CS 447 intermediate project report

Manasa Hariharan (manasah2)

November 26, 2018

Finding a Character's Voice: Stylome Classification on Literary Characters

1 Goal of the Project

The goal of the project is to implement the paper cited [3] and to distinguish and classify styles of different characters and to verify if an author has managed to create believable characters with individual styles. This is done by taking an epistolary novel called "Liaisons Dangereuses" written in the 18th Century with 175 chapters written in the form of letters between characters (with 7 main characters). A classifier will be created that will have text features from a chapter as its input and will identify which character has "wrote" the letter. (bear in mind they are written by the same person, the author, but the letters are from the points of view of different characters). The accuracy of the classifier will be calculated per character and averaged. An analysis of which features in the classifier help the most with the identification of styles and characters will also be included. The text features that are given to the classifier will themselves come from a bag-of-words model. This project will also experiment with different variations of the bag-of-words model and compare the results obtained.

2 Background

The broader goal of this project fits into the field of Stylometry and is very related to authorship attribution. Stylometry is the study of linguistic style in written language. It has legal as well as academic and literary applications ranging from authorship attribution to forensic linguistics. Authorship attribution has been a long studied field and is based on the assumption that there are stylistic features that can help distinguish the real author of a text from any other theoretical author. Past studies have identified several features that help in authorship attribution like function word frequencies[6], grammatical structures[1], part-of-speech n-grams[4], lexical richness[7], and character n-grams[5]. Applications like author verification, author profiling, plagiarism detection all come under this topic.

A related but less analyzed field is that of distinguishing between the writing styles of fictional people, namely literary characters. This problem is interesting to study whether an author managed to create characters that are believable as separate people with individual styles. This is also relevant as scholars sometimes cite the diversity of characters' speech styles as grounds for doubt about the efficacy of authorship attribution.

This claim was also analyzed in [2] where the characters of 5 17th century playwrights were compared and analyzed. In this paper, the authors had considered the characters and writing styles of 5 famous 17th century playwrights: Shakespeare, Fletcher, Middleton, Jonson and Chapson. Major characters across all the author's plays were considered and word vectors for each of these characters were then taken as a data set. One- on- One comparisons were then made between pairs of authors and their characters using Principle Component Analysis. A cluster separation line was arrived at using Support Vector Machines for each of the comparisons.

The purpose of this was to analyze whether different characters written by the same author were more different in linguistic style when compared to different authors' styles. These PCA type plots can help us visualize both aspects of these differences. The results of this paper showed that while the characters written by the same writer showed an impressive range on the PCA component space thus proving their ability to create distinctive and believable characters, there was also almost always a distinct separation between characters written by different authors. Hence this paper concluded that no matter how different 2 characters written by the same author are, they wouldn't affect or bias the problem of author attribution.

I chose this paper to talk about more in detail apart from the one being implemented in this project because, this paper used methodology that is related to the one being implemented in this project to answer a very valid question in the field of stylometry research. Other analysis in this field includes the analysis of the works of 2 authors who are known to work together to distinguish character and author styles[8]. This project takes ideas from all the cited works to solve the problem of fictional character classification.

3 Task

The task for this project is mainly taken from the title paper. I take the 18th century French epistolary novel "Liaisons Dangereuses" by Pierre Choderlos de Laclos. The plot of the book is built around two main characters, the Vicomte de Valmont, and the Marquise de Merteuil, who engage with various other characters especially as part of games of seduction, deceit or revenge. The other characters act as their victims, in various roles: Cécile, the innocent young girl who Merteuil plans to corrupt, Danceny, her young passionate admirer, Madame de Tourvel, a faithful wife who Valmont intends to seduce[3]

This novel and format were chosen because each of the chapters is in the form of a letter between one character to another making it ideal for labeling our text samples with the character that the text is attributed to. Also, the original french text is used so as to prevent any noise that may be introduced due to translation.

The book contains 175 chapter with an average of 800 words per chapter. The task is to build a model that classifies these characters using a language model that takes in text features from these chapters as it's input. I have followed the methodology as given in the paper.

To formally define the task: To see if fictional characters can be classified by their text features using a natural language model. We use the test as a set of labeled data samples with the labels being the sender's name. We exclude one sample at a time and create a balance data set to train the classifier on, using the excluded text sample as the test set. Average accuracy rates are calculated as a measure of evaluation. Also, different types of text features are used and compared to understand what kind of features are most helpful for this type of task. Finally a complete analysis of the results is presented and the most discriminating features for each of the characters are determined. Each step of this process will be explained in the following sections. First, below the pre-processing stage is explained:

3.1 Cleaning and processing text

The text itself is available in public domain in it's original version. Cleaning it involved

- Locating the beginning and ends of chapters and then saving the chapters as a list of words.
- The 1st line of every chapter was removed as that usually contained the name of the sender and recipient
- the text had annotations inside to help the reader understand certain terms. These annotations were located and removed as well and the text converted to lowercase
- Finally, only characters that had written at least 3 letters were considered main characters. Therefore, any chapters written by characters that had ¡ 3 letters were removed

The final data set had 168 chapters with a minimum of 9 chapters per character

4 Model

The model for this project is a linear support vector classifier. The features that serve as the input for the linear SVC come from a bag-of-words model. This bag-of-words model takes the chapters text as it's input and outputs a matrix of token counts (or tfidf vectors). These features are then sent to the SVC to train the model. Finally, the test set is also passed through the bag-of-words model (and in some cases, the tfidf vectorizer) and these features are passed to the trained model to get the predicted label.

The variations of n-grams done are as follows (unless mentioned, using tf-idf as features):

- simple n-grams with n = 1, using counts as features
- n-grams with content words only
- k-best features using chi-square feature selection
- character 3-grams, using counts as features

- character 5-grams, using counts as features
- bigrams using all words

k is taken as 1000 in the paper. A pipeline is built for every model using all steps.

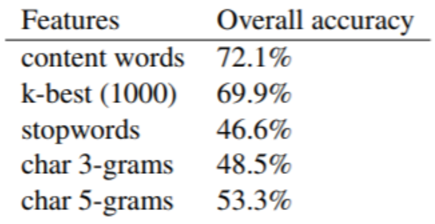
4.1 Methodology

- The cleaned data is saved as a .csv file with 2 columns: label of character, and a list of all words in the chapter in order (but without punctuation or numbers)
- the models.py script is called with one argument indicating which variation of bag-of-words we want to see.
- depending on which model is called,we build a pipeline that contains all stages of that classifier.
- we iterate through each chapter in the data set and for each chapter,
  - a function is called (passing the data and the current chapter and label as arguments) that builds a temporary data set including all chapters except the current one.
  - we then count the minimum number of chapters that any character has in this temporary data set (usually 9, sometimes 8)
  - The paper says to artificially balance the data set such that each character only has as many chapters as the minimum number of chapters for any character. However it did not mention how it had done the balancing.
  - I tried to balance it by randomly sampling the required number of chapters from all chapters written by that character. However this leads to a variation in the results depending on how this sampling is done.
  - Finally, the indices of the sampled chapters are returned from the function and created as a dataset
  - The returned chapters are now passed through the pipeline where their frequency counts are calculated (and possibly tf-idf scores) and sent to the SVC classifier. The frequency counts are calculated using the CountVectorizer function in the sklearn feature extraction module and the tf-idf scoresare calculated using the TfidfTransformer function in the same module. The SVC is constructed and trained using the linearSVC function in the sklearn svm module (linearSGD classifier from the linear model module and the generic SVC function were some of the other options tried)
  - Once the classifier is trained with the text features, we predict the label of the chapter that we have held out on. This is similar to the out-of-bag approach and is proven to be an effective estimate.

- as we do this above process for every chapter, we keep a count of correct predictions and store the values in a dictionary that contains accuracy per character
- Finally, the character-specific accuracy and the average accuracy is displayed and stored

5 Experiments

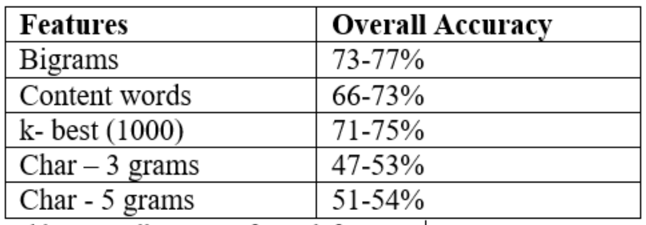
In this section, I will compare the results in the paper along with the results I obtained. The accuracy as given in the paper is given below in the figure [3]



Features	Overall accuracy
content words	72.1%
k-best (1000)	69.9%
stopwords	46.6%
char 3-grams	48.5%
char 5-grams	53.3%

Table 4: Overall accuracy for each featureset

Since stopwords was the lowest accuracy model, I tried a different variation instead. I looked into bigrams of all words and realized that it actually gives better accuracy than my content words model. As we can also see, my content words model shows lesser accuracy than the one in the paper. I believe that this could be because my definition of content words might be different from theirs (I removed all stopwords using the french stopwords list in the nltk package). All my results are summarized below:



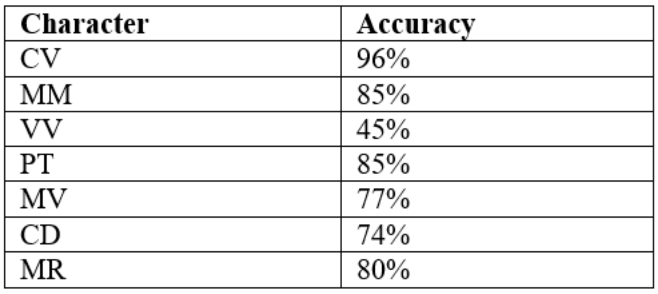
Features	Overall Accuracy
Bigrams	73-77%
Content words	66-73%
k- best (1000)	71-75%
Char – 3 grams	47-53%
Char - 5 grams	51-54%

Table : Overall accuracy for each featureset

Some points to note:

- The accuracy varied due to the random sampling of chapters when balancing the dataset. I tried to take the 1st k or last k chapters as a way to prevent the fluctuation but I believe that introduces an inherent bias
- I also tried to sample the chapters using a different code logic (using pandas and sklearn), however that didn't change the results much.
- For deciding the best hyperparameters, Since tuning it for each chapter is an onerous process, I have attached separate code that I used to tune the classifier on a small number of chapters before settling on parameters that were deemed best (In most cases, the default values worked best)

- While tuning the classifier, I have not trained the CountVectorizer or tf-idf parts as their parameters are already fixed, Therefore only the SVC hyperparameters were tuned.
- I also tried n grams with high value, but the test accuracies only decreased and the model was overfitting. Given the size of the data set, I believe any more complexity would only overfit the data



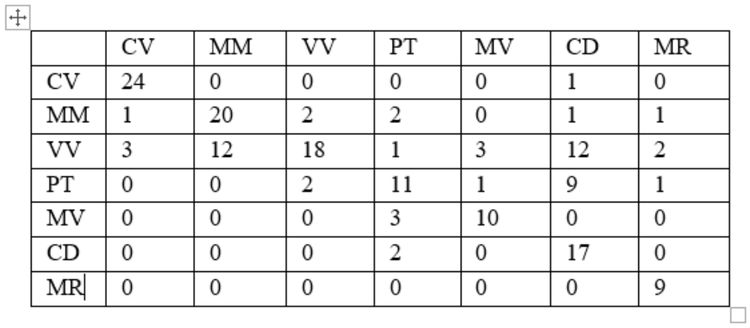
Character	Accuracy
CV	96%
MM	85%
VV	45%
PT	85%
MV	77%
CD	74%
MR	80%

Character-wise accuracy for bigram model

Above is the character-wise accuracy for my bigram model As we can see, the accuracy varies a bit but is similar to the results in the paper. CV for example has very high accuracy because most of her letters are to 1 other character that does not have chapters of their own. Hence it is very distinct and easy to capture. These relationships are further examined in the next section

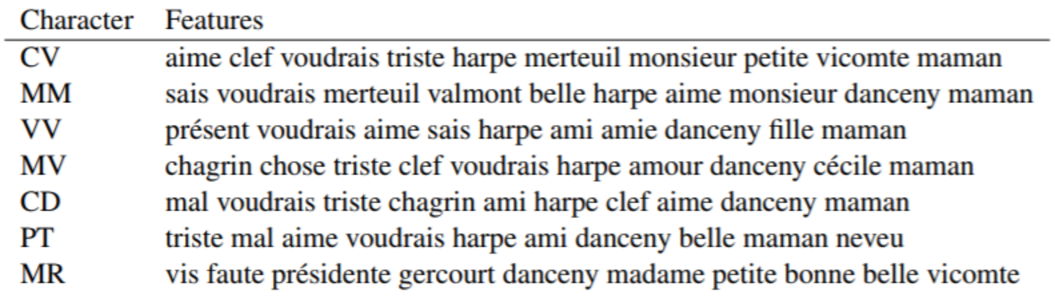
6 Further Analysis

To understand how the misclassifications have tended to happen I have attached a confusion matrix of my final result (all these results are for the same random state in the mode)



	CV	MM	VV	PT	MV	CD	MR
CV	24	0	0	0	0	1	0
MM	1	20	2	2	0	1	1
VV	3	12	18	1	3	12	2
PT	0	0	2	11	1	9	1
MV	0	0	0	3	10	0	0
CD	0	0	0	2	0	17	0
MR	0	0	0	0	0	0	9

Like in the paper, VV is the character hardest to classify. Additionally, he most often gets confused with MM, who is his main interlocutor and ”partner in crime”. This may point to a common style, but possibly also to common topics of conversation. Another interesting result is that the top discriminating features are very similar to the paper. This makes sense as it will be the same regardless of how the model is tuned



Character	Features
CV	aime clef voudrais triste harpe merteuil monsieur petite vicomte maman
MM	sais voudrais merteuil valmont belle harpe aime monsieur danceny maman
VV	présent voudrais aime sais harpe ami amie danceny fille maman
MV	chagrin chose triste clef voudrais harpe amour danceny cécile maman
CD	mal voudrais triste chagrin ami harpe clef aime danceny maman
PT	triste mal aime voudrais harpe ami danceny belle maman neveu
MR	vis faute présidente gercourt danceny madame petite bonne belle vicomte

Since my knowledge of french is limited to none, I couldn't make much sense of the words. However, after a little bit of research, while some common words like 'maman' are occurring in all characters, there are also some unique words like 'triste' and 'chagrin' that occur as well. Also, words that indicate other characters also rank high,thus meaning that the recipient or subject or conversation help in identifying the sender.

7 Conclusion

In conclusion, a relatively simple language model can be used to evaluate the linguistic style of a work and classify it's characters provided we use the right features for that work. It also shows that these models are very specific to the data and need to be tuned different for a different work. This analysis also provides valuable information on the design of the novel and it's characters and can be very effectively used to analyse an author's style and be used for important applications in the future with more research. Finally I would like to thank you for giving me this opportunity to perform this analysis in a domain that is so interesting to me. The code for this project is linked here https://github.com/manasahariharan/447project

References

[1] H. Baayen, H. Halteren, and F. Tweedie, "Outside the cave of shadows: Using syntactic annotation to enhance authorship attribution," *Literary and Linguistic Computing*, vol. 11, pp. 121–132, 09 1996.

[2] J. Burrows and H. Craig, "Authors and characters," *English Studies*, vol. 93, no. 3, pp. 292–309, 2012. [Online]. Available: https://doi.org/10.1080/0013838X.2012.668786

[3] L. P. Dinu and A. S. Uban, "Finding a character's voice: Stylome classification on literary characters," in *Proceedings of the Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature.* Association for Computational Linguistics, 2017, pp. 78–82. [Online]. Available: http://aclweb.org/anthology/W17-2210

[4] M. Koppel and J. Schler, "Exploiting stylistic idiosyncrasies for authorship attribution," in *In IJCAI'03 Workshop on Computational Approaches to Style Analysis and Synthesis*, 2003, pp. 69–72.

[5] ——, “Authorship verification as a one-class classification problem,” in *Proceedings of the Twenty-first International Conference on Machine Learning*, ser. ICML ’04. New York, NY, USA: ACM, 2004, pp. 62–. [Online]. Available: http://doi.acm.org/10.1145/1015330.1015448

[6] F. Mosteller and D. L. Wallace, “Inference in an authorship problem,” *Journal of the American Statistical Association*, vol. 58, no. 302, pp. 275–309, 1963. [Online]. Available: http://www.jstor.org/stable/2283270

[7] F. J. Tweedie and R. H. Baayen, “How variable may a constant be? measures of lexical richness in perspective.” *Computers and the Humanities*, vol. 32, no. 5, pp. 323–352, 1998. [Online]. Available: http://dblp.uni-trier.de/db/journals/lre/lre32.html#TweedieB98

[8] K. van Dalen-Oskam, “Epistolary voices. the case of elisabeth wolff and agatha deken,” *Literary and Linguistic Computing*, vol. 29, no. 3, pp. 443–451, 2014. [Online]. Available: http://dx.doi.org/10.1093/llc/fqu023