Progress Report: Author Attribution

Remy LeWinter
Joe Song
CS4120: Natural Language Processing Fall 2020
Northeastern University
lewinter.r@husky.neu.edu
song.jo@husky.neu.edu

Changes

Our original proposal centered on reproducing a 2003 approach to authorship attribution using character level n-grams to build author profiles [1]. We have abandoned this approach in favor of a more modern methodology based on a 2018 paper from Helena Gómez-Adorno, Juan-Pablo Posadas-Durán, Grigori Sidorov, & David Pinto [2]. We will still use the same data source as originally proposed [3].

While considering ways to update our project, we decided to avoid approaches using state-of-the-art deep learning with finely tuned convolutional and recurrent neural networks, although these are some of the more common and highly performing models today. This decision is due to our lack of proper instruction or prior experience in implementing neural nets and to the limits of our computer hardware, which struggled to handle basic multilayer perceptrons.

Our new approach is still modern, while also including the use of n-grams, linking it somewhat to our original proposal. Our methodology will somewhat differ from that in [2] because we use a dataset with almost 20,000 authors as opposed to the only 13 authors in [2]. We will also likely experiment on different choices of parameters, as informed by our data set and the computational limits of our computers.

Data Preprocessing

Our data [3] comes as a set of 19,320 files each containing the full content of a profile from bloggers.com in XML format. The website hosting the data source [4] reports an average of "approximately 35 posts and 7250 words per person." Extrapolating upon this, we can expect approximately 207 words per post. Examination of the data reveals that some users tend to make tweet-like posts of at most a few short sentences, while others tend to write paragraphs or even essays.

Before sampling for our training and test data, we remove and set aside profiles with fewer than 5 posts or fewer than 1000 words total. We shuffle the order of posts in each author's profile and separate 20% of each profile for testing data. We collect the vocabulary and word frequency for the entire dataset. The preprocessing stage outputs json files for training data, test data, left out data, and vocabulary/word frequency. G-zipped preprocessed data files are available at https://bit.ly/2UalCgj.

Method

Our methodology follows the same general sequential flow as Gómez-Adorno et al. (fig. 1) [2]. We train 3 paragraph vector models (doc2vec from the Gensim library [5]) using n-grams of characters, words, and POS tags. Word tokenization and POS tagging are performed with Spacy [6]. One vector is produced for each blog post by each doc2vec model. The output vectors from each model are then concatenated in different combinations for each document and used as input to a multinomial Bayesian classifier for assigning new documents to previously seen authors.

While [2] uses a multinomial logistic regression classifier, which has the advantage of not assuming independence among prediction features, this approach is heavily time-consuming for problems with a very large number of classes to learn. Bayesian classifiers on the other hand scale much more nicely for large numbers of classes.

Gómez-Adorno et al. use n-grams without overlap, claiming, "we suppose that the overlapping effect is not dominant since the size of the window used to create the embedding is high," while also reporting on the same page that, "the best results are obtained by appending the document embeddings of different n-gram sizes" [2, p. 748]. We use overlapping n-grams and do not concatenate vectors from different n-gram sizes to reduce the number of parameter combinations we need to run.

Results and Evaluation

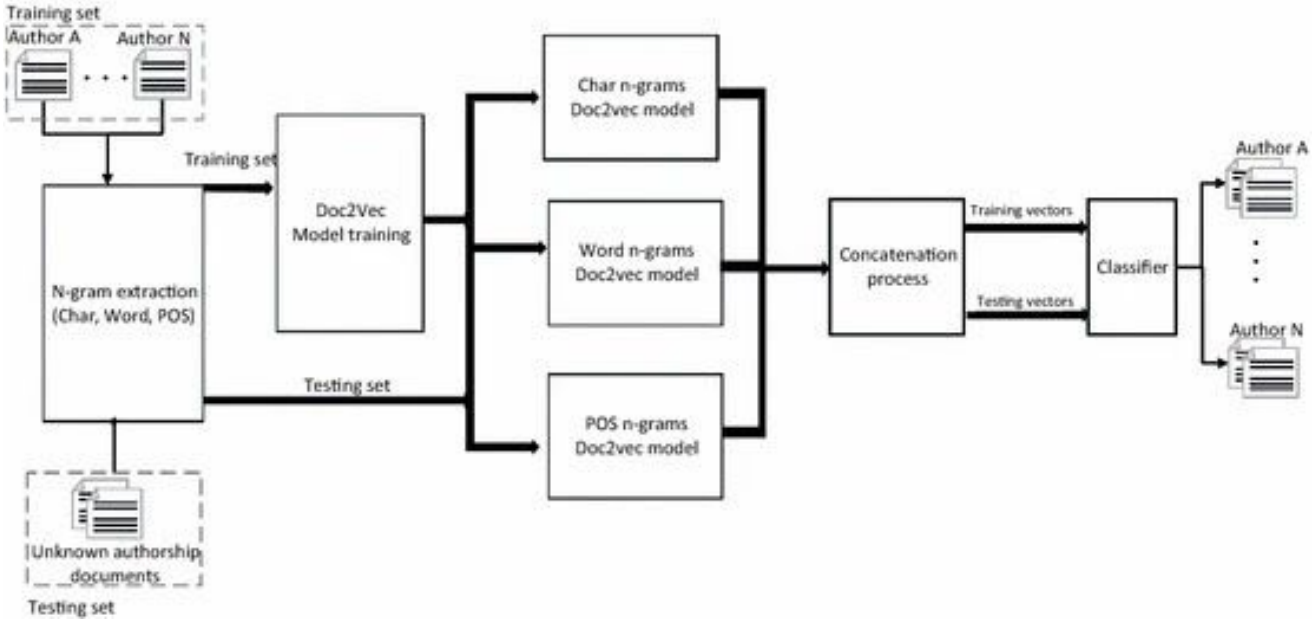
We have completed pre-processing our data and are working on n-gram extraction. Our training set contains 529,488 posts, while our test set contains 125,990 posts. We are beginning by producing 3-grams to train models that produce vectors with 50 features to give to the classifier both on their own and concatenated together. The idea is to have a comparison of classification accuracy with parameters that should allow our code to complete in a reasonable time before trying larger parameters.

Next Steps

Our next steps are to ensure that our extracted n-grams are properly read by doc2vec and to train those models. We will experiment with different lengths of n-grams, combinations of vector types, and lengths of output vectors from doc2vec. We can also experiment with different cutoffs for data inclusion, different splitting of test/training, shuffling of post order, or with leaving chronological post order intact to better simulate a classification problem with new data arriving in real time.

Figures

Fig. 1



References

[1] V. Keselj, F. Peng, N. Cercone, and C. Thomas. 2003. N-gram-based author profiles for authorship attribution. In *PACLING'03* 255-264 Halifax, Canada.

[2] H. Gómez-Adorno, J.P. Posadas-Durán, G. Sidorov, and D. Pinto. 25 Jan 2018. Document embeddings learned on various types of n-grams for cross-topic authorship attribution. *Computing* 100: 741-756.

[3] J. Schler, M. Koppel, S. Argamon and J. Pennebaker. 2006. Effects of age and gender on blogging. In *Proc. 2006 AAAI Spring Symposium on Computational Approaches for Analyzing Weblogs*.

[4] http://u.cs.biu.ac.il/~koppel/BlogCorpus.htm

[5] https://radimrehurek.com/gensim/models/doc2vec.html

[6] https://spacy.io/