

**Department of Computing**

**COMP5511**

**Artificial Intelligence Concepts**

**Group Project**

**Topic:**

**Classification for News Titles**

Group 1:

|  |  |  |
| --- | --- | --- |
| **Name** | **Student ID** | **Email** |
| Chan Wai Him | 18003067g | wai-him-edward.chan@connect.polyu.hk |
| Ng Chun Han | 19004507g | chunhan.ng@connect.polyu.hk |
| Bei Ho Yuen Bert | 19004521g | ho-yuen-bert.bei@connect.polyu.hk |
|  |  |  |
| Yu Yip Sheung | 19012876g | yip-sheung.yu@connect.polyu.hk |
| Wong Chi Hang | 19074889g | reeve.wong@connect.polyu.hk |
| Chan Hei Yin | 20002959g | heiyin.chan@connect.polyu.hk |
|  |  |  |

Submission Date: 7th May 2021

**Abstract**

In this project, three models from different approaches were developed to read a news title data set then assign a pre-defined label to it. There are 7 pre-defined labels including sport, business, entertainment, us, world, sci\_tech, and health. This is a multi-class classification that the class labels are mutually exclusive and only one label is assigned to a text. The motivation of the project, description for tools and techniques used, implementation, data, result and observation, finally, the discussion are included in this report.

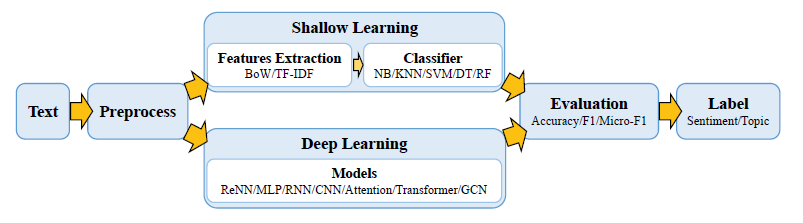
**Introduction & Motivation**

Nowadays, people can search for news online with their own mobile devices to get the latest information at any time and anywhere without purchasing the physical format of newspapers. However, there tons of sources and newspaper publishers releasing news articles 24/7 around the world, it is crucial to have a tool that can classify the news articles into distinct categories and return the result to the readers so that they can access the relevant news topics quickly and effectively.

Therefore, a machine learning model performing news classification to identify topics and categorize them automatically can help people filtering the news they are interested in. Our team aims to explorer the text classification technique and try to build an efficient model for the task of news classification.

**Description**

Regarding the research paper [1] in fig. 1, three built models were targeted to implement approaches from shallow learning, deep learning, and state-of-the-art.



**Fig.1 Flowchart of the text classification with classic methods in each module.**

To build these models, the following libraries were used:

1. Pandas

It is an open-source python library under NumFOCUS sponsored project for data analysis and manipulation [2]. In this project, data were imported from the data file to the DataFrame structure by using pandas.

1. NLTK

The Natual Language ToolKit (NLTK) [3] is a suite of python libraries for natural language processing to support research and teaching. In this project, it was used for tokenization, stop words removal, pos tagging, stemming and lemmatization in text pre-processing.

1. Sklearn

Scikit-learn (sklearn) [4] is a free software machine learning library for python. It provides efficient tools for machine learning and statistical modeling. In this project, it was used for supporting feature extraction, classifier models, and evaluation metrics.

1. Pytorch

It is an open-source machine learning library developed by Facebook’s AI Research Lab based on the Torch library [5]. It provides deep learning support by its two high-level features: Tensor computing with GPU support and deep neural networks built on a type-based automatic differentiation system. In this project, model 2 was built with PyTorch.

1. Hugging Face Transformer

The hugging face transformer package is a python library providing a pre-trained model for state-of-the-art natural language processing [6]. In this project, the DistilBERT model was used in our model 3.

Following techniques are used in our approaches:

1. N-gram

An n-gram is a contiguous sequence of n items from a given sample of a sentence or text. It is called unigram for n = 1, bigram for n = 2, and trigram for n = 3. For example, for a sentence: “I go to school by bus.”, unigram = {“I”, ”go”, ”to”, ”school”, ”by”, ”bus”}, bigram = {“I go”, ”go to”, ”to school”, ”school by”, ”by bus”}, trigram = {“I go to”, ”go to school”, ”to school by”, ”school by bus”}. Theoretically, higher value of n requires more running time and gives higher accuracy result in natural language processing. In this project, the accuracy of the result from unigram to trigram are compared in result section.

1. Term frequency-inverse document frequency (TF-IDF)

This is a weighting scheme calculated by term frequency and inversed document frequency. The term frequency refers to the frequency of a term that occurs within a document. Document frequency refers to the frequency of a term that occurs in a document across the entire collection. In TF-IDF calculation, the document frequency is used inversely. The formula for the term t in document d shown as below:

, where

1. Complement Naïve Bayes

Complement Naïve Bayes [7] was introduced in 2003 to solve the problems of the Naïve Bayes classifier caused by its systemic issue and assumption. The systemic issue is that when the training data set is imbalanced, Naïve Bayes selects poor weights for the decision boundary. For the assumption, it assumes that features are independent. In this project, Complement Naïve Bayes is used for fast and easy implementation with the better result as our project baseline.

1. Logistic Regression

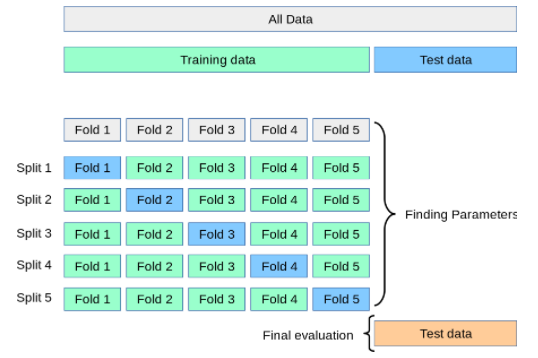
Logistic regression is a statistical model for binary classification. There are two options to use a binary classifier in multi-class classification: One-Vs-Rest and One-Vs-One. In this project, the logistic regression classifier from sklearn with default setting is used. By default, it uses One-Vs-Rest approach [4].

1. BERT

Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based machine learning technique developed by Google [8]. The key innovative technique of BERT is applying bidirectional training of Transformer instead of single direction, either from left to right or from right to left. The original English-language BERT has two pre-trained models: BERT-BASE and BERT-LARGE. In this project, DistilBERT [9] is used for faster and lighter development and evaluation.

1. K-Fold Cross-Validation

It is a statistical method for estimating the skill of machine learning models. In which, k is the number of fold. For k=5, it is called 5-fold cross-validation, for k = 10, it becomes 10-fold cross-validation etc. First, all data are split into training and testing data. The testing data is used for final evaluation. The train data are further split into k-fold, say 5-fold as an example. In k-fold cross-validation, k-1 folds are used for training and the remaining fold for validation. The training and validation process is repeated k times as fig. 2 below [4].



**Fig.2 5-fold cross-validation from sklearn**

1. Accuracy, Precision, Recall, and F1

These are the most common measurements for evaluation. The notations and formula are shown as below:

|  |  |
| --- | --- |
| Notations | Descriptions |
| TP | True positive |
| FP | False-positive |
| TN | True negative |
| FN | False-negative |
|  | True positive of the t th label on a test |
|  | False-positive of the t th label on a test |
|  | True negative of the t th label on a test |
|  | False-negative of the t th label on a test |
| S | Label set of all samples |
| Q | The number of predicted labels on each text |

**Table.1 Notations table**

1. Micro-F1, Macro-F1

In multi-label classification, an evaluation method for single-label classification are not suitable to evaluate the result of multi-label classification. Therefore, other evaluation metrics like Micro-F1 and Macro-F1 are introduced. Their formulas are shown as below:

, where ,

, where ,

**Implementation**

In this implementation section, the prerequisite, logical flow, and considerations for each model are included:

1. Shallow Learning – Model 1

*Prerequisite*

Required library: pandas, nltk, scikit-learn

Required download from NLTK: stopwords, punkt, averaged\_perceptron\_tagger, wordnet.

*Logical flow*

* 1. Read data from the file
  2. Text preprocessing [10] including tokenization, stop word removal, capitalization, stemming, and lemmatization.
  3. TF-IDF feature extraction
  4. Data manipulation: Preparation of training and testing data
  5. Train classifier and evaluate with 5-fold cross-validation
  6. Predict for the test data and evaluate result with Marco-f1, micro-f1, and accuracy

*Consideration*

As the baseline model of the project, we were tried to keep it as simple as possible. Therefore, word-embedding methods such as word2vec and GloVe were not used in this model. Due to the properties of news titles such as short and well checked, some procedures of text preprocessing mentioned in the research paper [10] were not implemented. The procedures not implemented include Slang and abbreviations, and Spelling Correction.

1. Deep Learning – Model 2

*Prerequisite*

Required library: pandas, PyTorch, scikit-learn

*Logical flow*

* 1. Read data from the file
  2. Encoding label from string to integer for tensor computation
  3. Prepare dataset
  4. Load dataset to torch data loader and convert to torch tensor
  5. Data manipulation: Preparation of training and testing data
  6. Train the nn model with training data
  7. Predict for the test data and evaluate result with accuracy

*Consideration*

The purpose of implementing this model was to compare between shallow learning model and the deep learning model in terms of its performance and easiness of implementation. In the PyTorch tutorial [11], it is a text classification for AG\_NEWS data set to 4 pre-defined classes. It is quite similar to the goal of our project. We tried to make the minimal change from its original program in the tutorial, especially on the defined nn model, so that we can also compare the result from different datasets avoiding misconfiguration and programming mistakes.

1. State-of-the-art – Model 3

*Prerequisite*

Required library: numpy, pandas, PyTorch, transformer, scikit-learn

*Logical flow*

* 1. Load distilbert model, tokenizer, and weight
  2. Import data from data files
  3. Find the max length of tokenized text and padding for the rest
  4. Mask the text data and convert it to tensor
  5. Extract features from the last hidden states
  6. Data manipulation: Preparation of training and testing data
  7. Train classifier and evaluate with 5-fold cross-validation
  8. Predict for the test data and evaluate result with Marco-f1, micro-f1, and accuracy

*Consideration*

The purpose of implementing this model was trying to catch up with the state-of-the-art approach of text classification. As same as the other 2 models, we tried to keep it simple and easy to read so that it can be compared easily. To evaluate this model, 5-fold cross-validation, Marco-f1 and micro-f1 were also used.

**Data**

The data file consists of 5000 samples with 2 parts using "######" as the separator. The first part is the news title, and the second part is the assigned class label. Figures from the data file are shown below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Title | Label |  | Frequency | Number of title count |
| Count | 5000 | 5000 |  | 5 | 1 |
| Unique | 4903 | 7 |  | 4 | 0 |
| **Table 2. Data set unique count** | | |  | 3 | 2 |
|  |  |  |  | 2 | 89 |
|  |  |  |  | 1 | 4811 |
|  |  |  |  | **Table 3. Title Frequency** | |

|  |  |  |
| --- | --- | --- |
| Frequency | Title | Label |
| 5 | pop quiz: do you remember the week that was? | entertainment |
| 3 | britney is back, but can she compete with gaga? | entertainment |
| 3 | multi-state accord would reduce tobacco sales to minors | health |

**Table 4. Top Frequency Titles**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | sport | business | entertainment | us | world | sci\_tech | health |
| Count | 1270 | 955 | 835 | 688 | 508 | 459 | 285 |

**Table 5. Class Labels Distribution**

**Results and Observations**

The evaluation metrics of the three models shown as below:

**Table 6. Evaluation Metrics**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | 5-Fold Cross-Validation / Epoch Accuracy | Standard deviation | Micro-F1 | Macro-F1 | Accuracy on Test |
| 1 | 0.7205 | 0.0092 | 0.7440 | 0.7020 | 0.7440 |
| 2 | 0.7059 | NA | NA | NA | 0.7080 |
| 3 | 0.7685 | 0.0009 | 0.7870 | 0.7698 | 0.7870 |

Model 3 has the highest accuracy around 76~78% on training and testing data. It also has the highest Micro—F1 and Micro—F1 scores.

The comparison of ngram range:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ngram\_range | 5-Fold Cross-Validation Accuracy | Standard deviation | Micro-F1 | Macro-F1 | Accuracy on Test |
| (1,1) | 0.7205 | 0.0092 | 0.7440 | 0.7020 | 0.7440 |
| (1,2) | 0.7256 | 0.0133 | 0.7260 | 0.6814 | 0.7260 |
| (1,3) | 0.7228 | 0.0162 | 0.7120 | 0.6741 | 0.7120 |
| (1,10) | 0.7019 | 0.0054 | 0.7080 | 0.6688 | 0.7080 |
| (2,2) | 0.3408 | 0.0191 | 0.3830 | 0.4244 | 0.3830 |
| (2,3) | 0.3085 | 0.0058 | 0.3010 | 0.3519 | 0.3010 |
| (3,3) | 0.1185 | 0.0111 | 0.1180 | 0.1196 | 0.1180 |

**Table 7. Ngram Comparison**

The ngram\_range take two integer number as input for the lower and upper boundary of the range. For example, (1, 2) means unigrams and bigrams are used, (1,1) means only unigrams, (2,2) means only bigrams. All results in Table.7 were contained from Model 1. It is observed that the results were close from (1,1) to (1,3) and slightly dropped in (1,10). There was a significant drop in accuracy when only using bigrams or trigrams.

**Discussions**

It is no surprise that the state-of-the-act model achieved the highest performance among the three models. However, the deep learning model's performance was slightly lower than the shallow learning model. The original tutorial [11], uses the AG news dataset and achieves 90% accuracy. There are two possible reasons for the disappointment of model 2:

1. Data Size

The AG's news topic classification dataset was constructed and used as a benchmark of the research paper [13]. It contains 120,000 training samples and 7,600 testing samples. The data size is around 25 times the dataset used in this project which contains 5000 samples. This may explain the accuracy difference in performance.

1. Pre-processing

Comparing model 1 and model 2, we did tokenization, stop word removal, capitalization, stemming and lemmatization using NLTK, however, the only capitalization had done on Model 2. As mentioned before, we tried to keep minimal change on the code so that we can compare the result with the tutorial, this may be one of the reasons that caused the difference in performance between model 1 and model 2.

For the observation on ngram, the performance was dropped significantly without using unigrams. It may be due to the reduction of tokens. For the example in the previous section, “I go to school by bus.”, there are 6 tokens in unigrams, 5 tokens in bigrams, and 4 tokens in unigrams. The number of tokens = number of words in the sentence – n + 1. The impact of the reduction is great in a short text like a news title. This may cause a significant drop in performance.

**Conclusion**

In this project, we have explored the techniques used from shallow learning and deep learning to state-of-the-art and built 3 models for news title classification. All models have above 70% accuracy. To further improve our models, there are few things we could do in the future:

1. Text Pre-processing

In this project, NLTK was used in pre-processing. However, NLTK is developed for research and teaching purposes that it uses the rule-based approach for tokenization and pos tagging. Instead of NLTK, spaCy [14] is designed to use in a production environment with a pre-trained model.

1. Pytorch Model

The PyTorch nn model used is not custom-made for this project. We could try to build one, especially for this task to see if the performance can be improved.

1. Bert fine-tuning

Due to the limited time, BERT-BASE and BERT-LARGE were not used in this project. We can try to use it later for further studying.

1. GNN-Based Model

A graph attention network-based model, MAGNET [15], was introduced in 2020. It is worth for study and try it in the future.

**Reference**

[1] Qian Li, Hao Peng, Jianxin Li, Member, IEEE Congying Xia, Renyu Yang, Member, IEEE Lichao Sun, Philip S. Yu, Fellow, IEEE and Lifang He, Member, IEEE. *A Survey on Text Classification: From Shallow to Deep Learning* [online] Available at: https://arxiv.org/pdf/2008.00364.pdf [Accessed 1 May.2021].

[2] pandas.pydata.org *Pandas document.* [online] Available at: https://pandas.pydata.org/docs/ [Accessed 1 May.2021].

[3] nltk.org *NLTK document*. [online] Available at: https://www.nltk.org/index.html [Accessed 1 May.2021].

[4] scikit-learn.org *Sklearn API Reference* [online] Available at: https://scikit-learn.org/stable/modules/classes.html [Accessed 1 May.2021].

[5] pytorch.org. *Pytorch document*. [online] Available at: https://pytorch.org/docs/stable/index.html [Accessed 1 May.2021]

[6] huggingface.co *HuggingFace transformer document* [online] Available at: https://huggingface.co/transformers/ [Accessed 1 May.2021].

[7] Jason D. M. Rennie, Lawrence Shih, Jaime Teevan, David R. Karger. *Tackling the Poor Assumptions of Naive Bayes Text Classifiers* [online] Available at: https://www.aaai.org/Papers/ICML/2003/ICML03-081.pdf [Accessed 1 May.2021].

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* [online] Available at: https://arxiv.org/abs/1810.04805v2 [Accessed 1 May.2021].

[9] Victor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper, and lighter* [online] Available at: https://arxiv.org/abs/1910.01108 [Accessed 1 May.2021].

[10] Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura

Barnes and Donald Brown. *Text Classification Algorithms: A Survey*. [online] Available at:

https://arxiv.org/abs/1904.08067 [Accessed 1 May. 2021].

[11] pytorch.org. *Tutorial: Text classification with the torchtext library* [online] Available at:

https://pytorch.org/tutorials/beginner/text\_sentiment\_ngrams\_tutorial.html

[Accessed 1 May. 2021].

[12] Jay Alammar. *A Visual Guide to Using BERT for the First Time* [online] Available at:

https://jalammar.github.io/a-visual-guide-to-using-bert-for-the-first-time [Accessed 1 May. 2021].

[13] Xiang Zhang, Junbo Zhao, Yann LeCun. *Character-level Convolutional Networks for Text Classification* [online] Available at: https://arxiv.org/abs/1509.01626 [Accessed 1 May. 2021]

[14] spacy.io *API* [online] Available at: https://spacy.io/api [Accessed 1 May. 2021]

[15] Ankit Pal, Muru Selvakumar, Malaikannan Sankarasubbu *Multi-Label Text Classification using Attention-based Graph Neural Network* [online] Available at: https://arxiv.org/abs/2003.11644 [Accessed 1 May. 2021]