Multilingual Newspaper Article Similarity using BERT encoder and Decision Trees

Wesley Machado Andrade and Lasse van den Berg

Abstract

This document is a report on the final project by Wesley Machado Andrade and Lasse van den Berg for the LING 227 course, Spring 2023. It is a write up of the problem task, approach and work process, implementation, testing and results of the project. We created a program that can evaluate the similarity of news articles regardless of the language. We explored different models of the BERT encoder and tested both LGBM and Random Forest Regressors.

1 Introduction

1.1 Problem Task

For our final project, we decided to work on a task proposed by SemEval 2022, an international NLP research workshop. More specifically, we opted for task No. 8, specified as "Given a pair of news articles, are they covering the same news story?"[1] Thus, comparison of the topic of newspaper articles that may be in different languages, are at the core of this task.

1.2 Motivations

There are several reasons why a program that can assess the overall similarity of newspaper articles in different languages may be useful. When studying journalistic coverage of global events in different geographic regions, a program like the one proposed by the task would serve as a great tool to identify relevant articles. Ideally, a program like this can help identify all articles regarding a certain topic, regardless of region of publication, language, or sentiment.

Furthermore, as international students, we are especially interested in the multilingual aspect of the project, hoping to create a program that would facilitate the previously mentioned analysis between our own native languages and English.

1.3 Specific Requirements

We now introduce the specific goals of our program. When presented with two different news articles, which may vary in language, the program should evaluate the similarity of the topic the two news articles cover and assign an overall similarity score. This score must then be mapped to an output on a continuous scale on the interval [1;4], a higher value denoting a higher level of similarity. The program should work regardless of which language the articles are written in and should ideally not be influenced by differences in tone or sentiment.

2 Implementation

2.1 Data Collection

Considering that a dataset of paired, annotated articles was already provided, we first converted the data into a usable format. The articles were presented as CSV files containing URLs to the articles as well as annotated values, denoting the average similarity score assigned by human annotators. Furthermore, a program containing a web-scraper was given[2], allowing us to easily extract the raw text from the URLs.

The execution of the scraping program took more than 20 hours. Therefore, we saved the raw text files locally, allowing for continuous access to them. The files contained the raw text, title, and similarity ratings of a pair of articles each. It also contained information on the overall similarity ratings assigned. After accessing all the data, we

[1] https://competitions.codalab.org/competitions/33835#learn_the_details-timetable

[2] https://github.com/euagendas/semeval_8_2022_ia_downloader

created some Python files that allowed us to load all the raw data into a program and convert it into our desired format.

2.2 Approach

Since some of our initial approaches do not resemble our ultimate implementation in any way, these ideas are discussed in a later section.

Initially, the problem task seemed rather complex. Therefore, we began brainstorming by thinking about how we would solve such a task ourselves. While fairly straightforward for a human, it seemed as though there were two main steps or sub-tasks into which the problem could be decomposed.

First, we would have to gain a sense of the individual meanings/topics of the articles we were presented with. This is quite easy for native speakers of any article's language. The underlying step our program would have to handle, is also very similar. Essentially, the two articles must be encoded or embedded into a format/representation that allows to exploit patterns arising in these representations from the data. The patterns must be meaningful in the sense that articles annotated with a high overall similarity rating (and ultimately articles that simply are similar in meaning) should exhibit similar patterns within the chosen representation, i.e. the chosen representation for individual articles should be similar in computationally conceivable way, if the articles, too, are similar in terms of topic. The reasoning behind our ultimate choice for the BERT model is discussed in the next section.

The second step of the overall task is to find a way of comparing the established representations by mapping attributes, features, or properties of the representations of the articles to the desired output. In our case, that desired output would be the 1 to 4 similarity scale. In human terms, this would correspond to the comparison of our mental representations of the topics of the articles and then assessing how similar these topics are. Note how this is reflective of the process the human annotators must have gone through while providing the test data. We ultimately chose to experiment with two different methods of comparing the embeddings and mapping them to the output.

2.3 Document Embeddings Using BERT

After consultation with Professor Frank and reviewing the ideas, we decided to use the BERT model as the basis of our document embeddings. BERT is a language model based on the transformer architecture that can be used to create context aware embeddings of texts, which is useful for our purpose.[3] Transformers have become the state-of-the-art method for a variety of NLP related tasks in the previous years, including machine translation, generative language models, etc. It uses the so-called attention mechanism to encode the meaning and structure of an input sequence of language tokens as a whole, rather than individually. It also learns which words in the context are more important than others, allowing for a deeper and more meaningful representation of each word's embedding. Different methods, including pooling, can then be used to create a document embedding from the individual context-aware word-embeddings.

The pre-trained BERT model can be run through a Python library and has functions that take raw text as an input and return a tensor embedding.[4] While there are some possibilities for finetuning, the BERT model itself should work well enough for our purposes. We initially used the multilingual version[5] of that is trained on data from several different languages. To obtain the final embedding, we did a pooling calculation from the embeddings of all input tokens using the attention matrix.

2.4 Comparison of Embeddings and Mapping to desired Output

Now that functions to create embeddings were in place, we needed a method to compare the two tensors representing each article. There are several ways of comparing tensors and the results from these distinct methods can be used as features to different regressor models. Our need for a regressor arises from the fact that we intend to output a continuous value on the interval [1;4]. While a classifier with 4 discrete output values would appropriately fulfill our task, it would be

[3] Jacob Devlin et. Al. (2019)

[4] https://huggingface.co/docs/transformers/model_doc/bert

[5] https://huggingface.co/bert-base-multilingual-cased

impractical to train the model against a dataset that uses semi-continuous values.

One of the regressors we chose to work with is the Light Regressor Boosting Machine (LGBM)[6] that trains decision trees through iterations over a data set. Its architecture allows it to also access information from the previous decision tree. It is a very common regressor in NLP tasks, since it handles high-dimensional input values as well as big data sets very effectively. Furthermore, it is also good at handling unbalanced or biased data sets. Our dataset contains from a variety of languages, but some appear more frequently than others, further supporting our choice.

The other regressor we used is the Random Tree Regressor[7]. It can also return continuous values, but it uses a different approach than LGBM. It randomly picks features from the data and generates decision trees, hence its name, and then outputs a value after analyzing the outputs of each decision tree. It has similar advantages to the LGBM regressor and is frequently used in similar contexts.

As mentioned, both of these models require a set of features according to which the document embeddings can be compared. Having consulted applications using models such as BERT, we picked the four features that are most frequently used in comparing tensor embeddings.

Our first feature is the cosine distance of the high-dimensional vector-space embedding. It is calculated using the following formula:

$$cos(\theta) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

The next feature is the Manhattan Distance of the embedding, which is defined as the summed absolute distances of the embeddings:

$$dist_m(A,B) = \sum_{i=1}^{n} |A_i - B_i|$$

Furthermore, we used the so-called Jaccard similarity measure which treats embeddings as sets and calculates the proportion of the intersection of those sets to the union of those sets. Thus, the higher the Jaccard similarity, the more intersection. The embeddings, represented as sets, have. It is defined as follows:

$$jaccard_{sim}(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

The last feature we used, is the so-called Dice similarity. It is defined as the proportion of twice the intersection of both embeddings to the summed sizes of both embeddings:

$$dice_{sim}(A,B) = \frac{2 \cdot |A \cap B|}{|A| + |B|}$$

Our different regressors have access to all of these values. During training, they process the data to learn how these features should be weighted.

2.5 Transition to Google Translate

After training our model, we weren't satisfied with the accuracy results. In order to evaluate if the BERT model was one of the issues, we decided to use a different approach and BERT model. The standard BERT model relies on purely English datasets and is a more powerful model overall. After consideration of the models of groups competing in the competition[8], we created a second model that would translate both documents to English and then create embeddings from the purely English versions, relying on the stronger embedder. Therefore, we needed to translate all documents using the google translate API. The regressors used the same methods as the multilingual model.

3 Testing

3.1 Accuracy Measurement

Since the model predicts a value on the continuous interval from 1 to 4, accuracy needs to be measured using a correlation. This is because we are not mapping to discrete numbers (considering, also, that the values in the training data are averaged and

[6] https://lightgbm.readthedocs.io/en/v3.3.2/

[7] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[8] Ishihara and Shirai (2022) and Wangsadirdja (2022)

not discrete numbers on the 1 to 4 scale). Pearson correlations, therefore, serve as a good measure to evaluate how well the model performs by comparing the linear relationship between the expected output and our predicted output.

4 Results

We now present the results of our model, measured in three different ways. At first, we notice that the translational model worked significantly better than the multilingual model. There was no significant difference in performance between the different regressors, while the LGBM performed slightly better in both cases.

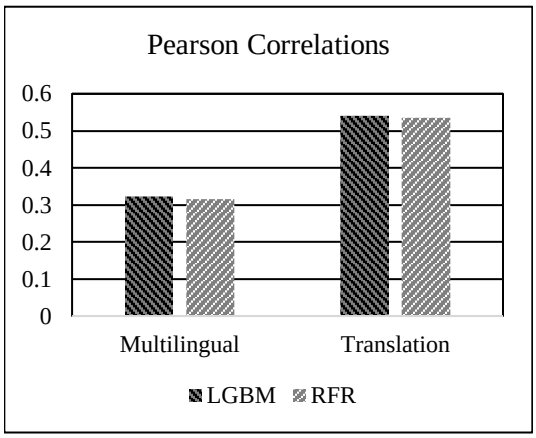

Diagram 1: Overall Pearson Correlations of all our models

To also evaluate the internal performance and significance of the features presented to the regressors, we analyzed the normalized weights assigned to each feature by the regressors.

Evidently, in all models, both regressors almost entirely discarded the Jaccard index as a feature. This is probably attributable to the fact that the Jaccard index and the Dice index have very similar definitions. Therefore, their values are generally similar, and it would not make sense for the decision trees regressors to consider both. Thus, through continuous training, both models will learn to prioritize the Dice feature over the Jaccard feature.

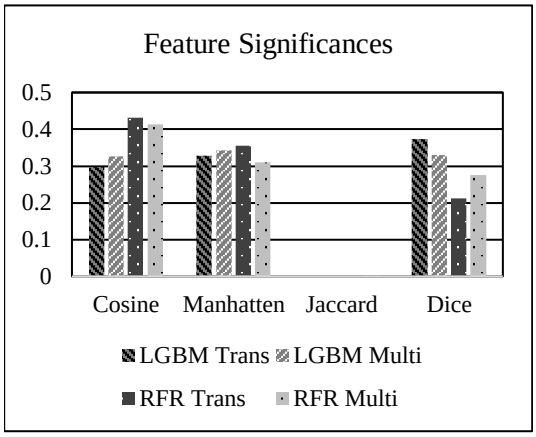

Diagram 2: Significances assigned to the different features by each regressor in each model

We further analyzed the Translational model's performance on the individual language pairings. There were significant differences despite the texts having been translated via Google Translate. The worst performance occurred on the German-to-German pairing, while the best occurred in the Spanish-to-English pairing (Table 1). There are many possible explanations for this, but we hypothesize that during the translation from certain

Language Pairs	LGBM	RFR
All-All	0.5410	0.5355
German-English	0.6621	0.6462
Chinese-English	0.6455	0.5686
French-French	0.6494	0.6653
Spanish-Italian	0.5909	0.5645
Polish-English	0.5764	0.5526
English-English	0.6124	0.5892
Spanish-Spanish	0.6212	0.5954
Spanish-English	0.6828	0.6613
Polish-Polish	0.4640	0.4968
Italian-Italian	0.6032	0.6263
Chinese-Chinese	0.4980	0.5354
Russian-Russian	0.3835	0.3424
Turkish-Turkish	0.6589	0.6689
French-Polish	0.5074	0.2670
German-German	0.2601	0.2617
Arabic-Arabic	0.4001	0.3943
German-French	0.4467	0.4146
German-Polish	0.5414	0.4906

Table 2: Language Pair Pearson correlation results for the Translation Model

languages to English, information is quite literally "lost in translation" to a different extent. We further note, that on average, any pairings that include English outperform those that do not by far. We attribute this to the fact that in these cases, only one of the texts is being translated. Thus, only one step of translation during which potential structural and contextual information could be lost, is executed.

Across most language pairs, LGBM and RFR performed comparably. One surprising instance was the RFR's weak performance on the French-to-Polish pairing. From our analysis we cannot arrive at a conclusion that would explain this finding.

5 Unsuccessful Alternatives

5.1 Discarded approaches

Initially, we planned on using a far simpler approach. We wanted to train a Naïve Bayes classifier that would recognize words judged as important, for instance, geographic locations, names, and dates, since those are likely to be consistent across similar articles and similar articles only. However, it proved difficult to find a way of reliably extracting such data from the dataset without having to manually check all instances. Therefore, we decided to stick with approaches that would work with the data format we were provided with.

6 Relation to Previous Work

In general, our results for the translation model are somewhat consistent with those of the participants' models we analyzed. However, due to our simpler architecture and strongly constrained computational power/time our final correlation score is not as strong as the top rankling ones of the competition. Nonetheless it compares to intermediate scores some of the teams report during their work process. However, differing from our results, most participants usually received similar scores for the translational and multilingual approaches. We attribute the weakness of our multilingual approach to the pooling method we use in the multilingual BERT encoder. Our approach took all the generated embeddings of each input, while other participants that obtained better results relied on more complex architectures and more specific embedding selection from the different layers of those architectures. We hypothesize that our approach weighs structural differences of languages more heavily than purely semantic information, thus resulting in very weak cross-language comparison.

7 Linguistic Assumptions

As is the standard in many modern NLP applications, our model relies heavily on purely statistical learning methods. Nonetheless, there are a few components of our project that are closely related to linguistic theory.

The multilingual BERT model creates embeddings for texts from a multitude of languages. This model is trained on data from a set of different languages without a clear distinction between them. Essentially, it treats all of the different documents as though they are part of the same language. One could argue that this relates to the theory of Universal Grammar, proposing that all languages rely on the same set of parametrized rules and principles and simply vary in the applying parameters of any language. While connectionism is frequently regarded as an alternative approach to Universal Grammar, the functionality of a connectionist model that works cross-linguistically could be considered an argument for Universal Grammar. Perhaps a deep learning architecture exposed to a large volume of data from different languages "picks up" on those principles and patterns and abstractly embeds those generalizations in its architecture. Perhaps it manages to exploit certain aspects of input data and use them as parameters to the described principles and rules. Considering, however, that the pure English BERT model generally performs better than the multilingual one, we can assume that at this point our implementation of the multilingual BERT does not pick up on such generalizations, or at least doesn't do so to such an extent that it can enhance the performance of multilingual tasks. Nevertheless, language modeling has improved significantly in recent years and might become a relevant tool for the study of human language beyond its numerous applications.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. 2019. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* Prentice-Hall, Englewood Cliffs, NJ. https://doi.org/10.48550/arXiv.1810.04805

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens and Mattia Samory. 2022. *SemEval-2022 Task 8: Multilingual news article similarity*. https://foi.org/10.18653/v1/2022.semeval-1.155

Shotaro Ishihara and Hono Shirai. 2022. *Nikkei at SemEval-2022 Task 8: Exploring BERT-based Bi-Encoder Approach for Pairwise Multilingual News Article Similarity*. https://doi.org/10.18653/v1/2022.semeval-1.171

Dirk Wangsadirdja, Felix Heinickel, Simon Trapp, Albin Zehe, Konstantin Kobs and Andreas Hotho. 2022. *WueDevils at SemEval-2022 Task 8: Multilingual News Article Similarity via Pair-Wise Sentence Similarity Matrices.* https://doi.org/10.18653/v1/2022.semeval-1.175

Supplementary Material

The code and documentation explaining how to use our model can be found on the following GitHub repository: `https://github.com/wesleyandrade05/ling227final`