AUTOMATED META-DATA TAGGING AND TOPIC MODELING

Aidan Gawronski, Duc Vu, Florencia Díaz

FourthBrain - Machine Learning Engineer

July 2021

https://github.com/agawronski/word-embeddings

Summary

Topic modeling is an unsupervised machine learning technique that allows us to scan a set of documents, and automatically group words and similar expressions to help us discover patterns in the documents. Dynamic topic modeling refers to the introduction of a temporal dimension into the topic modeling analysis. In particular, dynamic topic modeling in the context of this project refers to studying the change over time of specific topics.

Our experiments demonstrate that word embeddings find topics that are significantly more informative and representative of the corpus trained on than probabilistic generative models.

0. Introduction

Given a corpus composed of many texts, referred to as documents, a topic model will discover the latent semantic structure, or topics, present in the documents. Topics can then be used to find high-level summaries of a large collection of documents, search for documents of interest, and group similar documents together.

There are several existing algorithms you can use to perform the topic modeling. The most widely used methods are Latent Dirichlet Allocation (LDA).

However, we know that algorithms cannot work with raw text directly, the text must first be converted into numbers. So we were interested in investigating how to come up with good numerical representations of words.

1. Background and Significance of Project

GLG is a global company that connects its clients to the world's largest and most varied source of first-hand expertise, including executives, scientists, academics, former public-sector leaders, and the foremost subject matter specialists.

The goal is to match each request to a topic specialist in their database. This project on Natural Language Processing (NLP) is aimed at improving the topic/keyword detection process from the client submitted reports and identifying the underlying patterns in submitted requests over time. The primary challenges include Named Entity Recognition (NER) and Pattern Recognition for Hierarchical Clustering of Topics.

We were asked to solve these particular questions:

1. Can we group similar client requests together? (Eg. Google News)
2. Can we perform NER for unstructured data geared towards the Tech Industry or Healthcare industry with reasonable accuracy?
3. Can we find hierarchical patterns in the topics for requests to identify temporal directions of the requests?

2. Related Work

The most widely used topic modeling technique is Latent Dirichlet Allocation (LDA).

LDA processes documents as a 'bag of words', which basically counts how many times a word appears in a document but ignores the ordering and semantics of words.

Learning word representations lies at the very foundation of many NLP tasks because many NLP tasks rely on good feature representations for words that preserve their semantics as well as their context in a language.

In 2003 the concept of word embeddings — or distributed representations of words and documents were introduced by Bengio et al. They learn numerical representations of words, in the forms of vectors, looking at the words surrounding a given word. The most known are Word2Vec, Glove, and FastText.

Once you have vector representations of words, you can use Euclidean distance (or cosine similarity) between two-word vectors for measuring the linguistic or semantic similarity of the corresponding words.

But they are still context-free models because it generates a single word embedding representation for each word in the vocabulary.

Things changed in 2018 with the invention of BERT. This was enabled by recent advances in neural architectures, such as the Transformer, followed by the emergence of large-scale pre-trained models such as GPT-like, BERT-like, BART/T5-like, models which eventually revolutionized NLP and pushed the state of the art for a number of NLP tasks.

3. Explanation of Datasets

For this project, we downloaded a dataset from jstor.org, a digital library that provides access to more than 12 million academic journal articles, books, and primary sources in 75 disciplines. This dataset contains 1500 papers on Healthcare industry from 1900-2021

We also work with a database from Wikipedia that contains 42,786 names and text from their Wikipedia page[1]. We queried a public API to add the occupations to each of the records in the dataset.

[1] https://www.kaggle.com/sameersmahajan/people-wikipedia-data

4. Explanation of Processes

We performed a simple preprocessing on the content of full-text columns to make them more amenable for analysis, and reliable results. To do that, we removed any punctuation, stopwords and then lowercase the text. We then extracted the embedding for every unique word in the article using SentenceTransformer("all-mpnet-base-v2"), which map sentences & paragraphs to a 768-dimensional dense vector space. The vectors can then be used for tasks like clustering or semantic search.

After that, we averaged the embeddings across all words for each of the documents to get a single vector with 768 values for each document.

We also explored two other methods of creating vector representations for each document. The second was very similar to the first, however after removing stop words and punctuation, we calculated the frequencies of each word, and kept only the top 30 for each document. Then we used the same sentence transformer on each of the 30 words, and reduced this to a single vector by using a weighted average, with the average being based on the frequency of the word relative to the total.

The third method used the sentence transformer on the entire document, in hopes that it could potentially capture the context and meaning of the document better.

We looked at the training and architecture of several different NER models, including Python's spaCy, Flair, and BioBert, and compared their accuracies. Especially we want to apply NER in biomedical areas in which medical named entities are recognized from medical texts, such as diseases, drugs, surgery reports, anatomical parts, and examination documents.

For topic modeling, we applied LDA and BERTopic.

Bertopic is a technique that leverages transformers and class-based TF-IDF to create dense clusters allowing for easily interpretable topics whilst keeping important words in the topic descriptions.

For evaluation, we calculated Topic Coherence measure for each of the models. Coherence score is for assessing the quality of the learned topics, LDA results in a coherence score lower than Bertopic (0.538 vs 0.640).

For the final application, we deployed a Streamlit web app via a docker container running on Amazon Elastic container service. The application accepted some text as user input and then performed Named Entity Recognition as well as extracting the vector representation of the text and looking for the closest documents in the vector space and returning the results to the user.

We created several different iterations of the application to explore different types of named entity recognition, and how the results of the different embeddings performed. We also explored returning the nearest documents from the Jstor data as well as the wikipedia data to simulate finding similar documents in the former and finding experts in the latter.

5. Explanation of outcomes

Although we had hoped that the full document embeddings would perform the best it seemed like the weighted embeddings were the best both for finding similar documents as well as for finding experts. It is tough to say though, as best in this case is a very subjective term. In summary and future directions we explore implications of this subjectivity.

Another exploration was performing Kmeans on the embeddings, and what we found first was that there was no clear “best number of clusters". The elbow method seemed to imply that it should be at least 20 clusters, however the curve was fairly smooth and there

was no obvious “elbow”. The initial experiment was with 20 clusters and looking at a random sample from each; It did not appear to find documents that were relevant to each other topically. A second experiment was to only look at the five documents closer to the center of each cluster, and this seemed to find documents that had more relevance to each other. A third experiment was to set the number of clusters to n/10 such that we ended with many, many very small clusters. This also yielded more relevant groupings. The takeaway was that finding the closest document to another document seemed to yield something relevant and we took this information when building the final application.

6.Ethical considerations

One ethical consideration of the project is consideration of what data was actually used to train the models which create our embeddings. In the same way that face recognition can perform poorly on people of a particular race if it did not include proper representation in the training data, so too can sentence embeddings do a poor job of capturing certain meaning or topics, if the data they were trained on did not include them. It is unclear how this would impact the specific business use in a practical matter, but one recommendation would be to schedule quarterly review brainstorming for possible implications. Questions to ask at these meetings: Are we recommending a diverse pool of candidates? Are there topics that don’t seem to ever come up that perhaps should? Do we need to retrain/fine tune some of the embedding models to include text with certain kinds of information from more diverse text sources?

Other possible ethical implications include what the clients are using the requests for. For example if a government agency is trying to find experts in order to guide policy making, and the result of the recommendations has implications for society at large then ensuring that

experts who are broadly knowledgeable, or offering experts who approach a topic from several different angles may be important to consider.

7. Summary and future directions

One obvious future direction is the exploration of the plethora of options for creating vector representations of documents. One issue that was mentioned earlier is the difficulty of defining which ones are actually "the best" as best is subjective:

Due to the difficulty of defining what method of "finding experts:" is best, we would recommend to GLG that they let business performance guide the choice of method. Initially, you could use 2 or 3 different methods to embed a text request and find relevant experts then recommend the top choices from each to the client. The important part here is to properly track which options were presented to the client, and which models were used, as well as which choice was made (if any) and then subsequently use a survey to indicate their satisfaction. Properly tracking the business performance from each of these models would allow the poor performers to be phased out, and new contenders to be added into the mix.

For NER, one can use spaCy and Biobert to pretrained pretty much right out of the box as a baseline, and it is relatively easy to feed data through to obtain named entities. However, since these are pretrained models , there is little flexibility in terms of types of NER datasets that one can train/finetune on. As a next step, we fine tune a BERT model (using the HuggingFace library built on PyTorch) on a publicly available NER dataset (https://www.kaggle.com/abhinavwalia95/entity-annotated-corpus).

Due to time constraints, we could not do the following: Try to fine tune a transformer on a more specialised NER dataset, such as one pertaining to healthcare (such as this covid specific one https://xuanwang91.github.io/2020-03-20-cord19-ner/). Try to fine tune a lighter

weight transformer model such as distilbert and/or mobilebert, and compare performance. We plan to incorporate some level of flexibility into the fine tuning pipeline.

With regards to the second bullet point above, using a distilbert or mobilebert model will be more suited to large scale production models, due to their smaller size (and hence quicker latency).

References

- Natural Language Processing with TensorFlow, Thushan Ganegedara
- Exploring Topic Coherence over many models and many topics: https://aclanthology.org/D12-1087.pdf
- How to use BERT to create your own topic model: https://towardsdatascience.com/topic-modeling-with-bert-779f7db187e6
- The General Ideas of Word Embeddings https://towardsdatascience.com/the-three-main-branches-of-word-embeddings-7b90fa36dfb
- Transformers: “The End of History” for Natural Language Processing? https://arxiv.org/abs/2105.00813
- A Neural Probabilistic Language Model https://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf