COMP90051 StatML - Authorship Attribution

Group 24

Haonan Zhong *(867492)* Zhi Hern Tom *(1068268)* Yalin Chen *(1218310)*

Introduction

Authorship attribution is the task of identifying the author or authors of an anonymous or disputed text. Such tasks are widely applied to a broad range of applications, such as historical text analysis, academic plagiarism detection, and forensic analysis. In this project, our task is to come up with a method to verify whether the proposed candidate is a true co-author of a given paper based on samples of 2302 authors, verification is done by producing a verification score between 0 and 1 that can be viewed as the confidence estimation.
Tradition authorship attribution tasks generally use stylometry, which analyzes an author's writing style over the entire corpus of their published work. However, our dataset only provides a set of keywords extract from the titles and abstracts of the publications. Several methods has been adopted throughout the project, including binary classification, multi-class classification and multi-label classification. A brief analysis is provided in this report.

Dataset Exploratory

Both training and testing sets are in JSON format. The training dataset provides information on 26,108 papers published between 2000 and 2019 by 2,302 authors, including the key of the paper denoting paper ID, the year of publication, venue of publication, a list of keywords encoded in numbers, and a set of IDs of the co-authors. While the test set contains an extra attribute, the proposed target author's ID. An example of a training and testing instances are listed below.

Training: {'0': 'venue': '5', 'keywords': [64, 1, 322, 134, 136, 127], 'year': 2017, 'author': [1605, 759]}
Testing: {'3': 'venue': '6', 'keywords': [44, 23, 322, 145, 178, 255], 'year': 2018, 'coauthor': [888], 'target': 43}

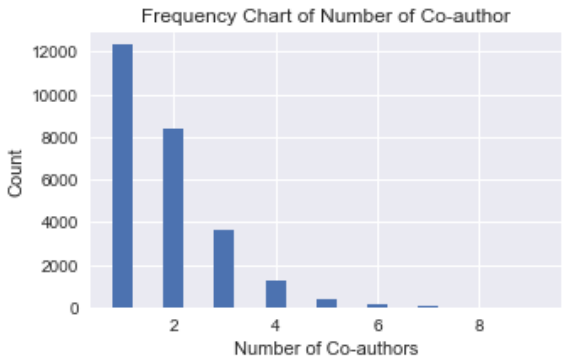

Figure 1: Frequency chart for number of coauthors

Figure 2: No. of Papers written per author (Sorted)

As figure 1 depicted, a majority of the papers only have one author, and as the number of coauthors increases, that frequency decreases. Another issue we might face is class imbalance if we model this as multi-class classification. As we can see, there is a heavily right-skewed distribution shown in figure 2. Some author has coauthored as many as 188 papers, while some have only coauthored as low as three papers.

Data Preprocessing

Data preprocessing and feature engineering plays a major role in machine learning processes, which helps us to get the best out of our data. Several methods has been utilise to fit our approach.

Bag of Words and One-Hot Encoding

To start with, we treated the problem as a multi-class classification problem, where each author is a class label. We first apply Bag-of-Words technique to convert the given *keyword list* into a fixed length vector representation. Since the keyword attribute only records presence of certain keywords, therefore the presence of words will be mark as a boolean value, corresponds to the absence (0) or presence (1). One-Hot encoding methods has also been applied to *coauthor*. As for *venue* and *year*, we decided to simply drop these two categorical attributes to reduce the dimension of the already big dataset. The result of this preprocessing step is a training set with 48,000 instances and 2,802 columns of attributes, with 2,302 authors as class labels.

Negative Sampling

Another approach is to treat the problem as a binary classification, where we categorize our data into two buckets: the proposed target is indeed the genuine coauthor *(positive class)*, or it is not *(negative class)*. However, given the training instances are all positive, where all authors are the actual author of the given paper. Therefore, we first perform random sampling on the training set by randomly picking authors other than the current paper's authors and assigning them as negative samples. One downside of this approach is that it is too simple, given the randomly sampled authors might also have written papers on a similar topic.

Negative Sampling with Cosine Similarity

Thus, we decided to perform word embedding using *Word2Vec*, which allows keywords with similar "meaning" to have a similar representation in vector space. Therefore, when performing negative sampling, we can avoid selecting authors that have written similar papers by computing the cosine similarity between two keywords vectors. We expect a performance improvement by reducing the number of false-negative samples.

Predictive Modelling

Here, we choose to use a Bernoulli Naive Bayes classifier as our baseline model. And we hope that we will be able to improve our performance by using more advanced neural network classifiers. A summary of models and a subset of the model parameters used along with the AUC score obtain from Kaggle in-class competition are listed in table 1.

Bernoulli Naive Bayes Classifier

Bernoulli Naive Bayes assumes all features are generated from a Bernoulli distribution, which is similar to our Bag-of-Words encoding method, where 1s in the feature vector means "keyword occurs in paper" and vice versa. Classification is done by calculating the probability of an instance being each class, and the class with the highest posterior probability is the prediction outcome. However, this classifier has a strong assumption on feature independence. Therefore it is not ideal for our problem, given that keywords and authors might share interactions.

Artificial Neural Network with Embedding

A significant drawback with One-Hot Encoding is that feature mapping is uninformed, in which that similar keywords or authors are not placed closer to each other in the vector space. Our following approach is to utilise the power of the neural network with embedding. Embedding gives us a way to use a dense representation where similar keywords or authors will have a similar encoding. The embedding weights will first be randomly initialised like other hidden layers. Then during training, the weights will be gradually adjusted through backpropagation. We trained the deep learning model using the data obtained from the negative sampled dataset as a binary classification model, with a RMSProp optimizer and *learning rate* $= 0.001$ for ten epochs to minimise the loss. Using such a configuration provides a significant performance boost to our AUC score compared to the simple Naive Bayes classifier.

Multi-class Logistic Regression

Our last approach is to train a Multi-class Logistic Regression with the Bag-of-Words dataset, which is an extension of the logistic regression that adds support for multi-class classification. Evaluation is done using stratified-5 fold

cross-validation since a significant class imbalance problem is presented in the training data. Although this model gives a great performance boost to our AUC score. But it suffers from overfitting, as our model is over-parameterized.

Multi-label Classification

Along the way, we also use multi-label classification like classifier chain, where the first classifier is trained just on the input data, and each following classifier is trained on the input data plus all the previous classifiers. However, the training time quickly becomes computationally expensive and infeasible, since we are constructing a classifier for each label (2,302 in total). Therefore, this approach is soon deprecated.

Model Performance		
Model Name	Model Parameters (subset)	AUC
Bernoulli Naive Bayes	$\alpha = 1.0$	0.58164
ANN with Embeddings	$n_{epochs} = 10$, $\eta = 0.001$, $\mathrm{d}_{embeddings} = 128$	0.74639
Multi-class Logistic Regression	$n_{epochs} = 25$, *batch size* $= 64$	0.87262
Multi-label Classification	Deprecated	N/A

Table 1: Result for authorship verification on test set (AUC scores are obtained from Kaggle)

Model Evaluation

Multi-class logistic regression provides the best AUC among the three classifiers we trained. And it might be suitable for authorship verification since it generates a pretty confident probability score for whether the proposed target author is correct or not. However, we noticed overfitting is a major problem during the cross-validation process due to overparameterization, which might be due to the model's linear nature and not handling highly sparsed data well. As for the artificial neural network, given all the keywords sequence are varied in length, perhaps a recurrent neural network might be better for such application. Another probable reason why the neural network did not perform so well might be our naive negative sampling strategy, since using *Word2Vec* and cosine similarity for sampling does not seem to improve the performance a lot.

Conclusion and Future References

In conclusion, we briefly summarised how various models could be used for authorship verification. We first trained a Bernoulli Naive Bayes classifier as our baseline model that achieved an AUC of 0.58164, and further improved our verification performance using Multi-class Logistic Regression and Neural Network. Finally, we scored 0.87262 on the Kaggle public in-class competition.
For future references, we might be able to improve our classifiers by exploring the structural relationships between authors, where we treat each author as a node. And train a graph embedding that describes, for example, how similar two authors are based on their previous co-authorship and written papers. This could significantly enhance our negative sampling strategy and can be used directly as a pre-trained embedding in our neural network.

References

[1] Jain, S. (2020). *Multilabel Classification.* Analytics Vidhya.
https://www.analyticsvidhya.com/blog/2017/08/introduction-to-multi-label-classification/

[2] Brownlee, J. (2019). *What Are Word Embeddings for Text?* Machine Learning Mastery.
https://machinelearningmastery.com/what-are-word-embeddings/

[3] Halvani, O Graner, L Vogel, I. (2018).
Authorship Verification in the Absence of Explicit Features and Thresholds.
10.1007/978-3-319-76941-7-34.

[4] Jawahar, G. Ganguly, S. Gupta, M. Varma, V. Pudi, V. (2016).
Author2Vec: Learning Author Representations by Combining Content and Link Information.
49-50. 10.1145/2872518.2889382.