Authorship identification with RNNs

Nick Fireman

1 Introduction

This project tackles the *authorship identification* problem, which is defined by determining the author of a given passage of text. This is a common application of the field of *stylometry*, which describes the more general attempt to quantify linguistic style in various ways. Authorship identification has clear applications in plagiarism detection, and has also seen use in the fields of law enforcement and intelligence to identify sources of harmful messages.

Early attempts at authorship identification used statistical methods such as principal component analysis to classify texts using human-selected features such as the punctuation frequency or average lengths of words or sentences [1]. While this has been shown to be effective in certain limited settings, the techniques of deep learning have been found to improve the state-of-the-art performance in more general, realistic, and thus useful contexts. This project shows one such context, where the particularly well-suited recurrent neural network (RNN) architecture is used to classify the authors of public domain literary works.

2 Approach

In this project's setting, the possible set of authors $\mathcal{A}$ is predetermined and fixed for each experiment, so the authorship identification problem can be viewed as a supervised classification problem, where the network is tasked with mapping an input passage to an author in $\mathcal{A}$. As with any other classification problem, the network is trained from a large selection of author-labeled passages for each author in $\mathcal{A}$.

The GloVe model [2] is used to provide a meaningful embedding $G : \mathcal{W} \to \mathbb{R}^n$ of tokens or words $w \in \mathcal{W}$ to n-dimensional real-valued vectors. The set $\mathcal{W}$ is called the *vocabulary* of the embedding G. The vector representations provided by GloVe are designed to respect *analogies*, in the sense that an analogies between two pairs of words $w_1 : w_2 :: w_3 : w_4$ (e.g., son : daughter :: father : mother) can be recovered from the words' vector representations. In particular, $|(G(w_2) - G(w_1)) - (G(w_4) - G(w_3))|$ will be small. This can be used to "solve" the analogy for the word w_4 (or any other word) by finding the word(s) $w \in \mathcal{W}$ minimizing $|G(w_2) - G(w_1) + G(w_3) - G(w)|$. A GloVe embedding is trained via an unsupervised regime, and pretrained embeddings based on various public datasets are used in this project, discussed further in Section 3.

The process of mapping a substring of an author's work to a list of tokens in $\mathcal{W}$ is called *tokenization*, which required some care, since the pretrained GloVe embeddings used in this project have a vocabulary which includes some nonword tokens. In particular, punctuation such as periods, semicolons, and parentheses each have a vector representation in all of the pretrained embeddings, and were split into tokens accordingly. Furthermore, words with apostrophes are also split into two tokens in $\mathcal{W}$. For example, $\{\texttt{n't}, \texttt{'re}, \texttt{'ll}\} \subset \mathcal{W}$. The tokenizer in the spaCy library [3] was used, since it had these tokenization rules implemented by default.

The RNN architecture is used for this project, since it's particularly well-suited for variable-length sequential data such as a list of tokens. To classify a passage with the RNN, the tokens of the passage are input to the RNN one by one, and the final output from the RNN is used to determine the network's classification. Note that the RNN produces an output after each token, but since only the entire passage is being classified, all outputs besides the last one are ignored.

3 Datasets

Project Gutenberg. Project Gutenberg [4] is an organization which curates and hosts a collection of literary works in the public domain. These works are made available in many encodings, including UTF-8, which is what this project used. A preprocessed collection of over 3,000 UTF-8 English works written by 142 authors was made publically available by [5]. The preprocessing specifically involved manual removal of external metadata such as license information and transcribers' notes (i.e. content not written by the original work's author) which is irregularly added to the work's full text by Project Gutenberg. Note that work's metadata written by the original author, such as its table of contents and chapter headings, are left intact in this dataset.

Pretrained GloVe embeddings. Alongside introducing the GloVe model, [2] also provided datasets of pretrained word vectors, five of which are used in this project. Four pretrained models (400K vocabulary; vectors of dimension 50, 100, 200, and 300, respectively) were built from 6B tokens drawn from Wikipedia and Gigaword 5 (a collection of newswire data). The fifth pretrained model (1.9M vocabulary; vectors of dimension 300) was built from 42B tokens drawn from Common Crawl [6].

4 Experiments

An experiment in this project consists of the following steps:

- Training and testing datasets are sampled from the corpus of literature. Each dataset consists of a large number of passages, where each passage contains a fixed number of tokens. The same number of passages are sampled for each author to be classified.

- A pretrained GloVe embedding is used to map each passage to a numeric tensor.
- A number of training iterations are performed, where each iteration trains the RNN model with one hidden layer for 10 epochs using the training dataset, then computes the accuracy of the RNN on the testing dataset. Note that the peak testing accuracy may occur before the last training iteration.
- The training loss and the testing accuracy are plotted against the number of epochs trained. This produces a graph that shows how the model improves over the training period.

In this project, the authors Charles Dickens and Mark Twain were classified. Training and testing datasets of 40,000 and 10,000 passages per author were used, respectively, where each passage consists of 30 tokens. An initial round of $18 = 2 \times 3 \times 3$ experiments were held to find the best configurations of the following three hyperparameters:

- The choice of the pretrained GloVe embedding; the `6B.50d` and `42B.300d` embeddings were considered.[1]
- The size of the RNN's hidden state (referred to as the *hidden size* of the RNN); the sizes 25, 50, and 100 were considered.
- The learning rate of the RNN; the rates 0.0005, 0.001, and 0.005 were considered.

The results of this first round of experiments are given in Table 1. The nine experiments using the `6B.50d` GloVe model have an average accuracy of 0.7699, while the same experiments with the `42B.300d` GloVe model yielded 0.7893. This shows that a better-trained word embedding model does have a small but noticeable impact on the strength of the resulting RNN. However, comparing the two GloVe models solely by the number of training epochs may be misleading, since the same number of training epochs take roughly four times longer for the larger GloVe model.[2]

The highest final accuracy of 0.8067 was attained with the `42B.300d` model, alongside a hidden size of 100 and a learning rate of 0.001. The highest observed peak accuracy of 0.8194, after 90 epochs of training, was attained from the same learning rate and a hidden size of 50.

The experiments using the `42B.300d` model fared noticeably worse after the full 100 epochs of training than the `6B.50d` model. Only one of the experiments using the 300-dimensional GloVe model had a peak accuracy occur after 100 epochs, while only three of the 50-dimensional model experiments peaked in test accuracy before 100 epochs (and those that did peaked at 90, only one iteration before the end of training).

[1] The meaning of e.g. `6B.50d` is that the model was trained on 6 billion tokens and consists of vectors of dimension 50.

[2] All experiments were performed locally on the same home machine while it was not otherwise in use.

GLoVE	Hidden Size	LR	Duration (s)	Peak Test Accuracy	Final Acc.
6B.50d	25	0.0005	1388	0.7597 (90 epochs)	0.7586
		0.001	1415	0.7854 (100 epochs)	0.7854
		0.005	1384	0.7972 (100 epochs)	0.7972
	50	0.0005	1404	0.7593 (90 epochs)	0.6640
		0.001	1433	0.7809 (100 epochs)	0.7809
		0.005	1442	0.7956 (100 epochs)	0.7956
	100	0.0005	1220	0.7631 (100 epochs)	0.7631
		0.001	1185	0.7897 (100 epochs)	0.7897
		0.005	1182	0.7947 (90 epochs)	0.7947
42B.300d	25	0.0005	4710	0.7966 (90 epochs)	0.7813
		0.001	5191	0.8036 (80 epochs)	0.7766
		0.005	5320	0.8075 (60 epochs)	0.7840
	50	0.0005	5439	0.7990 (100 epochs)	0.7990
		0.001	5332	**0.8194 (90 epochs)**	0.7929
		0.005	4784	0.7967 (60 epochs)	0.7810
	100	0.0005	4780	0.7959 (90 epochs)	0.7876
		0.001	4740	0.8083 (90 epochs)	**0.8067**
		0.005	4701	0.8129 (70 epochs)	0.7949

Table 1: First round of experimental results.

GLoVe	Hidden Size	LR	Duration (s)	Peak Test Accuracy	Final Acc.
42B.300d	50	0.001	15824	0.8289 (170 epochs)	0.8188
	100		15374	0.8363 (180 epochs)	0.8167

Table 2: Second round of experimental results.

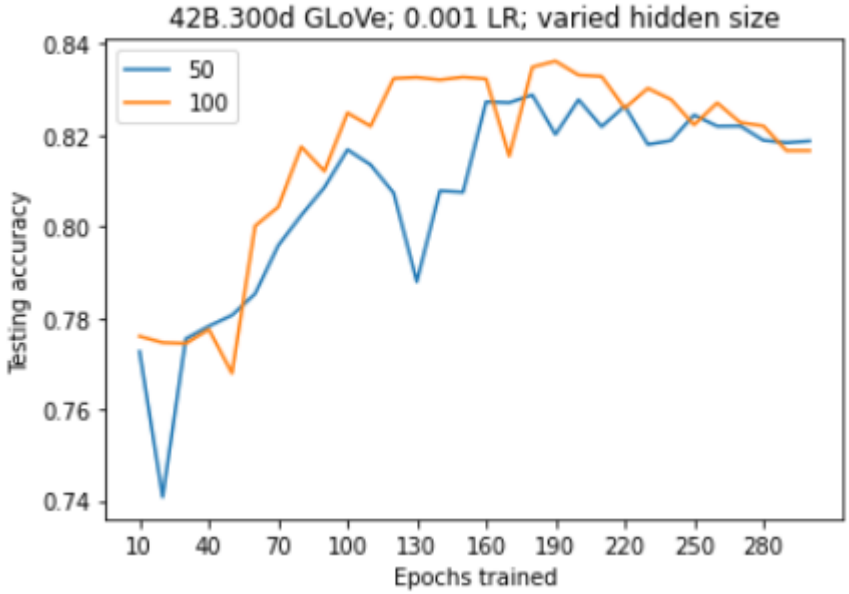

Figure 1: Testing accuracy in the second round of experiments.

Because of this, a second round of longer experiments was performed using the `42B.300d` GloVe model, a learning rate of 0.001, and hidden sizes of 50 and 100. These two models were trained three times longer than the first round (for 300 epochs in total) to assess whether the larger GloVe model would do better with more training.

The results of these experiments is given in Table 2, and the test accuracy after each iteration of 10 training epochs is plotted in Figure 1. The dropoff in test accuracy over the last 100 epochs suggests that both models begin to suffer from overfitting at this point. Even so, we find that both models gain some accuracy after the longer training regimen, which suggests that given more time and a larger training dataset, this approach could still yield better results.

5 Conclusion

The above experiments show that the RNN architecture is suitable for authorship identification, showing classification accuracies around 75 to 80% when distinguishing two authors,

which significantly outperforms the expected accuracy of 50% when randomly guessing. After an initial round of hyperparameter tuning, two promising models were trained for several hours and were shown to achieve nearly 83% accuracy, though both showed signs of overfitting to the training data.

The Jupyter nodebook containing the code used to perform all experiments is available on Github[3]. Because the pretrained GloVe word embedding and the collection of books are too large to host on Github, only a small subset of both are hosted in the Github repository. The full pretrained GloVe embeddings are hosted by [7] and the full Project Gutenberg dataset is hosted by [8].

References

[1] JNG Binongo and MWA Smith. "The application of principal component analysis to stylometry". In: *Literary and Linguistic Computing* 14.4 (Dec. 1999), pp. 445–466. ISSN: 0268-1145. DOI: 10.1093/llc/14.4.445.

[2] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543.

[3] *spaCy: Industrial-Strength Natural Language Processing in Python*. URL: www.spacy.io.

[4] *Project Gutenberg*. URL: www.gutenberg.org.

[5] Shibamouli Lahiri. "Complexity of Word Collocation Networks: A Preliminary Structural Analysis". In: Gothenburg, Sweden, Apr. 2014, pp. 96–105.

[6] *Common Crawl*. URL: www.commoncrawl.org.

[7] *GloVe: Global Vectors for Word Representation*. URL: https://nlp.stanford.edu/projects/glove/.

[8] *GloVe: Global Vectors for Word Representation*. URL: https://web.eecs.umich.edu/~lahiri/gutenberg_dataset.html.

[9] Shriya TP Gupta, Jajati Keshari Sahoo, and Rajendra Kumar Roul. "Authorship Identification Using Recurrent Neural Networks". In: *Proceedings of the 2019 3rd International Conference on Information System and Data Mining*. ICISDM 2019. Houston, TX, USA, 2019, pp. 133–137.

[3] https://github.com/graftss/authorship-identification/blob/master/authorship-project.ipynb