Optimal Model Input for Newspaper Topic Classification

Carmen Easterwood

W266 Natural Language Processing with Deep Learning

Summer 2018

Abstract

Topic classification is an important method of organizing text content and making it more easily searchable, but in the newspaper setting the question remains: what exactly should we be feeding into topic classification models? In this paper, I test five different model inputs: full text, lead paragraph, headline, nouns-only version of the full text, and lemmatized full text. In general I find that inputs with more words perform better, but that a larger vocabulary isn't necessarily better. While nouns make up only 27% of the words and 61% of the vocabulary in a newspaper article, they perform almost as well as the full text, making them possibly the most efficient model input.

1 Introduction

Topic classification is an important form of content analysis that has been widely studied in the NLP community. Tagging documents by topic makes searches more efficient, so users can find the information they're looking for more quickly and without having to sort through unrelated documents. Unfortunately, manual topic-coding on a large scale is quite time-consuming, so automating this process can save both time and money for organizations with a large amount of text data.

In this paper, I will look at topic classification specifically in the newspaper context. In addition to making searches more efficient for news consumers and researchers, classifying news articles by topic also helps us understand what journalists are writing about, whether that is changing over time, and the relative importance of different topics. Since many news organizations have seen

declining revenue over the last decade (Barthel, 2017), a cost-saving measure like automation of topic-coding would likely be welcome. However, in the existing literature there is no agreement on what specifically should be fed into topic classification models to get the best results. In this paper, I develop three supervised topic classification models and test several possible model inputs drawn from New York Times articles. For each model, my main focus is how the model inputs perform relative to each other, and which input gives the best result in each setting.

2 Background

As mentioned, there is a large literature looking at topic classification in many contexts, but only a few papers have focused on classifying newspaper articles. Most recently, Martin and Johnson (2015) found that using a nouns-only or lemmatized version of the articles improved topic classification on a dataset of 90,000 articles from the San Jose Mercury News. However, they were working on an unsupervised model, so they were also relying on the model to generate the topics from the words it received. Martin and Johnson evaluated their model using techniques that are not applicable in a supervised setting, so their results are not directly comparable to mine.

Previously, Wermter and Hung (2002) proposed a semi-supervised self-organizing memory (SOM) model for topic classification on 100,000 Reuters news articles. The authors were able to achieve accuracies around 90% for both full text articles and article headlines, and for the full text articles they were able to bump the accuracy up to 95% by using WordNet relationships to assist the model (which limited the model input to only the nouns and verbs found in WordNet). Crucially, however, the authors only tested their model on the eight

most common topics in their dataset and did not consider articles from other topics. To be useful outside of an academic setting, a topic classification model needs to be able to handle more different types of articles.

Other recent papers have looked at supervised topic classification for non-news documents in various settings. Karan et al. (2016) discusses classification of Croatian political texts using logistic regression (one-vs-rest), Gaussian Naive Bayes, and gradient-boosted trees, as well as utilizing some post-processing rules, to reach an F1 score of 77% on 22 topics. Their model included lemmas and bigrams as inputs. Meanwhile Glavaš et al. (2017) looks at similar political texts in multiple languages, and finds that CNNs perform best for monolingual English models with an accuracy of 57%. In this paper, I will extend some of the techniques used in these cited works to the news article setting.

3 Methods

3.1 Dataset

For this analysis I used the New York Times Annotated Corpus, which contains 1.8 million articles from 1987-2007, labeled with topics, subtopics, and newspaper sections. Due to computing power limitations, I have selected a random sample of 100,000 articles to parse and use as data for the model. The articles are randomly split so that approximately 75% are training data, 5% are development data, and 20% test data.

3.2 Model Input

I test the article's full text and four additional model inputs. Three are pulled from my review of the literature, and I have added the fourth based on my understanding of the structure of news articles.

1. **Full text** of article

2. Article's **lead paragraph**, which is supposed to hook the reader, and in a news context often summarizes important details of the story (Bloch, 2016)

3. Article's **headline** (Wermter and Hung, 2002)

4. **Nouns-only** version of the full text (Martin and Johnson, 2015). Created using NLTK's part-of-speech tagger.

Model Input	Training Words		
	Count	**% Full Text**	**Average**
Full Text	49.5M	100%	668
Lead Paragraph	7.6M	15%	105
Headlines	0.6M	1%	8
Nouns	13.2M	27%	178
Lemmas	50.4M	102%	682

Model Input	Training Vocab (Unique Words)		
	Count	**% Full Text**	**Average**
Full Text	327K	100%	276
Lead Paragraph	140K	43%	67
Headlines	42K	13%	7
Nouns	198K	61%	111
Lemmas	237K	72%	253

Table 1: Model Inputs

5. **Lemmatized** version of the full text (Martin and Johnson, 2015). Created using NLTK's lemmatizer.

Table 1 shows some key statistics for each type of model input. The full text for the training articles (approximately 75,000 articles) adds up to about 50 million words and a vocabulary of over 300,000 words. The lemmas have roughly the same number of words, but much fewer unique words, with a vocabulary size only 72% of the full text vocabulary. Then there is a big drop down to the nouns, which have only 27% of the total words that full text articles have, but 61% of the vocabulary. As expected, headlines are the smallest category with less than a million words total, and the average individual headline having 8 words.

3.3 Model Output

The NYT corpus has multiple ways of classifying articles. Table 2 shows four examples from the data of the four different ways the articles were classified. In my models I use the `desk` column as my model output variable for two reasons:

1. It has the lowest percentage of null values (0.4%)

2. It never assigns multiple descriptions to the same article

However, `desk` still requires some clean-up before use due to misspellings and possible changes to the desk names over time. Table 3 shows some examples of how I cleaned the `desk` variable so the model would have a standardized set of output labels for the articles. For example,

#	Desk	General Descriptor	Online Sections	Taxonomic Classifier
1	Foreign Desk	• Immigration and Refugees • Jews • Music • Religion and Churches	• World	• Top/News/World/Europe • Top/News/World/Countries and Territories/Austria • Top/Features/Arts/Music • Etc. (8 others)
2	Book Review Desk	• Books & Literature	• Arts • Books	• Top/Features/Arts • Top/Features/Books • Top/Features/Books/Book Reviews
3	Classified		• Paid Death Notices	• Top/Classifieds/Paid Death Notices
4	Style Desk	• Computers and the Internet • Labor	• Technology • Style • Opinion	• Top/News/Technology • Top/Features/Style/Fashion and Style • Top/Opinion/Opinion • Etc. (2 others)

Table 2: Possible Model Outputs

Clean Label	Original Label	Count
book review	Book Review Desk	1,750
	Book Review Dest	1
business & financial	Business Desk	4
	Business World Magazine	8
	Business/Financial Desk	6,078
	Business/Financial Desk;	1
	Business/Financial desk	1
	Business/FinancialDesk	5
	Business\Financial Desk	2
	E-Business	6
	E-Commerce	17
	Financial Desk	11,276
	Financial Desk;	16
	Money & Business/Financial Desk	3
	Money and Busines/Financial Desk	1
	Money and Business/Financial Desk	939
	Moneyand Business/Financial Desk	1
	Small Business	12
	SundayBusiness	82
	The Business of Green	3
cars	Automobiles	122
	Cars	28

Table 3: Label Cleaning Examples

there are significant clusters of articles around "Business/Financial Desk", "Financial Desk", and "Money and Business/Financial Desk" (plus misspellings of each of these), and these are all converted to have a single label of "business & financial".

After label cleaning, roughly 95% of articles fall into the top 20 categories, and then there are a large number of very small categories with just a few articles. I therefore created an <OTHER> category as a catchall for these tiny categories, which individually don't have enough data for the models to learn very well. Figure 1 shows the frequency distribution of the top 20 category labels, plus <OTHER>. The top 8-10 categories are "usual suspects" for newspaper sections (e.g. business, sports, classifieds, editorials), and then we see some smaller categories like book reviews and a couple of weekly sections that are unique to the New York Times (New Jersey weekly, the city weekly, Westchester weekly, Long Island weekly).

3.4 Models

For each model input, I tested two simple baseline models and a neural network:

1. Multinomial naïve Bayes

2. Multi-class logistic regression (one vs. rest)
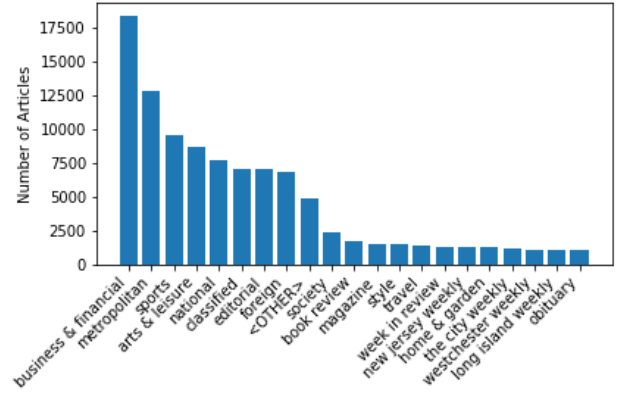
3. Convolutional neural network

Figure 1: Frequency of Top 20 Labels + <OTHER>

3.4.1 MNB and LR

For the multinomial naive Bayes and logistic regression models, I TF-IDF vectorized the model input, dropping any words with a document frequency count of one. This significantly reduced the vocabulary sizes and got rid of a lot of misspellings and one-off words that just added extra noise to the models.

In each of these models I tested multiple parameter values on the dev data, and chose the parameter values that resulted in the highest accuracy. Figures 2 and 3 show how the models performed with different parameter values, and reveal some interesting comparisons between the five model inputs that I will discuss further in Section 4.

3.4.2 CNN

When applying a convolutional neural network to this data, I found the main challenge was managing the size of the model, given how large the vocabularies are and how long some articles can be.

1. Initialized the model with GloVe word embeddings (Pennington et al., 2014)

2. Limited the vocabulary to the same size as the TF-IDF vocabulary (see Section 3.4.1)

3. Padded each model input to the 90th percentile of lengths for that model input

I chose this padding method because the length distribution for each model input is extremely right-skewed (e.g. the 90th percentile of full texts has 1,225 words, while the longest full text has over 9,000 words). It didn't make sense to pad model inputs out to the length of the longest article, since most articles are nowhere close to
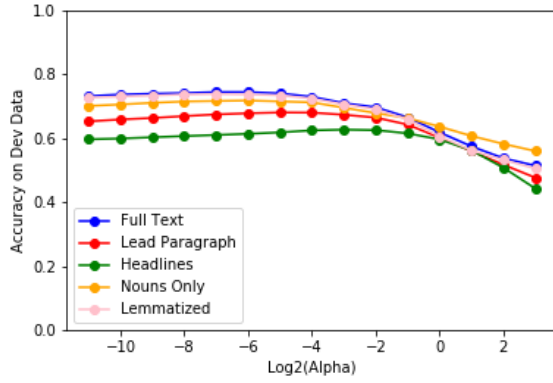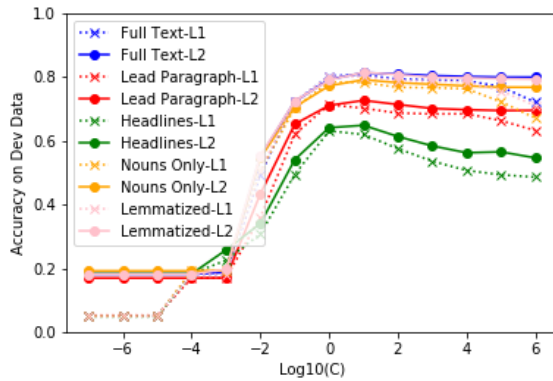
Figure 2: MNB Accuracy on Dev Data

Figure 3: LR Accuracy on Dev Data

	Model Type			
Model Input	MNB	LR	CNN	Best Model
Full Text	**0.**	**0.**	0.	X
Lead Paragraph	0.	0.	**0.**	X
Headlines	0.	0.	0.	X
Nouns	0.	0.	0.	X
Lemmas	0.	0.	0.	X
Best Input	X	X	X	**X**
	X	X	X	**in X**

Table 5: Accuracies on Test Data

	Model Type			
Model Input	MNB	LR	CNN	Best Model
Full Text	**0.**	**0.**	0.	X
Lead Paragraph	0.	0.	**0.**	X
Headlines	0.	0.	0.	X
Nouns	0.	0.	0.	X
Lemmas	0.	0.	0.	X
Best Input	X	X	X	**X**
	X	X	X	**in X**

Table 6: Weighted F1 Scores on Test Data

An interesting future project would be to run an unsupervised topic classification algorithm on this corpus (e.g. Martin and Johnson (2015) use the Latent Dirichlet Allocation algorithm) and compare those results to the supervised learning results.

that length, and the model would be unnecessarily large. I therefore sacrificed the data at the ends of the longest articles in the hope of having a model I could actually train with the number of articles I am using.

4 Results & Discussion

Note: (Wermter and Hung, 2002) uses only top 8 topics.

5 Conclusion

No time for: SVM, gradient boosted trees

Model Input	Vocabulary	90% Length	Padding
Full Text	129,165	1,225	500
Lead Paragraph	63,733	135	135
Headlines	19,724	10	10
Nouns	97,557	374	374
Lemmas	113,923	1,239	500

Table 4: CNN Vocabulary and Padding Sizes

References

Michael Barthel. 2017. Despite subscription surges for largest U.S. newspapers, circulation and revenue fall for industry overall. *PewResearch.org*.

Hannah Bloch. 2016. A good lead is everything – here's how to write one. *NPR.org*.

Goran Glavaš, Federico Nanni, and Simone Paolo Ponzetto. 2017. Cross-lingual classification of topics on political texts. In *Proceedings of the Second Workshop on Natural Language Processing and Computational Social Science*, pages 42–46. Association for Computational Linguistics.

Mladen Karan, Jan Šnajder, Daniela Širinić, and Goran Glavaš. 2016. Analysis of policy agendas: Lessons learned from automatic topic classification of croatian political texts. In *Proceedings of the 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences and Humanities (LaTeCH)*, pages 12–21. Association for Computational Linguistics.

Fiona Martin and Mark Johnson. 2015. More efficient topic modelling through a noun only approach. In *Proceedings of Australasian Language Technology Association Workshop*, pages 111–115.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Stefan Wermter and Chihli Hung. 2002. Selforganizing classification on the reuters news corpus. In *Proceedings of the 19th international conference on computational linguistics - Volume 1*.