COMP90051 Statistical Machine Learning Project 2

XiaoCASE (Groups 88)
Yu Weng(1235216), Leting Zhouli(1240313), Xiongfei Guo(1196869)

Abstract

This report will demonstrate a systematic process of data pre-processing and classifier model selection based on the given task of Authorship attribution for predicting 800 papers. Each paper has in total of 100 potential prolific authors involved in writing, and the result for each prediction may be zero, one, or many of these authors, namely multi-label classification, imbalanced data handling, and classification model selection, which altogether have brought the team to the Top 10 on Final Kaggle competition.

Keywords— Multi-label classification, Imbalance data handling

1 Overview and Imbalanced data handling

1.1 Data Overview

With the training data given in a JSON format file contains in total of 25.8K papers, and each paper is a dictionary with keys of five important features including authors, year, venue, title, and abstract. And quick exploitation of the training data shows that the label distribution is highly imbalanced, with almost around 70% of the training data not containing prolific authors. However, in the test set, there are only 25% of the data without prolific authors.

1.2 Imbalanced Data Handling

Synthetic Minority Over-sampling Technique (SMOTE)

It is an oversampling method. Instead of making duplicates which could cause the over-fitting problem, it operates by producing synthetic samples from the minor class[1]. Therefore, SMOTE is our first thought to oversample new data samples of minority labels. However, SMOTE method from the imbalanced-learn library does not support multi-label classification problems. Therefore, because of the time limitation of implementing this technique by our own, we switched to other methods.

Resampling

There are two ways to deal with highly unbalanced data. The first one is removing the samples from the majority class(Under-sampling). And the second one is to add more samples by making copies from the minority class (Over-sampling). However, despite the advantages of balancing classes, the over-sampling method could cause an overfitting problem, and the under-sampling method could cause information loss.

Class Weights

Assigning different class weights is another way to deal with highly unbalanced data. The core concept is to penalize the minority class for misclassifying itself by increasing class weight while simultaneously decreasing class weight for the majority class. Both resampling and assigning different class weights methods will be implemented and compared.

2 Feature Engineering

Author and venue features are categorical features, while abstract and title are text features, which are all mapped to a different integer number within a different range. And the year is another categorical feature, but it is not used as a feature for training the model. Because the training set does not contain any papers from a new year, while all the papers from the test set are from a fresh new year.

2.1 Categorical features(Author and Venue)

One-Hot Encoding

One-Hot Encoding is one of the most popular approaches for dealing with categorical data by creating new binary columns to indicate the presence of a specific categorical feature. The venue and author features are the most suitable features to do One-Hot Encoding. Due to the large dimension of the authors, it is better to do some dimensional reduction methods, such as the Truncated SVD and PCA. But, dimensional reduction methods could lead to data loss. Because the author's feature is one of the most important features of this problem. Instead of using dimensional reduction techniques, we allocated the last dimension to all the authors who have not collaborated with the prolific authors to further reduce the dimension.

Node2Vec

Node2Vec is a random-walk-based network embedding method [2]. It learns a low-dimensional representation for nodes in a graph $G(Vertices, Edges, EdgesWeights)$. And the generated embeddings can be utilized for downstream tasks, such as classification. As for this Authorship attribution classification task, it is a great way to treat each author as a node, and if some of them had cooperated once, an edge will be created between them. The corresponding edge weight can be seen as how many times they cooperated. As for the solo author, they will be treated as a single node that is connected by itself. However, due to the time constraint, we did not successfully retrieve the node vector embedding for each author.

2.2 Text features(Abstract and Title)

Bag-of-Words & TFIDF & Word Embedding

The bag-of-Words model is a simple model for representing text data by creating a vocabulary size dimension sparse array and counting the occurrences of tokens in the text. And TF-IDF(term frequency-inverse document frequency) is another measurement for quantifying the importance of the words in a document among a collection of documents. However, these two representations mentioned above are incapable of learning the semantic relationships between words which is of vital importance when dealing with text data. Therefore, word embedding method is utilized in this project for reconstructing word vector representation based on a window of neighboring words for better capturing the semantic relationships.

3 Multi-label Classification Techniques(Models)

3.1 Baseline

The ClassifierChain model is a multi-output model that arranges basic binary classifiers into a chain to take into account the correlations between labels. And this model combined with logistic regression(solver = Liblinear) base classifier is selected as the baseline model for this task, due to the relative low performance in high-dimension data.

3.2 Feedforward Neural Network

FFNN is selected as the main model for this classification task. And two types of FFNNs have been built and compared with different feature engineering methods. The last layer of both neural networks produces 101 neurons, of which the last neuron is utilised to distinguish all non-prolific authors. And the first 100 neurons represent 100 prolific authors. The binary cross-entropy loss is calculated for each neuron, and the final loss is the sum of each label loss divided by number of labels(101).

Multilayer Neural Network

Layer	Units	AF	Dropout Rate
Input Layer	7170	N/A	N/A
Dense	2048	tanh	0.3
Dense	1024	tanh	0.3
Dense	512	tanh	0.3
Dense	256	tanh	0.3
Dense	101	sigmoid	0.3

(a) Network Model parameters(Adam(lr = 0.0009))

	Text	Author	Venue
OHE		✓	✓
BOW			
Word Embedding	✓		

(b) Features implementation

Table 1: Model parameters and corresponding feature engineering

The MLP model parameters with Adam optimizer(learning rate fixed at 0.0009) is shown in Table 1a above. The dropout rate 0.3 is utilized for further preventing overfitting. The activation function tanh is chosen over relu for preserving enough of a momentum to keep convergence going since the features are all scaled between range 0 and 1, and the allowance for avoiding Vanishing/Exploding Signals. Meanwhile, callback function is utilized for saving the lowest loss model. As for the features, the abstract and title features are concatenated together to use Word2Vec function(epoch =10) from gensim library to retrieve the training word embedding. And the window size is set to 5 to further increasing the length of neighbor words by considering more neighbor words of the given text for better result. The final dimension is 128 for text feature after the average pooling.

Embedding-based Neural Network

However, the feature dimension is still large, therefore word embedding is considered not only for abstract and title but also for both author and venue features.

As demonstrated in the table below, all the features abstract, author,title and venue are all padded with max length of 128,100,32 and 1.And for each integer, it is mapped to a dimension of 128. After that, each embedding layer will go through an average pooling layer to further reduce the dimension same as the embedding dimension 128. Consequently, all the embedding features are concatenated together which will be fed to fully-connected layer later.

Layer	Units	AF	Dropout Rate
4 Input Layers	(128,100,32,1)	N/A	N/A
4 embedding layers	(128,128,128,128)	N/A	N/A
3 average pooling layers	N/A	N/A	N/A
Layer concatenation	128*4	N/A	N/A
Dense	512	relu	0.3
Dense	256	relu	0.3
Dense	256	relu	0.3
Dense	101	sigmoid	0.3

Table 2: Embedding-based Neural Network

4 Results and Discussion

Instead of implementing sample-based F1 score for tensorflow framework, Macro F1 score from tensorflow_addons library is utilized to provide a basic overview of the sample-based F1 score.

Model	ClassWeights	Upsampling	Downsampling	Test	Val(20%)	Kaggle
ClassifierChain			✓	N/A	0.510	0.382
MLP	✓			0.966	N/A	0.540
MLP		✓		0.992	0.774(macro)	0.5110
EmbeddingNN	✓			0.974	N/A	0.434
EmbeddingNN		✓		0.992	0.714(macro)	0.403
EmbeddingNN			✓	0.987	0.654(macro)	0.431

Table 3: Overall Result table for all training models with different imbalanced data handling methods

Overall, observation shows that all the methods returned a relative high f1 score on validation set, proving the dataset is lack of representation(cannot provide a trustworthy model performance). When applying the class Weights method by setting the last label to 0.3, while remaining the other labels to be 1. As for class weight method for handling imbalanced data, splitting training dataset is not implemented. Because some information can be lost if the dataset is split, particularly for some labels with only around 200 instances. Therefore, it becomes a tricky problem by only look at the training loss(not too low cause overfitting while not too high cause underfitting) and the corresponding kaggle score for hyperparameters tuning (Table 1a).

Model comparison

As for the comparison between two FFNN models, there are three main basic reasons why the performance of the embedding-based neural network performs no as well as the normal multilayer neural network.

1. Word Embedding for author feature lacks robust evidence support.
2. Duplicated embedding outputs for different instances(author combinations) could occur after going through the global average pooling layer.
3. Embedding outputs after the average pooling layer could cause relative high information loss.

Future Improvements

Due to the time and resources constraints, we have to miss out a few ideas for this project. This includes building an author-based GNN to utilize Node2Vec for retrieving a low-dimensional representation of the author feature, or implementing SMOTE for producing more robust synthetic samples for minority labels. And the current MLP model can be further utilized if more data is provided to achieve a better result.

References

[1] K. W. Bowyer, N. V. Chawla, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *CoRR*, vol. abs/1106.1813, 2011. arXiv: `1106.1813`. [Online]. Available: `http://arxiv.org/abs/1106.1813`.

[2] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," *CoRR*, vol. abs/1607.00653, 2016. arXiv: `1607.00653`. [Online]. Available: `http://arxiv.org/abs/1607.00653`.