Semantic Similarity Methods for Content Recommendation

December 2021

Sara Ho

Abstract

The task is to recommend contextual information regarding items in a given document. We start with a pipeline that matches news descriptions with Wikipedia articles. The existing pipeline inputs news documents and returns suggested Wikipedia articles. This paper is focused on the final step, which is to label the suggested Wikipedia articles as "relevant" or "irrelevant" context for the given news description. This paper covers similarity methods through embedding, and details a suggested pipeline to improve the relevancy of recommendations. The code can be found on GitHub.

1 Introduction

The goal is to recommend information that provides context to the items in a given document. This use case is becoming more common and important as people more heavily rely on online media to deliver crucial day-to-day information. Take, for example, a brief news headline. The site may want to suggest additional informational content surrounding the current event and the actors involved. At a micro-level, the suggested content should be interesting and helpful to the user. At a macro-level, suggesting helpful contextual content can serve as a preventative action against misinformation and misinterpretation. This paper experiments with various semantic similarity methods in a text-based recommender system pipeline to label suggested Wikipedia articles that can provide context to daily news headlines.

The next section discusses the related work in the field of semantic similarity. The third section describes the data. The fourth section describes the methodology behind the existing pipeline and the experimentation surrounding the final recommendation prediction. The fifth section summarizes the results and the final section discusses room for further experimentation.

2 Related Works

The task in this paper differs from previously explored news recommender systems in a few ways: 1) There is literature that combines user-based and content-based methods for a news recommendation problem, but the recommender in this paper is a pure content-based recommender. We assume the same articles are recommended for all users regardless of their individual characteristics. 2) In already-explored problems, the recommender needs to recommend additional news articles by ranking the relevance of a limited set of articles and choosing the top candidates. In our case, we have a suggested set of Wikipedia articles delivered using an imperfect pipeline and we assume some of them are objectively relevant and some are not. 3) For the traditional news-to-news recommender, there is little downside to recommending something irrelevant. For the news-to-context recommender, we *only* want to serve relevant articles.

Moerland (2013) provides an overview of what to expect in a text recommender system and why semantics are necessary. They start with cosine similarity and the TF-IDF weighting scheme which can be considered a baseline setup for news recommendation tasks; this is similar to our setup. Then they make the case for the need for semantic similarity over simple lexical similarity, which they experiment with by using WordNet.

We follow a similar approach but apply three different semantic similarity approaches, one using a simple cosine similarity threshold that does not incorporate semantics, one that incorporates semantics via pre-trained Glove Vectors (Pennington 2014), and one that

considers the sentence as a whole (Cer 2018). These are not the only semantic similarity methods that we could have chosen from; Chandrasekaran (2021) provides a current overview of available methods.

3 The Data

The news headlines are retrieved from NewsAPI.org. The headlines are diverse and span politics, entertainment, sports, and more. The Wikipedia articles are queried from Wikipedia's API. The existing pipeline suggested these Wikipedia articles for the terms "Social Democrat Magdalena Andersson" and "the Green Party". The first article "Magdalena Andersson" is relevant. The second article "Green Party of the United States" is not relevant.

"New Swedish PM resigns on first day in job, hopes for swift return - Sweden's first female prime minister, **Social Democrat Magdalena Andersson**, resigned on Wednesday after less than 12 hours in the top job after **the Green Party** quit their two-party coalition, stoking political uncertainty."	
Magdalena Andersson	"Eva Magdalena Andersson (Swedish pronunciation: [eːva magdalena andɛsɔn]) (born 23 January 1967) is a Swedish Social Democratic politician…[truncated]"
Green Party of the United States	"The Green Party of the United States (GPUS) is a federation of Green state political parties in the United States. The party promotes green politics, specifically environmentalism; nonviolence; social justice; participatory democracy, grassroots democracy; anti-war; anti-racism and eco-socialism…[truncated]"

Table 1: Example news-to-wiki match.

In reality, news topics will change from day to day, so we want to accurately evaluate a model trained on lagged data. The training dataset comprised of news articles from 6 months ago. The test dataset comprised of news articles from this week.

	Total Number	Average N Words	Min N Words	Max N Words
News Documents	128	26	11	47
Wikipedia Content	684	153.5	1	2998
Median number of wiki suggestions per news item: 6				

Table 2: Summary of training data.

	Relevant	Irrelevant
Training Data	358	537
Testing Data	35	77

Table 3: Label imbalance

The data is imbalanced as shown in Table 3. In preprocessing, the training data is randomly upsampled. For evaluation, to give an accurate evaluation of what would happen in production, the testing data is not upsampled.

There are a few limitations to using this data for this task. 1) The training data is limited. 2) Ideally, the labels for "relevant" or "irrelevant" should be clear and objective. The reality was not so simple. For example, "*Britney Spears hit up a shopping mall near her home the day before Thanksgiving*" matched to a Wikipedia article on "*Britney Spears conservatorship dispute*". Though the article is not directly related to the headline, it does provide helpful context for why Britney Spears is in the news. Therefore the labeling is somewhat subjective.

4 Methodology

The most basic text-based recommender system is based on text similarity. Given a document, recommend another document with high similarity. Lexical similarity and semantic similarity methods have been well-documented. Lexical similarity methods are limited to comparing exact words. They would not capture the fact that "I saw my doctor today" and "I had an appointment with my physician" are similar sentences. Semantic similarity, on the other hand, requires that the sentence be encoded into a vector that incorporates the similarity between words "doctor and "physician" by way of pre-trained embeddings. Semantic similarity, rather than lexical similarity, is a necessary part of this recommendation pipeline.

However, semantic similarity on its own is not enough to deliver optimal recommendations. 1) Very little if any weight should be placed on sentence structure similarity. 2) Some words or phrases in the sentence will be more important than others. But doing a simple word count is not enough. A word or phrase may only appear once,

but be a very good starting point for finding useful context. Conversely, a word that appears many times may not help find context. 3) Finally, a word or phrase that is used to find a contextual match may be too commonplace or general to require context, like "*Biden pledges no new lockdowns for Americans*" matches to an article on "United States of America". Such results should be filtered out.

The recommender system has two parts: 1) match news articles to Wikipedia articles. 2) filter out the irrelevant Wikipedia articles. Part I has already been built and the training data has been generated; the focus of this paper is to explore the embedding and classification methods detailed in part II. The experiment is limited to lightweight methods that can be run within a few minutes on an 8GB RAM computer.

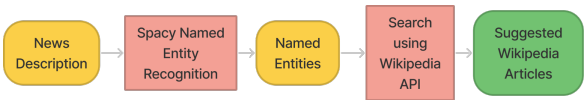

Fig 1. Pipeline Part I: Generate Recommendation Options

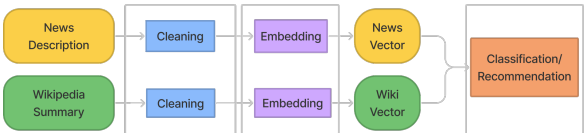

Fig 2. Pipeline Part II: Classify Recommendations as "Relevant" or "Irrelevant"

1) Text Cleaning and TD-IDF. Removing punctuation and stop words is necessary before embedding. Our initial list of stop words includes Apache nltk's list of English stop words and several news publication terms, since we do not want a headline published by NBC to return a Wikipedia result for NBC. This alone is not enough to address the problem that some phrases are too commonplace to need context. So we also apply TD-IDF. TD-IDF is fitted to the training news data, then used as a vectorizer that weighs each token in each document in both the training and test data. The tokens with the bottom 5 weights in each document are removed. An example of the cleaning process is included in Appendix Table A1.

2) Embedding. Before the recommendation, both the baseline news description and the Wikipedia description have to be converted into vectors. We try three methods. The first method simply counts each word's frequency in each of the documents. The second method uses a lightweight Glove model (Pennington 2014) that was pre-trained on a Wikipedia corpus. This returns a vector of 50 features. The third method uses Google's Universal Sentence Encoder (Cer 2018) which has been pre-trained on a variety of corpora on a variety of tasks including classification, question answering, and others.

Figure 3. compares a vectorized news article to three vectorized Wikipedia articles. The text is included in Appendix Table A2. The first Wikipedia article represents the best relevant match; the second is a potential false positive; an article with similar words, but is not about the same thing. The third article is completely irrelevant. Two articles that are perfectly similar should show points aligned at the 45-degree line. Two articles that are completely different should show points lining each axis. The title also shows the cosine similarity score between each pair of vectors. Overall, it looks like the Google encoder has the widest variation in the similarity scores, which is what we want.

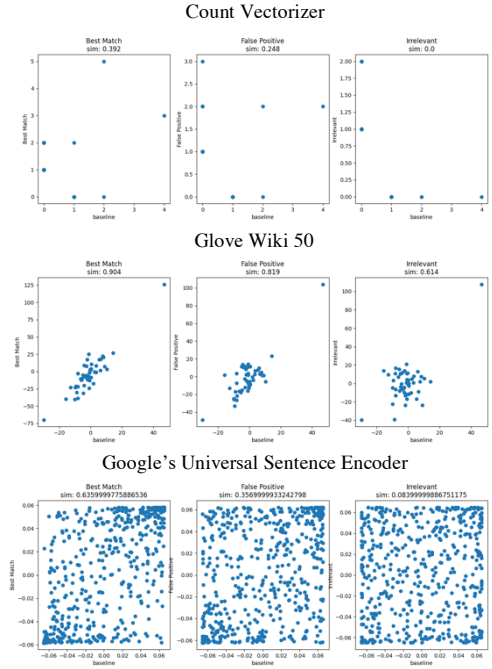

Fig 3. Embedding Similarity Methods

The intuition is that the Count Vectorizer is likely too simple. Whereas both the Glove and Google embeddings may be good candidates. The question is whether a word-level encoder will do better than a sentence-level encoder. The sentence encoder will likely generate many features that are not important for the classification task. However, that can be handled with a machine learning model in the next step.

3) Classification/Recommendation. The output of the embedding step is a pair of vectors that represent the news document and the

Wikipedia document. To get a prediction, you could calculate cosine similarity between the vectors and label "relevant" if the similarity measure is above a certain threshold. The threshold may be different depending on the dataset. We iterate over a few thresholds to find the threshold which maximizes F1 score and makes intuitive sense.

The limitation to the cosine similarity measure is that technically, each feature is considered equally. We calculate the difference between the vectors as features for a classification machine learning model. The model can learn to apply different weights to features which are more important to determining relevance. We apply a Logistic Regression and a Support Vector Classifier model with the requisite preprocessing (up sampling and normalization).

5 Results

Embedding	Training	Testing
Word frequency counter + similarity cutoff threshold of 0.3	Accuracy: 0.54 Sensitivity: 0.991 Specificity: 0.089 F1: 0.163	Accuracy: 0.704 Sensitivity: 0.928 Specificity: 0.091 F1: 0.141
Glove-wiki-gigaword-50 + Logistic Reg	Accuracy: 0.689 Sensitivity: 0.654 Specificity: 0.724 F1: 0.7	Accuracy: 0.551 Sensitivity: 0.641 Specificity: 0.303 F1: 0.265
Universal sentence encoder + Logistic Reg	Accuracy: 0.993 Sensitivity: 0.993 Specificity: 0.994 F1: 0.993	Accuracy: 0.628 Sensitivity: 0.691 Specificity: 0.455 F1: 0.395
Glove-wiki-gigaword-50 + SVM	Accuracy: 0.839 Sensitivity: 0.801 specificity: 0.877 F1: 0.845	Accuracy: 0.733 Sensitivity: 1.0 Specificity: 0.0 F1: 0.0
Universal sentence encoder + SVM	Accuracy: 0.96 Sensitivity: 0.959 Specificity: 0.961 F1: 0.96	Accuracy: 0.765 Sensitivity: 0.873 Specificity: 0.47 F1: 0.517

Table 5: Font guide.

As shown, the Universal sentence encoder with a Support Vector Classification model performs the best, using the F1 score as the final evaluation measure. A web app using this model is published on GitHub.

6 Discussion

We have discussed the limitations of lexical similarity and considered the use case of semantic similarity through embedding within a recommender pipeline. In our experiment with semantic similarity methods, the sentence-based encoding performed better than the word-based encoding.

There are a few ways in which this part of the pipeline can be improved. 1) More varied training data and more testing over time would be beneficial. 2) More embedding methods can be experimented with, especially embedding methods which require more memory. 3) TD-IDF can be fitted to a larger corpus of current events and the application of TD-IDF as a stop word remover can be explored more in-depth (recall that here we simply remove the lowest weighted 5 tokens from each document). 4) More interpretation should be done on why false positives or false negatives occur. We did not analyze the feature importances of the machine learning models. No experimentation was done on how characteristics of the text itself influence prediction, for example, the size of the documents and the distribution of parts of speech.

We believe this type of pipeline is something for online media outlets and content providers to consider. These recommendations could be a helpful step in combatting misinterpretation and misinformation.

References

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Céspedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, Ray Kurzweil. Universal Sentence Encoder. arXiv:1803.11175, 2018.

Dhivya Chandrasekaran and Vijay Mago. 2021. Evolution of Semantic Similarity - A Survey. ACM Comput. Surv. 54, 2, Article 41.

M. Moerland, F. Hogenboom, M. Capelle, and F. Frasincar. 2013. Semantics-Based News Recommendation with SF-IDF+. Third International Conference on Web Intelligence, Mining and Semantics (WIMS 2013). ACM, 2013.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. Conference: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)

Appendix

Raw Text: *"Biden warns against Omicron panic, pledges no new lockdowns - President Joe Biden urged Americans on Monday not to panic about the new COVID-19 Omicron variant and said the United States was making contingency plans with pharmaceutical companies if new vaccines are needed."*
Cleaned Text: biden warns omicron panic pledges new lockdowns president joe biden urged americans monday panic new covid 19 omicron variant said united states making contingency plans pharmaceutical companies new vaccines needed
Cleaned Text after TD-IDF: biden warns pledges new president joe biden urged americans monday new covid 19 variant said united states making plans pharmaceutical companies new vaccines

Table A1: Example of text cleaning and TD-IDF.

News Document		
Ohio State vs. Michigan State score, takeaways: Buckeyes obliterate Spartans, make strong playoff statement - Ohio State tore through Michigan State's defense on its way to an impactful win		
Best Match	False Positive	Irrelevant
"*The Ohio State Buckeyes football team competes as part of the NCAA Division I Football Bowl Subdivision, representing The Ohio State University in the East Division of the Big Ten Conference. Ohio State has played their home games at Ohio Stadium in Columbus, Ohio since 1922.*"	"*Ohio ((listen)) is a state in the Midwestern region of the United States. Of the fifty states, it is the 34th-largest by area, and with a population of nearly 11.8 million, is the seventh-most populous and tenth-most densely populated.*"	"*The Planetary Society is an American internationally-active non-governmental nonprofit organization. It is involved in research, public outreach, and political space advocacy for engineering projects related to astronomy, planetary science, and space exploration.*"

Table A2: Text used to generate Figure 3.