Author Attribution for Three Literary Texts With TF-IDF and GloVe

Language Technology, Spring 2021
Professor: Kyle Gorman
Author: Andrew Kirby

Abstract

Author attribution continues to be a meaningful area of study for the field of Computational Linguistics. It has many areas of application, ranging from online behavior to determining who in actuality wrote important historical and cultural texts. It is possible to use quantitative methods to make these attributions at a reliable rate of accuracy. Newer methods of creating word vectors and classifying them allow for better predictions. Although literature may be a helpful source of data, the findings can be generalized to other areas of study and used to solve other problems in the world.

Introduction

Determining who an author of a text ranges in applicability from the esoteric to the practical, and computational methods are increasingly becoming a part of that task. One example is the ongoing debate of who wrote Shakespeare's plays, and whether it was by a William Shakespeare or some other person from that time (Gabrielson 2013). Maybe it was not actually Christopher Marlowe after all, if function words and Part of Speech tags have anything to tell us (Zhao & Zobel 2006). Another more practical application involves tracking down a troll through that person's writing patterns on social media. An example of this consists of the Russian 'troll farms' that received much attention in the news following the 2016 election (Ghanem et al., 2019). Another possible application concerns plagiarism detection, especially relevant given the preponderance of online education models in the wake of COVID-19.

For all three of these cases, similar Natural Language Processing (NLP) tools persist throughout. Quantitative methods are necessary to develop because the sheer quantity of information and text present online is so large that it is highly impractical to employ a legion of human readers to make discernments and judgments. These tools play an ever more important role behind the scenes of our lives. For example, if someone were to make a threat online, and an NLP system misattributed that threat to another uninvolved person, lives could be permanently altered in an unjust way. On the other hand, these tools could help to reduce the proliferation of such harmful behavior, making the virtual world a safer place for all.

The goal of this project is on the lighter side: to take three texts, compare them using different types of word vectors, and use a classifier to make a prediction on the origin of other pieces of text within the set. To do so, the program will need a way to organize and interpret the text alongside a method for determining which is which, based on this information.

Text Processing Techniques

Global Vectors for Word Representation (GloVe) is an unsupervised learning algorithm that allows for obtaining vectors of words. It uses Euclidean distance to compare similarities of words in its model, and has many uses, such as for classification tasks or measuring similarity of documents (Pennington et al., 2014).

Common approaches of processing text, changing it into numbers that a program can understand, consist of using Bag of Words (BoW) and TF-IDF (Term Frequency-Inverse Document Frequency). The Bag of Words approach consists of counting up all the words and putting them into vectors so that they can be computed into other forms of information. TF-IDF takes this a step further, by giving less importance to words that occur very frequently while not having a lot of useful content, like 'the' and 'a' or 'it'. It also takes into account the quantity of documents in a document set in order to determine which words get more value (Eisenstein 2019, Ch. 2). The TF-IDF approach was introduced by Spärck Jones (1972).

Sentence Boundary Detection (SBD) involves figuring out where one sentence ends and another begins. While this may seem like an easy task, completed by looking for periods, question marks, or exclamation points, it becomes rapidly more complex once text begins to mention money, abbreviations, or other artifacts of punctuation. Methods exist that are rule-based, or that can make predictions after being trained on a model with the power of machine learning (Sanchez 2019).

A Convolutional Neural Network (CNN) is a way of making predictions through a series of linear probabilities put together in a series of big matrices or layers. While CNNs are typically known to be useful for image-related machine learning tasks, (with Recurrent Neural Networks favored for text), they can also be helpful for classification of text as well (Albawi et al., 2017).

For classification, the Support Vector Machine (SVM) and Logistic Regression (LR) are popular, especially for binary classification. The SVM works by increasing the dimensions of a training vector and then classifying using a line (Henrique et al., 2019). LR is used with binary dependent variables, which have two values as in 0/1 or True/False. As a side effect of classification, it estimates the posterior probability of true and false labels. using a scoring function for base features (like word vectors) and a binary label (Eisenstein 2019, Ch. 2).

Past Work

One system, based on texts from Project Gutenberg, relies on the following set of features: mean token log frequency, a ratio of the number of unique words to the number of tokens, an average of syllables per sentence, the number of sentences per text, and a mean phonetic sonority score

per text. It used word2vec to compute other semantic features like disgust or surprise, and a Multi-Layer Perceptron for classification (Jacobs & Kinder 2020).

One system distinguishes between ancient Greek prose and verses. It accomplishes this through a stylometric set of features: These features include types of pronouns, conjunctions and particles, subordinate clauses, and other sentence characteristics. It used NLTK for tokenization, took counts and n-grams of these features, and used scikit-learn's `RandomForestClassifier` for classification (Gianitsos et al., 2015).

Another system works to distinguish between the spoken dialogue of one character in a story from another in English language plays from the 19th and 20th centuries. It obtained word vectors using a word2vec algorithm trained on a Wikipedia corpus as well as with TF-IDF. It also used Sparse Additive Generative (SAGE), a generative word vector model which uses log deviation for computational efficiency (Eisenstein et al., 2011). It integrated these vectors with lexical polarity features such as concrete vs abstract, subjective vs objective, and colloquial vs literary. For classification, it used LR and Support Vector Machines (SVM) (Vishnubhotla 2019).

One review details the variety of characteristics that can be integrated into attributive analyses of written texts. These characteristics, ranging from syntactic, lexical, semantic features, along with the characters (as in letters in words, not story people) themselves. At the time of that review, most systems relied on n-gram models and SVMs for classification (Stamatatos 2009).

Methodology

The corpus in this project consisted of three literary texts obtained online as converted .txt files from zlibrary.[1] The texts are: *The Perfect Crime*, by Jean Baudrillard, *Julian*, by Gore Vidal, and *Oblivion*, by David Foster Wallace. *The Perfect Crime* is a nonfiction philosophy text, *Julian* is a historical fiction novel, and *Oblivion* is a set of contemporary fiction short stories.

sents	class
In the end, it is perhaps more a sphinx than a bitch.	1
I realize I am not a master of oratory, but after all what I have to say and the way in which I say it is—and I do not mean this immodestly -of obvious interest to the world.	2
She was also all but incapable of writing a simple declarative sentence and thus could not, by any dark stretch of the imagination, ever be any kind of rival for Atwater's salaryman position at Style.	3

Figure 1: Data sample of sentences and their respective classes. 1 = *The Perfect Crime*, 2 = *Julian*, 3 = *Oblivion*

[1] Zlbrary (link changes periodically): https://1lib.us/

For preprocessing, *The Perfect Crime* required elimination with regular expressions of page numbers and other non-contentful artifacts such as -- NA --. *Julian* and *Oblivion* required minimal processing. Afterward, sentence boundary detection was done with `DetectorMorse`, and the sentences were passed into each of their own cells in a dataframe.[2] While *The Perfect Crime* and *Oblivion* contained roughly 3200 sentences each, *Julian* was much longer with around 10,000 sentences. Each book's sentences were shuffled independently of one another, and the first 3000 sentences were taken. Each group of 3000 sentences was assigned a class of 1, 2, and 3, respectively. Both methods used an 80:20 training and testing split.

Text vectors for BoW and TF-IDF were obtained via scikit-learn's `CountVectorizer` `TfidfTransformer`, respectively. [3] For both of these, classification was done with a multinomial version of scikit-learn's `LogisticRegression`, with the following settings: *penalty='l2', n_jobs=1, multi_class='multinomial', C=1.0, solver ='newton-cg'*.

Pre-trained GloVe embeddings were created with TensorFlow-based Keras.[4] This process consisted of a series of steps. First, the vocabulary from the corpus was indexed and mapped to these indices. Only the top 20,000 words were considered. Then, the pre-trained GloVe embeddings were loaded, and a matrix from it that corresponds with the vocabulary from the corpus. These were loaded into an embedding layer. Classification was done with a one-dimensional convolutional neural network, with an *relu* activation for the layers and a *softmax* activation for the classifier. This model was run for 25 epochs.

Finally, a beta model with 9000 data points and three classes was created. Each of these data points consisted of groups of five integers numbering one through five. The data was then shuffled. Each class contained 3000 data points.

[2] `DetectorMorse`: https://github.com/cslu-nlp/DetectorMorse

[3] BoW: link TF-IDF: link

[4] Modified from this Keras tutorial: https://keras.io/examples/nlp/pretrained_word_embeddings/#build-the-model GloVe downloaded from: https://nlp.stanford.edu/projects/glove/

Results

	Logistic Regression				
	BoW UNIGRAM	**BoW BIGRAM**	**TF-IDF UNIGRAM**	**TF-IDF BIGRAM**	**BETA MODEL**
Median:	0.805	0.698	0.805	0.7	0.318
	CNN				
	KERAS GloVe				
Median:	0.929				

Figure 2: accuracy scores from each of the models.

The GloVe model outperformed the TF-IDF and BoW models. The GloVe model had an accuracy of 0.929, and the unigram BoW and TF-IDF both had an accuracy of 0.805. These scores were taken from a median of three runs with seeds 125, 150, and 333. The beta model's accuracy was 0.318, showing that the other models had significant effectiveness in comparison.

Discussion

The beta model shows a contrast between the use of text vectors, rather than making predictions based on meaningless features. The GloVe model performed the best, but also took much more computing power to run. Running the GloVe model for longer than 25 epochs produced no increase in accuracy (tested up to 50). The multinomial logistic regression worked best with TF-IDF and BoW unigrams, and was faster than all other methods. While some other models reviewed above achieved accuracy scores in the upper 90s with significantly more sophisticated linguistic features, neural networks were not a large part of their classification mechanisms.

Conclusion

While having powerful classification and text vector mechanisms goes a long way, it is painfully evident that the inclusion of additional specific linguistic features could push decent results to become great results.

Future work could include features such as metaphor detection, measurements on the colloquiality of speech, tabulations of morphological or phonetic features, or the integration of semantic properties. It could also be helpful to look for properties idiosyncratic to each text, based on qualitative literary analyses on them. For example, David Foster Wallace always pursued a sort of maximalist writing style, and mathematics played a large part in how he approached his prose. Finding a quantitative connection to his style could go a long way toward refining methods of author attribution.

“Who wrote that?” will become a more and more important question as more of the world’s events take place virtually. Yes, NLP systems that work with author attribution can help with targeting advertisements or figuring out who that guy was that made fun of you in a really rude way after defeating you in an online game. They could help us learn more about our history, our culture, and ourselves.

References

Eisenstein, J. (2019). *Introduction to Natural Language Processing*. MIT Press.

Eisenstein, J., Ahmed, A., & Xing, E. P. (2011). Sparse additive generative models of text. In *Proceedings of the 28th international conference on machine learning (ICML-11)* (pp. 1041-1048).

Ghanem, B., Buscaldi, D., & Rosso, P. (2019). TexTrolls: Identifying Russian trolls on Twitter from a textual perspective. *arXiv preprint arXiv:1910.01340*.

Gianitsos, E., Bolt, T., Chaudhuri, P., & Dexter, J. (2019). Stylometric classification of Ancient Greek literary texts by genre. In *Proceedings of the 3rd Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature* (pp. 52-60).

Jacobs, A. M., & Kinder, A. (2020). Quasi Error-free Text Classification and Authorship Recognition in a large Corpus of English Literature based on a Novel Feature Set. *arXiv preprint arXiv:2010.10801*.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).

S. Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), 2017, pp. 1-6, doi: 10.1109/ICEngTechnol.2017.8308186.

Sanchez, G. (2019, June). Sentence boundary detection in legal text. In *Proceedings of the Natural Legal Language Processing Workshop 2019* (pp. 31-38).

Spärck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation, 28,* 11–21.

Stamatatos, E. (2009). *A survey of modern authorship attribution methods. Journal of the American Society for Information Science and Technology, 60(3), 538–556.* doi:10.1002/asi.21001

Vishnubhotla, K., Hammond, A., & Hirst, G. (2019). Are fictional voices distinguishable? classifying character voices in modern drama. In *Proceedings of the 3rd Joint SIGHUM*

Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature (pp. 29-34).

Zhao, Y., & Zobel, J. (2007, January). Searching with style: Authorship attribution in classic literature. In *ACM International Conference Proceeding Series* (Vol. 244, pp. 59-68).