this model running due to errors with the Python package required to run it. Had we been able to run it, it would have given us a better baseline to compare our results to that didn't completely deviate from their original research.

6 Conclusion

For this project, we focused on applying a combination of pretrained models and similarity measurements on the legal linking task with the idea that this legal linking could be expanded to relate pieces of text. We learned that using the data without decreasing the influence of negative samples and not evaluating over hand annotated examples resulted in much lower performance than seen by Shaffer and Mayhew (2019). This could be partially accounted for by not having the time and compute speed required to iteratively test larger amounts of finetuning and annotated data. However, we were able to observe that some models and similarity techniques performed better on this task relatively speaking. Lastly, we discovered the the task of semantic search instead of sentence similarity is a better way to tackle this problem.

For future work, we would like to see these models and similarity techniques applied to exactly the same data Shaffer and Mayhew (2019) used in order to determine if using more complex architectures could bring better results in terms of accuracy. That is, the data used to finetune should include an equal number of direct references and indirect references, evaluate over hand-annotated examples, and account for the large number of negative samples. Access to more time and compute speed could also see improved results even on the generalized data. Ideally, we want to see the attempts performed in this project applied in the same context as the original work by Shaffer and Mayhew (2019) to accurately discern if there is an improvement in performance.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers forlanguage understanding.

Piyawat Lervtittayakumjorn, Ivan Petej, Yang Gao, Yamuna Krishnamurthy, Robert van der Gaag, Anna annd Jago, and Kostas Stathis. 2021. Supporting complaints investigation for nursing and midwifery regulatory agencies.

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

Chin Man and Yeung Tai. 2019. Effects of inserting domain vocabulary and fine-tunning bert for german legal language.

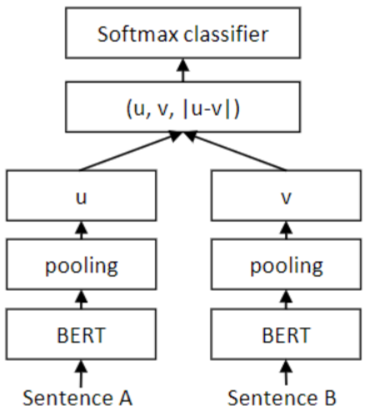
Nils Reimers and Irina Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Robert Shaffer and Stephen Mayhew. 2019. Legal linking: Citation resolution and suggestion in constitutional law. In *Proceedings of the Natural Legal Language Processing Workshop*.

A Appendix

Model	Precision	Recall	F1
Rule-based	91.8	47.0	62.2
Linear (original)	79.0	45.8	58.0
Neural Network (original)	82.1	46.8	59.6
Linear (modified)	68.3	54.3	60.5
Neural Network (modified)	76.5	56.2	64.8

Table 3: Baseline results



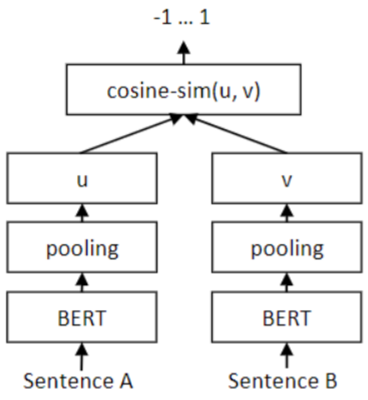


Figure 2: SBERT architecture with classification objective function and SBERT architecture to compute similarity score