COMP90051 Statistical Machine Learning
Authorship Attribution

Group 40 (BayesianNoobs)

Nithin Mathew(1328669) **Ritwik Giri(1301272)** **Vibhuti Rajpal(1305409)**

1. Introduction

In this report, we aim to provide different approaches that can be used to solve an important real-world problem of Authorship Attribution, a process of determining the writer/s of a document. We are provided with a data set of about 25K papers and are required to predict authors (from a set of 100 prolific authors) for the given test data.
The training data consisted pre-processed information about co-authors, abstract, title, venue, and year, where text was tokenised and mapped to a unique integer. This made the task at hand an interesting one.
Since the text component(s) for our input was encoded it is difficult to understand the sentence structure, analyse the grammar and identify the parts of speech. Every author has his own unique way of presenting the information. Stylometric analysis is quite helpful in identifying the literary style of an author. It gives an insight into the vocabulary of the author, the punctuation(s) used, average word length, average words in a sentence, the number of incomplete sentences [1]. However, as discussed earlier, the data provided was in the form of integers making it difficult to analyse the text. After analysing the data, we also realized that most of the papers were written by multiple authors making it even more challenging to identify the unique writing style.
Our approaches were based on analysing the co-authorship network, the relationship between different factors (venue, year, abstract, title, and co-authors), and feature selection algorithms for abstract and title fields. They are discussed in detail in the next section.

2. Approach

Before applying our approaches, our team conducted an extensive explanatory data analysis(EDA) to gather information about the relationship between co-authors and prolific authors (id < 100). The approach to create a author co-author network graph [2] is discussed below.
It was observed from the data that there were more than one author for many papers. To incorporate this relationship into our model, we created an undirected and unweighted graph for the authors using *networkx*. In this graph, the vertices represented the authors and all authors who have worked together were connected by an edge. The graph helped us to capture the transitive relationships between the authors.
The assumption was that if author 1 has worked with author 2 who has, in turn, worked with author 3, then it can be assumed that author 1 and author 3 could have done a paper together. The number of hops between the authors was decided to be 2, that is, If the distance (number of vertices) between the given co-author and the prolific authors ≤ 2, we consider that the two are more likely to work together. The author-coauthor network was also used as a search space for stopwords matrix and autoencoders.

2.1 Stopwords Matrix

As mentioned earlier, the words were mapped to a unique integer value for the given data. Therefore, we had no idea about the conjunctions, determiners, prepositions etc. used to write the abstract for a given paper. In order to identify these, we assumed that these are usually the words that occur most commonly the entire corpus. Therefore, we identified the most common words used in the corpus created by combining abstracts from the training data. We assumed these to be *stopwords*.

For each author, we identified the stop words used in the abstract (say n in number) and created a (n x n) matrix. Here, we assume this matrix to be symmetric. Each element of the matrix, which is the edge weight between stopword i and s, is calculated as $w_{si} = w_{is} = w_{si} + e^{-|P_i - P_j|}$ where P_i is the posi-

tion for ith stop word and P_j is the position for sth stop word. w_{si} is the previous weight assigned. The value of w_{si} was initialised to 0, for each i. The above step is also done for a test sample and *Kullback-Leibler Divergence* is calculated which helps to identify the similarity between the writing styles of 2 authors allowing us to predict the prolific authors [3].
Using this matrix alone,we only achieved an F1 score of 0.12. The algorithm was generating close range similarity scores for multiple authors for a paper , making it difficult to disambiguate between the authors. Hence, we decided to limit our search space using the network graph created earlier.**This resulted in a significant increase in the F1-score taking it to 0.34.**

2.2 Autoencoders

The results were not satisfactory for the previous approach, hence we researched for ways to extract the latent features associated with the writing styles of each authors from abstracts written by them. To achieve this we referred [4], which utilises autoencoders' capacity to encode useful data patterns.
An autoencoder is a type of neural network that tries to compress the input (encode) and then tries to recreate the same input (decode). It trains by optimising the *reconstruction loss*. i.e, the difference between the input and generated output. Autoencoders were created for each of the prolific authors and trained using the set of abstracts of each author.
For any given test record, cosine similarity was calculated between the doc2vec representation of abstract and the reconstructed output from each autoencoder. This similarity was used to rank authors based on their likelihood.
But since multiple authors were associated with each paper and there could be non-writing authors we ran into issues in disambiguating between multiple authors. Hence we reduced the search space by using the earlier cited author-coauthor network graph and checked for similarity between authors within two hops from each coauthor set.**With this approach we saw a slight improvement with a F1-score of 0.37.** But with this approach we were restricted to predictions within the search space and increasing the search space reduced the effectiveness of the approach. We also faced issues with not enough prolific authors with individual publication which could be used to learn their distinct styles. Hence we moved on from unsupervised modelling to exploring other supervised models.

2.3 Multi Label Classification Approach

Our above unsupervised approaches did not provide us with satisfactory results because of the reasons listed in the previous sections. Next,we moved towards supervised approaches. In this, we considered the correlations amongst the explanatory variables to convert the task into a multi-label classification problem. The co-authors, venue, year, abstract and title were considered as the input to the model.
This approach is implemented in different stages as follows:

2.3.1 PRE-PROCESSING

1. Venue: Since it is a categorical feature,we used one hot encoding to generate a sparse matrix representation to be fit into the model.
2. Year: Like venue, it is also a categorical feature and is one hot encoded to generate a sparse matrix representation input for the model. However, we think it does not have a significant impact on the model as the training dataset goes till 18, while the test dataset only contains year 19.
3. Abstract and Title: TF-IDF (Term Frequency - Inverse Document Frequency) was used to generate the vector representation of the encoded abstract data. TF-IDF and word2vec were tested. But word2vec did not perform well as it creates a vector for each word and the dimensions of the feature space grew significantly. We tried the approach of averaging word vectors to obtain a sentence vector, but it did not provide an improvement over TF-IDF.
4. Coauthors: It is converted using multi hot encoding because we needed to incorporate the dependency of a set of coauthors to the actual prolific authors.

2.3.2 Data Selection

To train our classification models we created a sample of data containing papers with prolific authors and with 10% of non-prolific authors, by random sampling. This will avoid class imbalance. The dataset is then split into train data(80%) and holdout test data(20%).

2.3.3 Modelling

Initially we started with a multi-layer perceptron, but due to the lack of sufficient data, we noticed the models was overfitting on the train data. We tried resolving it by incorporating a dropout later but we did not see a significant improvement in its performance.
Next, we tried solving the task by creating multiple multiclass classifiers. We used LabelPowerSet [5] with compliment Naive Bayes and Linear Support Vector Classifier (SVC). The LabelPowerSet is used to create multi-label dataset to a single multi-class dataset. **This resulted in a significant increase in the F1 score to 0.513.**

To further enhance this approach we, incorporated a MultiLabel Binarizer for the output column to translate our label set (set of prolific authors) into a binary matrix representation. This allowed the model to learn the probability for a set of authors occurring together instead of their individual probabilities. Here the LabelPowerset is used as a problem transformation layer to transform our multilabel classification into a multiclass problem by creating labels for each unique combination of labels. With this we are left with a multiclass classification problem to be solved. While looking for classifiers we explored classifiers like linearSVC and complementNB, since these are widely regarded as some of the best classifiers for text classification. We know that SVMs are designed for binary classification problems but LinearSVC handles multiclass classification by building multiple classifiers for each pairs of class labels using a one-vs-rest strategy which involves building a classifier for each class in the model. Since the linearSVC uses a linear kernel it converges faster and is less prone to overfit. Also, SVMs perform well even with high dimensional data, as is in our case, because an SVM tries to learn a plane that splits the data and measures the complexity of such a plane and not the number of features used. We see this in our case with the linearSVC performing the best among the classifiers. **We tried with an F-score of 0.548 on the public leader board test and 0.729 on our holdout test.**

3. Enhancements

As discussed in the previous section, we were provided with an encoded representation of the data. If the actual text was provided, we could apply parts of speech (POS) tagging to extract the grammatical writing style of each authors and perform stylometric analysis to better identify the author. Pre-trained text models like BERT could also be used to better arrive at a vector representation of the data and used to improve the classifier.
Also, if there were enough unique papers for the authors, we could have trained the separate style metrics for each prolific authors based on the sentence structure, usage of stopwords and grammar analysis.

References

[1] Ilker Bozkurt, O. Baghoglu, and Erkan Uyar. Authorship attribution. volume 1, pages 1 – 5, 12 2007.

[2] M. E. J. Newman. Coauthorship networks and patterns of scientific collaboration. *Proceedings of the National Academy of Sciences*, 101(suppl_1):5200–5205, 2004.

[3] R. Arun, V. Suresh, and C.E. Veni Madhavan. Stopword graphs and authorship attribution in text corpora. In *2009 IEEE International Conference on Semantic Computing*, pages 192–196, 2009.

[4] Anamaria Briciu, Gabriela Czibula, and Mihaiela Lupea. Autoat: A deep autoencoder-based classification model for supervised authorship attribution. *Procedia Computer Science*, 192:397–406, 2021.

[5] Jasmin Bogatinovski, Ljupčo Todorovski, Sašo Džeroski, and Dragi Kocev. Comprehensive comparative study of multi-label classification methods. *Expert Systems with Applications*, 203:117215, 2022.