Quote Attribution for Literary Text with Neural Networks

Arun Chaganty
Department of Computer Science
Stanford University
chaganty@cs.stanford.edu

Grace Muzny
Department of Computer Science
Stanford University
muzny@cs.stanford.edu

Abstract

We propose a method for using neural networks to attribute quotes in literary texts. Since previous work has been unable to successfully solve this problem based on bag-of-words features, we study the issue of whether this is due to the limited expressiveness of such features. By re-framing the modeling of quotes and characters as based off of word vectors, we hope to demonstrate that individual characters do, in fact, have recognizable vocabulary-based characteristics. The limited performance of our resulting models demonstrate that there are more complicated forces at play than simply the words in the quote when determining its speaker.

1 Introduction

Quote attribution is the problem of, given a quotation and the text in which it appears, identify the speaker. Such systems are useful because they can give insight into questions such as: "who started that rumor?", and "how do quotes contribute to a news story?". In addition, they can be used to perform corpus analysis of documents, charting the changing styles of dialogue and speech as more and more historical documents and texts are digitized.

We concentrate on this problem within the literary domain, that is, given a quote, identify the character who made the utterance. A full, end-to-end, quote attribution system of this variety requires many pieces that can be summarized as: 1) quote identification 2) character identification and 3) quote attribution.

As O'Keefe et al. [3] note, it is fairly simple to build a deterministic quote extractor that is very accurate, given the following stipulations: 1) the data is clean and 2) the quotes you are interested in are delimited via standard quotation-marks. For the purposes of this project, we use a simple deterministic quote extractor to identify quotes in a text.

The problem of character identification, though interesting, is encompassed within this project as it is a rather different problem of proper noun coreference. We get around this issue by using hand-crafted character lists which contain all speaking characters in the text. And are thusly, able to concentrate solely the third problem of quote attribution itself.

2 Background

The first major attempt at quote attribution was by Elson and McKeown, [1], work in which the task is formulated as a kind of *mention* identification. In this approach, given a quote q, the task is to determine which candidate mention c_i of a list of candidate mentions C refers to the true speaker of the quote. Elson and McKeown [1] used a supervised learning technique in which they used a set of classifiers, each trained as an expert for identifying the speaker of a specific type of quote (e.g.

Quote-Said-Person or Anaphora trigram) to achieve an overall performance of 83%. With this work, they released the Columbia Quoted Speech Attribution Corpus (CQSA), a corpus of selections from 11 novels and stories annotated for quote attribution.

O'Keefe et al. [3] followed this work up by recognizing the task as one of sequence labelling. Using the same formulation of the problem as Elson and McKeown [1], O'Keefe et al. [3] point out that Elson and McKeown worked off of the unrealistic assumption that all previous quotes were attributed correctly and show that once this expectation is removed, performance drops dramatically. Their sequence modelling approach achieves state of the art results on the WSJ data and on a corpus they annotated from Sydney Morning Herald text, but does worse than the baseline (which gets 53.3%) on CQSA.

Since this work showed the problem of quote attribution in literary text, which tends to be dialogue-heavy, to be unsolved, He et al. [2] have worked on solving it specifically for the literary domain. In this work, He et al. [2] reformulate the problem as *speaker* identification–an assignment of a target quote q to it's speaker s rather than the span of text that mentions s. Using a supervised learning method to learn how to rank speakers so that their method can be applied to novels different from the one trained one (with a drop in performance of less than 10%, but do not further report the drop in accuracy from going cross-novel). They achieve state-of-the-art performance ranging from 82.5% (on *Pride and Prejudice*) to 74.8% (on *Emma*, from the Elson and McKeown's CQSA [1]).

3 Approach

3.1 Models

We use to learn the appropriate transformation between quote vectors q and character vectors c. For every quote in our training data, we construct a quote vector q_i that is associated with a character vector c_j. Quote vectors are created by averaging the word vectors of each component word in the target quote together, while character vectors are created by averaging the word vectors of each unambiguous referent to that character together (e.g. the character vector for "Lydia Bennet" will be the average of "Miss", "Lydia", and "Bennet").

We initialize our word vectors with pre-trained GloVe vectors [4] and update these vectors according to the raw text of the novel in question.

3.1.1 n-way Classification

Our first model is a single-hidden layer neural network with a softmax output layer that takes as input quote vectors q and predicts the appropriate character label, drawing from all possible characters. This model suffers from the need to have a pre-determined character list and is thus less-easily adapted to new texts.

3.1.2 Binary Classification

Our second model is also a single-hidden layer neural network with a softmax output layer, but instead of predicting one of many different labels, it predicts the binary decision of "speaker" or "not-speaker". It follows from this design that the model takes as a input both the quote vector q and a character vector c concatenated together. The final character label for a quote is determined by choosing the character that resulted in the highest "speaker" score. If, for a quote, no characters are scored as "speaker", the quote is left un-attributed.

3.2 Context

We experiment with including context windows in our quote vectors, considering the tokens contained by the preceding and following characters surrounding the quote to be a part of it when constructing our quote vectors. These experiments follow the intuition that the text surrounding a quote often gives a high level of signal in indicating which character is the speaker (e.g. "'Take that duck away!' shouted Darcy" or "... to which Lady Catherine haughtily replied 'I don't like crumpets"').

3.3 Data

We use the data published by He et al. [2], which contains the text of Jane Austen's *Pride & Prejudice*, a quote list of quotes, and a list of characters and all their unambiguous referents. In addition, we test the cross-novel compatibility of our models on the *Emma* portion of the Columbia Quoted Speech Attribution Corpus (CQSA) [1], which contains a variety of 19th century novels and short stories.

It is worth noting that these corpora differ slightly in their annotation schema. In the *P & P* corpus, quotes are not linked to specific character mentions, they are simply annotated with a unique character identifier. In CQSA, quotes are linked to specific mentions of the speaker and due to the difficulty of coreference, many of the quotes cannot unambiguously be resolved to a single character, and so yields a lower number of unambiguously-identified quotes per novel.

We split our train, dev, and test sets as in He et al. [2].

Dataset		total	train	dev	test
Pride & Prejudice	Quotes	1724	1387	162	175
	Unique speakers	25	24	9	10
Emma	Quotes	492	328	82	82
	Unique speakers	11	6	8	7

Table 1: Dataset descriptions

4 Experiments

We evaluate our system based off of accuracy, precision, recall, and F1-measure. This approach makes sense because not all of our models will be attributing every quote to a speaker, and because previous work has evaluated off of accuracy, making our results more easily comparable.

Our baseline is a majority classifier. In the case of *Pride & Prejudice*, this classifier attributes every quote to Elizabeth Bennet, while in *Emma*, it attributes every quote to Emma Woodhouse. The baseline classifiers' relatively good performance show that one of the issues that a quote attribution system will have to overcome is the unbalanced nature of the data.

Dataset	Model	Context Size	Accuracy	Precision	Recall	F1
Pride & Prejudice	n-way	0	8.64	5.54	2.65	2.91
		3	10.49	6.43	3.51	4.07
		10	13.58	6.27	3.34	4.12
		15	13.58	5.73	2.85	3.63
	binary	0	42.59	4.73	11.1	6.63
		3	42.59	4.73	11.1	6.63
		10	42.59	4.73	11.1	6.63
		15	42.59	4.73	11.1	6.63
Emma	n-way	0	32.92	22.90	18.77	15.86
		3	28.80	13.82	15.35	11.54
		10	28.04	17.72	0.11.75	13.68
		15	26.82	13.71	15.06	11.89
	binary	0	42.68	5.33	12.5	7.47
		3	42.68	5.33	12.5	7.47
		10	42.68	5.33	12.5	7.47
		15	42.68	5.33	12.5	7.47

Table 2: Context experiment results

As shown in table 3, we perform cross-novel compatibility experiments to test the generalizability of our approach. In these experiments, we train our binary classifier on one dataset and test on the other. From the results of these experiments, it becomes obvious that our models suffer from a general lack of data, an issues discussed in greater depth in sections 4.1 and 6.

Dataset (train)	Dataset (test)	Accuracy	Precision	Recall	F1
Pride & Prejudice	*Pride & Prejudice*	30.20	1.25	4.16	1.93
Pride & Prejudice	*Emma*	20.73	3.45	16.6	5.72
Emma	*Pride & Prejudice*	3.74	0.15	4.16	0.30
Emma	*Emma*	41.76	6.96	16.66	9.8

Table 3: Cross-domain testing results

Dataset	Method	Accuracy	Precision	Recall	F1
Pride & Prejudice	Majority classifier	42.59 (dev)	4.73	11.1	6.63
	n-way	13.58	6.27	3.34	4.12
	binary	42.59	4.73	11.1	6.63
	He et al. [2][1]	82.5	-	-	-
Emma	Majority classifier	42.68 (dev)	5.33	12.50	7.47
	n-way	32.92	22.90	18.77	15.86
	binary	42.68	5.33	12.50	7.47
	He et al. [2]	74.8	-	-	-

Table 4: Overall quote attribution results

Unfortunately, in the end the performance of our models was limited at best. We discuss the underlying reasons and possible future directions more in sections 4.1 and 6.

4.1 Error Analysis

Error analysis shows that the poor performance of our models is likely due to a variety of factors, some of which we enumerate below:

- Some quotes, such as, "but it is ," contain so little content along every dimension that it is unsurprising that our system seemed to randomly choose a speaker. This indicates that a successful system will need to incorporate some notion of the global conversation that is at play.
- The quote vectors are not predictive of the character. Since a character is able to speak a wide variety of quotes with large variance in their syntactic and semantic content, both the ourselves and the neural networks were unable to find a pattern in the transformation between quote vectors and characters.
- The models appear to function as random classifiers. Analysis of the output shows that the models appear to function as random classifiers, unable to consistently outperform the majority classifier due to this randomness.
- While deterministic quote classifiers based off of the context of a quote work very well for a small portion of quotes, this context is un-needed and perhaps harmful in other cases. We hypothesize that the general lack of performance gains when adding context to the quotes is in part due to the model not being able to learn the difference between context with signal and context without signal. The constant performance of the binary classifier leads us to believe that the opposing upward and downward trends for our different datasets are not indications that we can trust this particular method of integrating context.
- Our models suffer from lack of training data. This becomes especially clear when looking at the cross-novel compatibility experiments when training on *Emma* (328 quotes) and testing on *P & P*. Here, we see a dramatic drop in performance of more than 25% accuracy.

4.2 Vector analysis

We conducted additional error analysis by inspecting the quote vectors that were produced with PCA, shown in figure 1. These figures support our conclusion that our models are unable to find

[1]The data sets used by He et al. are slighlty smaller than those we use, containing 1260 and 397 quotes for *P & P* and *Emma*, respectively. It is our hypothesis that this is due to an unknown preprocessing step.

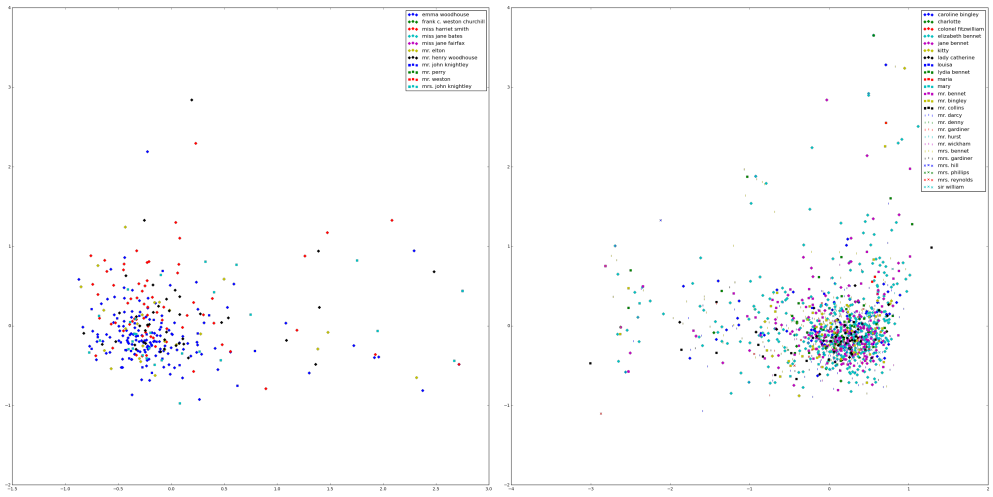

Figure 1: *Emma* (left) and *P & P* (right) quote vectors using PCA

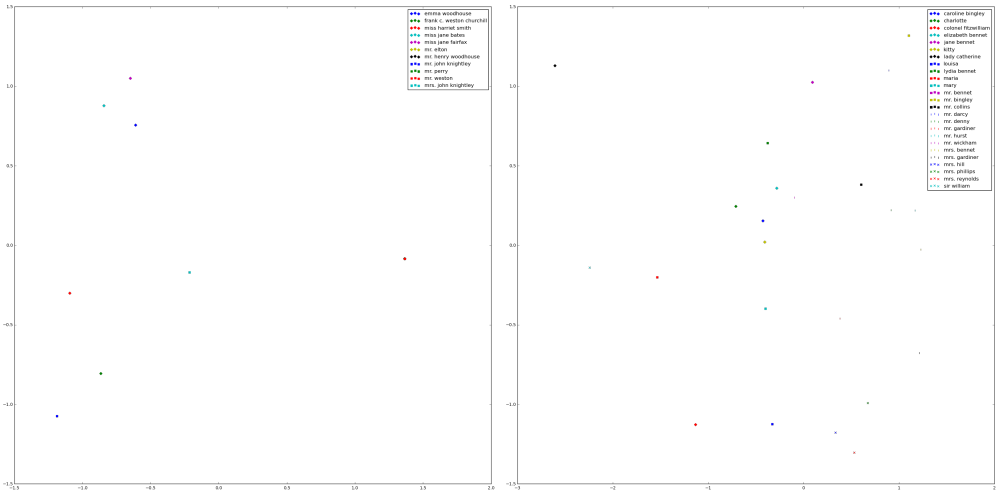

Figure 2: *Emma* (left) and *P & P* (right) character vectors using PCA

structure in distinguishing the quote vectors of one character versus another. Analysis of the character vectors, shown in figure 2, yield slightly better results–male and female characters tend to be closer together, but our experiments show that this was not enough to overcome the other shortcomings of the models.

5 Conclusion

While we were unable to perform as well as the state of the art systems, or even better than our majority-class baselines, which have no network-based components, we gained valuable insights into the performance of such models and the nature of dialogue in literary text. We found that, just as a human would have an incredibly difficult time identifying the speaker of a quote from it's text alone, so do our models.

Our results show that deep models based only off of bag-of-words intuitions will likely not be able to solve, or even perform well on this task. This indicates that for the most part, characters in novels tend to have similar language-usage patterns at the word level. This rejects the hypothesis that previous work was unable to do well based of of bag-of-words features because the models were not powerful enough.

6 Future Work

If one were to continue working on the problem of quote attribution with neural networks, we believe that the models developed would benefit from added structure. We believe that a sensible place to start follow-up experiments would be by modeling the problem as one of sequence-labelling, testing whether RNNs are able to outperform those models used by O'Keefe et al. [3]. These models should be better able to determine when the context surrounding a quote matters to determining its' speaker, leading us to believe that such a system would also benefit from using units such as LSTMs.

Another intuition that we gained while conducting error analysis is that we simply didn't have enough data. We believe that a neural network-based quote attribution system will require more training data (in the range of ten thousand examples rather than one thousand). While one could extract a thousand or more training examples from CQSA, we believe that a successful model will need training data with a higher degree of conversational coverage, that is, for each conversation, all quotes should be labelled, which is not the case for CQSA, as described in section 3.3

References

[1] David K Elson and Kathleen McKeown. Automatic attribution of quoted speech in literary narrative. In *AAAI*, 2010.

[2] Hua He, Denilson Barbosa, and Grzegorz Kondrak. Identification of speakers in novels. In *ACL (1)*, pages 1312–1320, 2013.

[3] Tim O'Keefe, Silvia Pareti, James R Curran, Irena Koprinska, and Matthew Honnibal. A sequence labelling approach to quote attribution. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 790–799. Association for Computational Linguistics, 2012.

[4] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 12, 2014.