Registration number: 1908054

Text Analytic Assignment II

School of Computer Science and Electronic Engineering

University of Essex

ABSTRACT

Given the multi-label text classification problem in digital publication document indexing, traditional machine learning and neural networks methods have been proposed for this problem. The qualitative variable, full-text from the documents are required for creating a good performance multi-label text classification system. On the other hand, the qualitative variables like full-text of the digital publication documents are not likely available because of some specific circumstances compared to the quantitative variable such as the title text of the digital publication document. The main disadvantage of using only the title text of the digital document is that the learning algorithm will lose some information about the document, which could lead to poor prediction performance. Many researchers have been trying to create a system which is based on the title text only while trying to keep the same level of the prediction performance of full-text based system.

This study carries out the large scale multi-label classification or known as eXtream Multi-Label Classification (XMLC) with two scientific datasets, PubMed and EconBiz using title text only. We applied one-vs-rest and label powerset as a multi-label classifier with traditional machine learnings such as Support Vector Machine, Logistic Regression, and Naïve Bayes with the use of a famous framework, Scikit-learn.

The classifiers' performance is tested with an average f-1 score from Kaggle evaluation template. The evaluation results showed that the Logistic Regression classifier performed best compared to the other two learning algorithms. On the other hand, the label powerset method is not suitable in this large scale; the combination of the label was too big to be usable.

KEYWORDS

Titel text classification, multi-label classification, document annotation, XML.

1 Introduction

Multi-label classification has been adopted to text analytics, and this has reduced human workload in digital document indexing task. The unstructured data are indexed with adequate labels. Therefore, the unstructured data become structured data which make it a lot more convenience to retrieve the relevance information. As the time past, the potential labels have grown to the point that the problem is considered as an eXtream Multi-Label Classification problem. The eXtream Multi-Label Classification problem or XMLC is when the pool of label is a significant big, while the actual labels to be assigned to the data object is just a few numbers.

The objective is to review some similar works that have been done in the past then apply the obtained knowledge to this study. The aim is to design and implement the title text as a source for developing an automated semantic document annotation. Also, the evaluation of the XMLC will be carried out.

In the literature review, previous work of full-text and title text only for automated semantic document annotation will be reviewed as well as the state-of-the-art approach such as of title text only with a deep learning-based approach for XML system. According to the assignment instruction, the provided papers cover all the required information already.

The multi-label text classification usage domains are introduced after the literature review section. The systems that have been helped by the multi-label text classification will be explained with an example.

The design of the system is illustrated in the system design section. The diagram will be used to help explain each process of the system. This includes input, pre-processing, feature selection, model training, and evaluation.

Next, the implementation of the XML classifier is explained in the function level. In this section, the configuration of each process is presented. As in this study, we will try to make use of the useful framework like Scikit-learn in the Python library. The motivation of the system design and the potential alternative methods are introduced as well.

The results comparison will be discussed with some explanation and assumption. In this section, the previous work results are compared with this study results followed by the conclusion of this study.

2 Literature Review

The previous works about XMLC are summarized with the main focus of getting to know the advantages and disadvantages of each method. The automated semantic document annotation systems with title text and full-text were designed and implemented by the expert researchers. The traditional machine learning classifiers and neural networks classifier were involved in the previous researches.

2.1 Using Titles vs. Full-text as Source for Automated Semantic Document Annotation

Similar research [1] implemented and compared the classifiers' performance using title and full-text as a source for training the classifiers. This research claimed that even though training classifier using full-text produce better performance classifier, but the full-text of the digital documents mostly are unavailable. They are restricted by copyright and legal control. It is no doubt that using full-text can give the learning algorithm more pieces of information, but using only the title text will make the learning algorithms have more training sample because of its high availability.

They aimed to make the Simple Knowledge Organization System (SKOS) that used digital document's title text to build classifiers only. Besides, the performance has to be competitive with full-text based system.

It was challenging because of the classifier will lose informative feature while training. However, the tradeoff of using the full-text compared to title text was less training samples. Gathering the title text of the digital documents is more straightforward and more availability.

They achieved their goal, which was to develop the best performance classifier that can be on par with the full-text system. They designed the SKO system with the following process:

1. **Pre-processing** – the first process was to lowercase text in the document and then lemmatized text using Wordnet library [4]
2. **Vectorization** – the concept is to transform the text data into vector representation which can be used as the classifier input. They used term frequency (TF) of the word as the training feature. In addition, they applied the finite state machine to extract the term phrase to match the possible match of words sequential. They assumed that the longer phrase would have more specific categories.

   Then, they applied the discounting frequent terms and concepts. Inverse document frequency (IDF) [2] as a re-weighing scheme. The diving by zero avoidance is also added by adding one increment number to the count of the term.

   Thereafter, the extension of IDF, Okabi BM25 [3] was applied to modify the weight with the text length. Finally, the term and the concept was combined with L2 normalization method.
3. **Classification** – the second process of their pipeline was training the classifier then use it to predict a set of labels of the document.
4. **Evaluation** – the last process is to test the performance of the trained classifier. The dataset was split into 90% as training data and 10% for test data. Then, they applied 10-fold validation which every 10% of the data will be used as the test data once. The predicted labels were compared with the gold standard and computed the f-1 score (harmonic between precision and recall).

They addressed the problem by trying a set of machine learning algorithms using the developed pipeline. Not only the famous machine learning algorithms were used, traditional learning algorithms, Naïve Baye, Rocchico, and Logistic Regression were also used. Also, prominent learning to rank was also used [5].

The results showed that using the title text only for SKOS could be on a par with full-text SKOS. They evaluated the classifiers via the average of the f-1 sample with four large

datasets. The evaluation results showed that lazy learners performed poorly with the title text only. On the other hand, eager learners, such as Logistic Regression and Feed-Forward Neural Networks, performed well in this task. The sample average f-1 score of title TF-IDF was 0.351, and the best title was CTF-IDF 0.368 on EconBiz dataset with the kNN classifier.

2.2 Using Deep Learning for Title-Based Semantic Subject Indexing to Reach Competitive Performance to Full-Text

The semantic subject indexing systems are useful for expert annotators who work at digital libraries as the system can help classify the categories of the digital documents. Also, users that use digital libraries. The appropriate index of the document will help them search for the relevant document faster and more precise.

Similar research [6] developed a deep learning approach using only the title text of the digital scientific publication document. The deep learning method has been known as the state-of-art in text classification field [10].

They implemented the title text semantic subject indexing system using the deep learning approach with two large English datasets Econbiz and PubMed. They developed the title text system because it is more flexible to use the title text as a source. Using full-text can be difficult since most of the digital scientific documents are prevented from access to the content inside. Some of them are available only for the abstract, not the body content. This is why creating the title text classification system can give wider usage. They claimed that in making the semantic subject indexing system using only the title text cannot reach the same level of prediction performance if the classifier is trained with the same amount of sample.

Therefore, using the title text can benefit from collecting the training samples. According to [7] full-text document require high computation to the library. However, the title text system considered less accurate compare to the full-text system if the number of training samples is equal. A full-text document gives more indication information even to a human than the title text. The human expert often makes a better annotation with the full-text document rather than a title text document.

Although, they argued that the machine might treat this problem differently. A machine might need a million of training sample to differentiate the different category [8]. One main problem is that the full-text data is not this big available. It requires a human expert to annotate it, but this solution is expensive and time-consuming. They take advantage of the title text data, which is far more available than the full-text to build the classifier.

They developed the methods as follow:

1. **Training Procedure** – they first drop the Binary Relevance as encoding labels technique in the in this XML problem, because it requires more training data with the combination of labels as many as they can be. The neural networks have more a better way to deal with this issue. Therefore, the activation function [1], such as softmax is used for this multi-label text classification process.
2. **Multi-Layer Perceptron** – the baseline model was multi-layer perceptron (base MLP) with TF-IDF to create the bag-of-word of a unigram. Regulation dropout was applied after the hidden layer. They also applied some alternative method of using the FastText multi-label classification as a linear BoW and created bag-of-bi-grams (MLPs).
3. **Convolutional Neural Network** – they implemented the 1-d CNN with the use of core architecture from [9] sentence prediction task. The CNN model worked on word embeddings with the pre-trained model then fine-tune it after that. They defined the window size of the word vectorization generating and found that the window size of 2, 3, 4, 5, and 8 produced a better result than the original window size by [9].
4. **Recurrent Neural Network** – the RNN was the final model that combined with two techniques in of NLP. As the RNN produces the output every time step, so they chose the last output then calculate the summation weight average. Then they sent the output to LSTM that read the sentence from left-right then right-left to make the model learn more information about the sentence [10]. Finally, they added another LSTM on each of each other for the more capability.

The dataset was processed into subsets then sent them to the deep learning algorithm. Then they evaluated the performance. The base-MLP model outperformed the traditional approaches. The result showed that the deep learning model on the title text could be as good as the full-text system when we have enough training data.

The MLP performance was at the same level as the best full-text model on the EconBiz data. On the other hand,

when the model was trained with the entire titles data, the performance was better than the best full-text model.
Finally, for the PubMed dataset, the RNNs model was the best performance and almost achieved the performance of the best full-text model. CNN got the lowest accuracy score compared to the other three models.
In conclusion, MLPs model performed best since it outperformed three out of four times in both the datasets test.

3 Multi-Label Text Classification in Use

The multi-label text classifications have helped humans reduce their workload. The systems can be used in many fields, as follow:

1. **Digital Library** – the multi-text classification systems can help annotators to annotate the digital document. The annotators only need to verify the correctness of the prediction.
2. **E-commerce Suggestion System** – the multi-label text classification can help retrieve the relevance product item information based on the customer search. The system classifies the product category by the product description. For example, one product can belong to many categories, such as a toy, car, and controllable. Therefore, this improves both the customer experience and the store.
3. **Movie Streaming Website** – the multi-label text classification systems help the movie streaming websites by categorizing the type of movie. The movie can have many types tagged on it. For instance, violence, sci-fi, and sexual. The movie types can be classified by its description.
4. **Hospital** – In the hospital, a doctor usually records the detail about the treatment. So, the hospital can use the multi-label text classification system to classify the possibility of being sickness according to the doctor note. This can help doctors to analyze what are all the possible sickness that the patient might have.
5. **The Topic Analysis of Post in Social Media** – these days, when people start a post in social media, the system will suggest with the tags of the post. This is because the multi-label text classification system classifies the text content in the post.
6. **Search Engine Indexing**– the textual data is increasing exponentially day by day, and it is almost impossible to have human manually index the document. The search engine can use multi-label text classification to auto index new textual documents. So, the most relevant document will match the search keyword. This can improve the search experience of users.

4 XML System Design

According to literature research, the pipeline process by [1] is interesting. Some ideas from the literature review will be applied in the XML system design.

The overall of the XML system is presented as in figure 1 below.

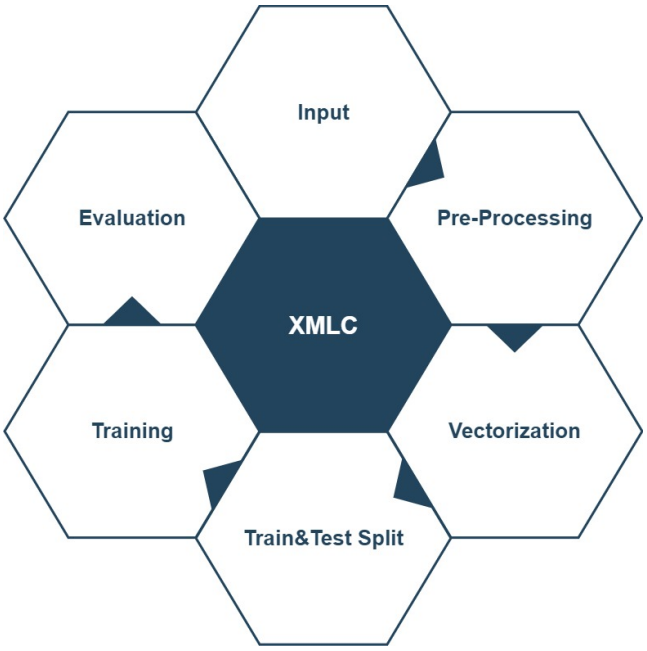

Figure 1 System Design

According to figure 1, each process method is explained as follows:

1. **Input** – the input data (CSV file) is read via this process and will return as a data frame of title text and label value. This study carries two types of data, the data which has both title and full-text and the data that does not have to have the full-text.
2. **Pre-Processing** – the title text input will turn into completely lower case text then lemmatization is applied after that. The library NLTK is used at this process. Also, the label is transformed using the

Multi-label binary from Sklearn library. The output of this process is the pre-processed title text and the pre-processed labels.

3. **Vectorization** – after text pre-processing, the TF-IDF vectorizer is used to create the training feature. The output of this process is the weight score of words (terms).
4. **Train & Test Split** – the dataset is split into 90% for training and 10% for testing.
5. **Training** – the training features are used as input for the classifier to learn. One-vs-rest classifier from the Scikit-Learn framework is used as the multi-label algorithm with the eager learner, Logistic Regressing algorithm from the Scikit-Learn as well.
6. **Evaluation** – the 10-fold validation is applied for testing the performance. At this process, the classifier will be trained ten times and test ten times with different training and testing data. Every 10% of the data will be used as the test data. Then the sample-average f-1 score is calculated.

Some of the processes have been motivated by previous research [1].

5 Implementation

In the implementation, some of the parts have been borrowed from [11]. The system is developed on Python language. In this experiment, each set of data will feed to the pipeline one at a time.

5.1 Input

Two datasets are used, EconBiz and PubMed. The datasets are in CSV format. One data set is used at a time by using read CSV in the program and that store it in data frame format. Only the title and label column values are used.

5.2 Pre-processing

In the pre-processing, the title text is transformed into the entire lowercase character and then lemmatize it to reduce the various form of the word. Then, the label value is processed to multi-label binary from using pre-processing from Sklearn library. Finally, only the pre-processed title text is sent to the vectorization process. The label is sent to the training process. Figure 2 below is an example of text before and after lowercase and lemmatizing.

```
Behavioural effects of obligations
behavioural effect of obligation
```

Figure 2 Text Pre-Processing

5.3 Vectorization

This process extracts training features from the pre-processed title text. TfidfVectorizer from the Sklearn is applied for counting the occurrence of the word (Unigram) and calculate with the IDF [12]. The TF-IDF weight of words is produced for classifier input. This process transforms words into weight scores in the format the machine learning algorithm can understand.

5.4 Train & Test Split

At this process, the dataset is split into 90% and 10% for training and testing respectively.

5.5 Training

The training features and pre-processed labels are used in this process. The selected learning algorithm, Logistic Regression, is trained with the one-vs-rest multi-label scheme. Both of the algorithms are applied with the used of multiclass from the Sklearn framework. The classifier is trained with a portion of 90% of the dataset.

5.6 Evaluation

The classifier performance is evaluated on the test set (10% split data). The sample-average f-1 score is calculated with the 10-fold validation. The training and evaluation processes are called ten times to train and evaluate each time of the model performance.

6 Results and Comparison

In the performance results, we have evaluated the performance of the classifier with two datasets with using entire title text. The selected classifier is Binary Relevance Logistic Regression (BR+LR). Four models were trained. First, the Binary Relevance Logistic Regression was trained with EconBiz both entire titer text and title text data that contains full-text; second, it was trained with PubMed with the same configuration (subset). However, the PubMed data is too big, so only 5% of the data was used as a subset of it.

Input	Classifier	EconBiz	PubMed (subset)
Entire Title	BR+LR	0.38	0.16
Title of Full-text	BR+LR	0.25	0.09

Table 1 Models Performance

Table 1 shows the experimental results of two settings on two datasets. It is seen that the results followed the same trend in the previous literature [1]. Using all the title text data could a better performance model than using the title text data that consists of full-text.

Using the EconBiz entire title data with the developed system produced the sample-average f-1 score at 0.38 while the title of the full-text sample-average f-1 score was just 0.25. For PubMed dataset, only 5% of the dataset was used, the sample-average f-1 score of the entire title model was 0.16 while the title of full-text sample-average score fell behind at 0.09 only.

To compare and contrast the experimental results with the literature performance, our best performance model of the entire title was lower than the performance of Logistic Regression using title text [1]. They achieved 0.42 sample-average f-1 score with the same dataset.

Also, comparing with the state-of-the-art, deep learning performance from the literature [6], the title all model, MLP achieved 0.504 sample-average f-1 score.

The compared resulted based on EconBiz dataset only because the models of PubMed were not trained with the entire data.

Our developed pipeline is not as good as the previous work pipeline, especially the deep learning system (SOTA). They implemented a more complex system with extensive computation time to unlock the capability of the classifiers in both the vectorization and classification process.

7 Alternative Method

The alternative method of an XML system to improve the prediction performance can be both adding bag-of-bigrams to the vectorization process so that the classifier will learn more information from the additional feature. Also, the data augmentation technique could be used in this XML classification. Data augmentation add the training sample with modification some words by their wordnet similarity. This could give the classifier see more combination of the similar meaning in of the sentences. Lastly, the BERT algorithm [13] could be used to improve the vectorization process performance as it can produce more precise and representative vector representation as to the input of deep learning classifier. Furthermore, BERT itself can be the multi-label classifier by fine-tuning it with the top layers.

8 Conclusion and Lesson Learned

This study aims to design and implement an XML classification system for dealing with the unstructured data.

The previous literature of full-text and title text only for automated semantic document annotation was reviewed as well as the state-of-the-art approach deep learning-based approach to gain basic knowledge before designing the system pipeline.

The title only for eXtream Multi-Label system, the system was developed using Binary Relevance Logistic Regression as a learning algorithm. The system consists of the input function, pre-processing, vectorization, train and test split, training, and evaluation.

Two datasets EconBiz and PubMed were used in this study in two manners, the title text documents that contain full-text and every title text data (not considering the full-text availability).

The best model of this study was using all title text from the dataset with Binary Relevance Logistic Regression. However, the performance was far behind the state-of-the-art model with the same dataset. By comparing our best model with previous work [1], our model performance was a noticeably lower as well. The assumption is that the pre-processing and vectorization process of this study needs some improvements for generating a better informative training feature.

The alternative methods were discussed. The bigrams could be used in the vectorization process to improve the training feature. Applying the data augmentation might also increase the prediction accuracy since it allows the classifier to see more combination of the sentence. The BERT algorithm from Google might be applied for better generating vector representation for neural networks classifier or even use it as a classifier by fine-tuning it.

Finally, the lessons learned are explained as follows:

1. **Desing of XMLC** – the processes of implementing the XML system was learned from the literature review.
2. **SOTA of XMLC Using Title Text Only** – the state-of-the-art of XML system using the title text

only from research [6]. The performance results show that in some models that were trained using the title text data have already outperformed the full-text system.

3. **Python Framework** – we learned how to use a useful framework, Scikit-Learn. This made the implementation much easier because of all the machine learning algorithm are included as well as the evaluation metric, f-1 measure.

REFERENCES

[1] L. Galke, F. Mai, A. Schelten, D. Brunsch, and A. Scherp, 2017. Using Titles vs. Full-text as Source for Automated Semantic Document Annotation, Knowledge Capture (KCAP); Austin, TX, USA.

[2] Gerard Salton and Christopher Buckley.1988. Term-weighting approaches in automatic text retrieval. Information processing & management 24,5(1988).

[3] Stephen E Robertson,Steve Walker,Micheline Beaulieu, and Peter Willett.1999. Okapiat TREC-7:automatic adhoc, filtering, VLC and interactive track. Nist Special Publication SP (1999).

[4] Princeton University.2010. About WordNet. wordnet.princeton.edu.(2010).

[5] Gregor Große-Bölting, Chifumi Nishioka, and Ansgar Scherp.2015.A comparison of different strategies for automated semantic document annotation .InKnowledge Capture.

[6] F. Mai, L. Galke, and A. Scherp, 2018. Using Deep Learning For Title-Based Semantic Subject Indexing To Reach Competitive Performance to Full-Text, Joint Conference on Digital Libraries (JCDL); Fort Worth, TX, USA.

[7] Lukas Galke,Florian Mai,Alan Schelten,Dennis Brunsch,and Ansgar Scherp. 2017. Using Titles vs. Full-text as Source for Automated Semantic Document Annotation.InK-CAP.9.

[8] Damien Brain and G Webb. 1999. On the effect of data set size on bias and variance in classification learning.In Australian Knowledge Acquisition Workshop, AI'99.117–128.

[9] Zichao Yang,Diyi Yang,Chris Dyer,Xiaodong He,Alexander J.Smola,and Eduard H.Hovy.2016.Hierarchical Attention Networks for Document Classification. In NAACL-HLT.1480–1489.

[10] Yann Le Cun,Yoshua Bengio,andGeoffreyHinton.2015. Deep learning. Nature 521,7553(2015),436–444.

[11] Kaggle.com. n.d. Multi-Label Classification Evaluation Template. [online] Available at: <https://www.kaggle.com/hsrobo/multi-label-classification-evaluation-template>.

[12] Journal of Convergence Information Technology, 2013. Research on the Application of an Improved TFIDF Algorithm in Text Classification. 8(7), pp.639-646.

[13] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining.