Machine Learning for Authorship Attribution Problem

Nofaldi Putranto (1332849) Lulu Ilmaknun Qurotaini (1328810) Dwi Putra Handi Pareh (1239863)

1 Introduction

Authorship attribution is the task of identifying the author of a given document. It has been studied extensively in machine learning, natural language processing, linguistics, and privacy research and is widely applied in real-world settings. Establishing the provenance of historical texts is helpful in placing important documents in their historical contexts. In this assignment, we undertook a variety of machine learning methodologies to build a prolific authors attribution prediction model in several publications. Given the information about authors, year, venue, title, and abstract of publications, the machine learning model is designed to predict if the author of the publication is a prolific author and which prolific author writes it.

2 Evaluation Method

Every model and feature implementation in this report is evaluated by F1-score of the hold-out cross validation to the training data. F1-score is a weighted average of precision and recall. Meanwhile hold-out cross validation is splitting the training data into two parts. A part is responsible for training and the other part is used for testing.

3 Initial Feature Engineering

We extract machine-readable features from the given raw attributes. The aim of this step is to make the model understand important information in our data and eventually improve the performance of machine learning models. We perform evaluation based on some combination of features to see which features contributes importance into our model (See Table 1). The evaluation is done using artificial neural network model (See Section 5.2).

To transform the authors and co-authors attribute, we utilize a method called one-hot encoding[1]. For each authors j, we create a matrix $A_{nxl} = [a_{ij}]$ where $a_{ij} = 1$ if and only if author j contributes to paper i and 0 otherwise. Besides, we are also inspired by a graph network concept to extract a feature of adjacency matrix between the prolific and non-prolific authors[2]. An adjacency matrix $M_{pxq} = [m_{ij}]$ describes the strength of collaboration of each non-prolific authors with prolific authors. The element m_{ij} is added by 1 every time non-prolific author j is found collaborating with prolific author i in a paper. Subsequently, we aggregate the sum of m_{ij} for all non-prolific authors j in each paper as features of size nxp.

Table 1: Performance result with different combination of features

Schenarios						**F1-score**
1	2	3	4	5	6	
		✓				0,22125
	✓	✓				0,30291
	✓		✓			0,34708
			✓	✓	✓	0,30958
✓						0,44930
✓	✓		✓		✓	0,41092
✓			✓	✓	✓	0,36458

Description:

1. Co-author encoding
2. Adjacency matrix
3. Bag of words
4. TF-IDF (title and abstract)
5. Venue encoding
6. Year

In relation to abstract and title, we represent attributes using bag of words (BoW) and term frequency–inverse document frequency (TF-IDF)[3]. Both methods calculate the co-occurrence of words in each document. In other words,

[1]Jalaj Thanaki. 2017. Python Natural Language Processing: Advanced machine learning and deep learning techniques for natural language processing. Packt Publishing.

[2]Gupta, P., Maji, S., amp; Mehra, R. (2022). Stress and machine learning: Future with possibilities- a bibliometric approach. Journal of Scientometric Research, 11(1), 37–46. https://doi.org/10.5530/jscires.11.1.4

[3]Ibid

a matrix TF has element $tf_{t,d} + 1$ everytime a word t is found in document d. However in TF-IDF, the calculated frequency is weighted by the inverse document frequency. Eventually, we get $w_{t,f} = tf_{t,d} \times log(\frac{N}{df_t})$ for each word t in document d. For the venue and year attributes, we encode the venue attributes using one-hot encoding. As for year attribute, we use it in its raw form without any transformation.

From the result in Table 1, we conclude that the co-author encoding feature contributes the most to our evaluation performance. In fact, co-author encoding alone gives the best performance without other features involvement. We use this initial findings to develop our models in the next section. However, we keep all the features for any possible improvement in advance tuning.

4 Baseline

A baseline provides a point of comparison for measuring the performance of machine learning models. We designated naive Bayes classifier as our baseline performance because it is a simple model that is easy to implement and run. The naive bayesian equation is

$$P(Y_k = 1|X_1, \dots, X_n) = \frac{P(Y_k = 1) \times \prod_{i=1}^{n} P(X_i|Y_k = y)}{P(Y_k = 1) \times \prod_{i=1}^{n} P(X_i|Y_k = 1) + P(Y_k = 0) \times \prod_{i=1}^{n} P(X_i|Y_k = 0)}$$

where X_i denotes the presence of coauthor i in data and Y_k denotes the presence of author k in data. Training data will be used to calculating $P(Y_k)$ and $P(X_i|Y_k)$. We predict the $P(Y_k = 1|X_1, \dots, X_n)$ when the result of probability greater or equal to 0.2. We choose 0.2 as the decision boundary after we have done 10 cross-validation by using highest f1 as decision parameter. The f1 score of this version of Naive Bayes is 0.528. This is our performance baseline.

5 Model Selection

We consider penalized Support Vector Machine (SVM) and Artificial Neural Network (ANN) as our algorithm candidates for our model.

5.1 Penalized Support Vector Machine (SVM)

The task of predicting if there is any prolific authors and which prolific authors, among the predefined 100 prolific authors, write a publication can be considered as a 100 binary classification problem. Each binary classification is responsible for each prolific author.

The SVM algorithm generates a hyperplane decision boundary that separate the observations into two classes. In the case of authorship binary classification for authors, the problem becomes:
$argmin_{w_i} \|w_i\|$ subject to $y_{ij}(w_i' x_j + b) \geq 1$ for $i = 1, ..., 100$ and $j = 1, ..., n$ where n is the number of observation in the training dataset. y_{ij} represents whether publication j is written by author i.

Some modification of SVM that weighs the margin proportional to the class importance is needed due to the tiny amount of prolific authors compared to non-prolific authors in the dataset.

The resulting f1 score is 0.392. It is below the baseline score.

5.2 Artificial Neural Network (ANN)

Given the huge and possibly non-linear dataset for the classification task, we consider ANN to be an alternative candidate for the algorithm. We implement ANN with the following specification: number of input nodes is the same as the number of author matrix column, number of layer is 4 (balanced trade-off between the rate of converging and the time consumed for each epoch), and 100 output nodes which represents binary predictions for 100 prolific authors. We also used tanh as the activation function for all the hidden layers and set an initial bias proportional to the author occurence in for each authors. Using the algorithm with the one-hot encoding author matrix, we split the train dataset, train the model, and test the performance. The resulting f1 score is 0.535. Based on the result, ANN performed better than SVM and baseline model.

6 Hyperparameter Tuning

We introduced drop rate in our ANN model to regularize the model in order to reduce overfitting. The results of our hyperparameter tuning trials is in the Table 2. The evaluation result showed an improvement over ANN model without a drop rate.

7 Author Order

We took into account that the order of authors in the train dataset may improve the prediction performance. Consider y_{ij} for $i = 1, ..., 200$ and $j = 1, ..., n$ where n is the number of observation in the training dataset. y_{ij} for $i = 1, ..., 100$ represents whether author i is the first author of publication j, meanwhile y_{ij} for $i = 101, ..., 200$ represents whether author i-100 is the non-first author of publication j. We retrained the ANN drop rate model with a new set of response variable. We saw an improvement on the result of the trial as written on Table 2.

8 Title and Abstract

We revisited title and abstract in an attempt to complement coauthors feature in our model. A number of most common words was removed in each trial. We argue that the most frequently used words do not have a distinguishing power to predict authors. Some of them can be stop words as well. After removing the words, we extracted TF-IDF from the remaining words. The results of the trials is in Table 2.

There is an increase in cross validation training data split performance. However, later we found out it did not get translated into Kaggle testing performance.

Table 2: Trials with different specifications

Features	Model Specification	F1-score
Co-authors encoding	Naive Bayes (Baseline)	0.528
Co-authors encoding	Penalized SVM	0.392
Co-authors encoding	ANN without drop rate	0.535
Co-authors encoding	ANN drop rate = 0.4	0.543
Co-authors encoding	ANN drop rate = 0.25	0.552
Co-authors encoding	ANN drop rate = 0.3	0.566
Co-authors order	ANN drop rate = 0.3	0.572
Co-authors order, TF-IDF (minus 400 most common words)	ANN drop rate = 0.3	0.570
Co-authors order, TF-IDF (with 400 least common words)	ANN drop rate = 0.3	0.582
Co-authors order, TF-IDF (with 201-600 least common words)	ANN drop rate = 0.3	0.567
Co-authors order, TF-IDF (with all words but using PCA for dimension reduction)	ANN drop rate = 0.3	0.495

9 Conclusion

The best f1-score we produced is from the coauthor feature and 100 authors differentiated between first author and non-first author trained in ANN model with drop rate equals to 0.3. We used that result as our final submission. The model selection, hyperparameter tuning, and the order in author response feature that we implemented is justified. We are also confident in the way we use the coauthors feature. However, the same cannot be said with the other features. Further feature engineering still needs to be applied for the other features. If we have more time, we would like to experiment on more combination of words removal in title/abstract before extracting TF-IDF and research more to look for patterns in combination of year and venue. Consequently, we did not use other features in our final submission. There are several other conclusions that can be made from this project.

Information of coauthors is critical in predicting the author of a publication. The result makes sense as there are not many authors who have a professional relationship.

ANN did not show a significant improvement over the Naive Bayes baseline. This suggests that in order to fully take advantage of the technical prowess of a complex model like ANN, good features are needed.

With coded/masked words in the dataset, text manipulation is limited. For example, we cannot used the state of the art pretrained word embeddings. Other limitations include computational power and memory size required to build the models. Given the circumstance, we have devised a model with a considerable predictive capability.