Classification of French Dialects Based on Weka

Yimin Zhao, Yunzhu Liu, Yang Xiao,

Lingfeng Ren, Chenghao Zhao, Junhong Lou

Southwest Jiaotong University School

Abstract

This paper describes a dialect classification system for French speaking countries. The purpose of the system is to judge which French dialect the input sentence belongs to. SketchEngine was used to collect corpora from Six French speaking countries. Python pre-processing was used for deleting meaningless content like html tags, Waikato Environment for Knowledge Analysis (Weka) was utilized as a data analysis tool for filtering and classifying, for example, tokenisation, stemmer and stop-word removal. After several choices of suitable filters and classifiers, adjusting the parameters in them, and validating them in cross-validation, we finally reached 0.768 in F-Measure by applying SMO algorithm.

1 Introduction

The dialect classification system for French speaking countries described by this paper is a typical text classification project of machine learning. French is one of the major languages in the world. At the beginning of the 21st century, more than 25 countries set French as the official language(Posner, 2019). At present, 100 million people all over the world regard French as their first language. French has many dialects in France and other parts of the world. The French generally take the "Metropolitan French" as the standard language, some people in the south of France also use the "Meridional French" influenced by the oak language. The French dialects in Europe include Belgian French, Swiss French, Aostan French of Italy and so on. Besides, in Canada, the main French dialects are Quebec French and Acadian French(a20, 2021).

It is of great significance to study the characteristics of different dialects or variants in French speaking countries. This can enrich the current French corpus and study the differences of French dialects in different countries. The goal of the project is to judge the source country of the input French sentences and improve the accuracy of the classification as much as possible. More importantly, through practice, we can deeply and intuitively understand the data mining, learn more theories and technologies during the experiment, and experience the whole process of using machine learning to complete text classification. In this project, we achieved dialect classification through python pre-processing, Weka filters and different classifiers based on the corpus collected from Sketch Engine.

2 Data

The tool used in the data collection is Sketch Engine, which is a very popular corpus collection and management website. Considering that datasets may be optimised as the research progresses, one of the most prominent advantages of the Sketch Engine is that it supports researchers in creating a corpus from scratch and receiving external data files (Alfaifi and Atwell, 2016), allowing to build an initial corpus and save it before work begins, and if the results of the classification are not satisfactory, the initial dataset can be uploaded into the Sketch Engine for optimisation. Initially, five words from the University of Leeds' seed corpus that were more frequent but not related to each other were selected , but after guidance from the teacher, the selected countries were considered , most of which are well-off and have a developed tourism industry, so tourism was chosen as the theme, with the five seed words being tourism, food, landscape, architecture and customs. This ensures that the content of the corpus is all about tourism and life, so that it is easier to distinguish between different national dialects on the same

topic. An important tool used in the collection of the corpus was WebBootCat, where team members ensured that the corpus was sourced from a specific country by restricting the domain names in sitelist.

Twitter is great as a corpus source for dialects, and as a social platform it allows people to freely use dialects to express themselves(Alfaifi and Atwell, 2016). It is possible to access the corpus by creating a program to access the Twitter API, limiting the user to a specific country to obtain high quality data(Alshutayri and Atwell, 2017). However, the amount of work involved in writing the program was too much, so this approach is abandoned.

After downloading the web pages through WebBootCat, the pages were read through artificially to ensure that all content is travel-related, similar to how other researchers manually checked the content and copied and pasted it during the dataset collection (Al-Sulaiti et al., 2016). During the filtering process, we removed almost all PDFs as its large percentage in the whole corpus. In a similar vein, very short pages or pages with a huge number of images were also deleted, as such data is meaningless. Advertisements from multinational companies are also removed, since companies are likely to produce one copy of the advertisement in the same language when placing it in different regions, without taking into account the dialect. In order to ensure a balanced dataset, the final number of data sets collected by each individual was between 5w and 7w.

3 Method

3.1 Python Preprocessing

Python script is written to delete the html label and transfer six text file to .arff file ready to be processed by Weka.

First, the script read the six files one by one and stored the string into a list. For each instance of the list, the script deleted the html tags, punctuation, number and emoji symbol due to their meaninglessness.

We have encountered some problems in processing text files. Weka couldn't recognize file with punctuation double quote " in the string. So all double quotes in the text were removed. In addition to html tags, all punctuation, emoticons and numbers were also deleted because they did not have benefits for classification. Some processed instances were too small, which made the matrix sparse and affected the final result. After several attempts, the string with length less than 100 was decided to be abandoned.

3.2 Weka Preprocessing

Weka is used for further data processing. StringToWordVector is an crucial filter for converting string attributes to numerical attributes for classification. In this filter, TF-IDF is used to extract text features and vectorize data to indicates how important the word is in the document collection(Alshutayri et al., 2016).

Tokenisation is also employed, which is the segmentation of text into smaller parts called tokens, e.g. words, phrases, n-grams, sentences, etc. This process can be helpful to improve the accuracy in the text classification(Merlini and Rossini, 2021). Weka software offers different tokenizers, for this classification task, NGramTokenizer was chosen with arguments of unigram, bigram and trigram.

What's more, stemming operation was also performed, which converts all word variants into root words, so that a word has only one representation, which improves the performance of the system by reducing the number of different words and increasing the frequency of some of them(Merlini and Rossini, 2021). In fact, the previous n-gram extraction also performed some simple stemming operations. LovinsStemmer in Weka was selected for stemming, which was larger, faster and more efficient than the porter algorithm (Snowball).

In order to reduce the volume of searches and improve the efficiency and results of the classification, stop words were removed during preprocessing. Stop words are words that have no real meaning, but are very common. In general, about one-fifth to one-third of a text is stop words(Merlini and Rossini, 2021). This is generally achieved by using a pre-populated deactivation list, which is supported in the weka software by loading a custom list from an external file. On top of that, the method without the stop words was also adopted to compared with them.

3.3 Model

After processing the data, the experiment entered the training model. The experiment selected several algorithms to model, train through the same training set, test with the same test set, and get the results containing various parameters. Finally,

the advantages and disadvantages of distinct algorithms and models would be compared and analyzed (Witten, 2004).

First, considering that linear regression cannot handle multi-valued nominal classes, this experiment carefully selects algorithms that are more suitable for text analysis and judgment for comparison. Following are the two algorithms that are mainly used to build the model

The original plan of this experiment was to use Naive Bayes multinomial text. This algorithm should be more suitable for the text analysis of this project. However, the mixed matrix parameters have abnormal distribution and a large number of invalid parameters after it was appliced(A. and H., 1994). Due to the unbalanced proportion of multi valued nominal classes in multi-national dialects and the inevitable error in text analysis and dialect judgment, it was abandoned and replaced with a more concise and more inclusive Naive Bayes.

Naive Bayes is a simple method to construct classifier. The classifier model will assign class labels represented by eigenvalues to the problem instances, and the class labels are taken from a finite set. It is not a single algorithm for training this classifier, but a series of algorithms based on the same principle: all Naive Bayesian classifiers assume that each feature of the sample is not related to other features(E., 2020).

Sequential Minimal Optimization (SMO) is an algorithm used to solve the optimization problems generated in the training process of support vector machine. SMO algorithm is an iterative optimization algorithm. In each iteration step, the algorithm first selects two vectors to be updated, then calculates their error terms respectively, and calculates the threshold according to the above results. Finally, the offset is calculated according to the definition of SVM. For the error term, it can be adjusted according to the increment without recalculating each time. For the analysis of a large number of texts, it can be dynamically planned to reduce the analysis time, optimize the process and increase the accuracy.

4 Results

This system is evaluated using the cross-validation method. We found that the accuracy of validations on training set or segmenting the training set were too high(Alshutayri et al., 2016). Furthermore, it is not scientific to use training data set(hH1sG0n3, 2020). Indicating it's unscientific. The basic idea of cross-validation is to divide the original data into groups, one part of which is used as the training set training model and the other part as the test set evaluation model. There are two major advantages to using this method(Celisse, 2010). The first of all, cross-validation can be used to evaluate the prediction performance of the model, especially the performance of the trained model on new data, which can reduce the degree of over-fitting to a certain extent. The second point is that we can get as much useful information as possible from limited data.

The three sets of experiments used some uniform parameter settings that are recognised to improve the accuracy of text classification, namely: IDFTransform=True, TFTransform=True, normalizeDocLength=True, outputWordCounts=True.

The first set of experiments (1) was designed to verify the accuracy of SMO and NaiveBayes when using different NGramokenizers, UniGram, BiGram and TriGram respectively. The second set (2) was designed to verify the effect of stemmer. The results from the first set show that SMO works best with UniGram, so this set also fixes the following parameters: tokenizer=UniGramTokenizer, classifier=SMO. It provides the accuracy with and without the use of LovinsStemmer. The third set (3) was designed to verify the effect of using stopwordsHandler on accuracy. The results from the first two show that SMO works best with UniGram and LovinsStemmer, so the following parameters were also fixed : stemmer=LovinsStemmer, tokenizer=UniGramTokenizer, classifier=SMO. The experiments were tested to derive the accuracy of the test with and without stopwordsHandler using WordsFromFile respectively, where the stopwords list is from online. resources(Bouge). For the last set of experiments (4) we used the SMO algorithm, the two cases being the case where all parameters in the StringToWordVector filter are optimal and the case where they are not adjusted.

Results of twelve runs of different classifiers are shown in1234.

5 Conclusion

The results is acquired from confusion matrices, which are tables that summarize classification and

Algorithm	Accuracy	Precision	Recall	F-Measure
SMO(UniGram)	76.8	77	76.8	76.8
SMO(BiGram)	70.6	70.8	70.6	70.5
SMO(TriGram)	66	68.2	66	66.4
Naive Bayes(UniGram)	59.5	64.2	59.5	59.2
Naive Bayes(BiGram)	48.6	56.2	48.6	48.4
Naive Bayes(TriGram)	30	50.7	34	33.3

Figure 1: Algorithms

SMO(UniGram)				
Stemmer	Accuracy	Precision	Recall	F-Measure
Null	76.8	77	76.8	76.8
LovinsStemmer	77.8	77.9	77.8	77.7

Figure 2: SMO(UniGram with stemmer)

StopWordHandler	Accuracy	Precision	Recall	F-Measure
Null	76.7	76.9	76.8	76.8
WordsFromFile	76.8	77	76.8	76.8

Figure 3: SMO(UniGram with StopWordHandler)

	Accuracy	Precision	Recall	F-Measure
SMO(best arguemrnt)	76.7	76.9	76.8	76.8
SMO(StringToWordVector)	71	71.2	71.0	71.0

Figure 4: SMO(Null compare best argument)

segmentation performance and are used in scene analysis to evaluate the predictions of classification models. A common case is the binary confusion matrix, which is operated to represent the positive and negative classes of some binary classification problems. The four cells of the matrix are True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN) as shown below (E., 2020).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Recall = \frac{TP}{TP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

This paper discusses the classification of French dialects in six different regions, France, Luxembourg, Swiss, Belgian, Italian, and Canadian. The French corpus from six countries were created by the group using Sketch Engine, Which was classified appling the SMO algorithm and the Naive Bayes.

Firstly, according to the Figure 1, experiments show that the SMO algorithm achieves higher accuracy than the Naive Bayes method. In the SMO algorithm, SMO (UniGram) obtained 76.8% accuracy, 77% precision, 76.8% recall, and 76.8% F-Measure. The Naive Bayes method (UniGram) obtained 59.5% accuracy, 64.2% precision, 59.5% recall, and 59.2% F-Measure.

Secondly, on the premise of obtaining the best accuracy of the SMO (Unigram) algorithm, we compared the two cases with and without the stemmer which shown in the Figure 2. The results show that using LovinsStemmer gets 77.8% accuracy, 77.9% precision, 77.8% recall, 77.7% F-Measure, which is better than when the stemmer is Null.

Similarly, we compared the two cases with and without the stopword handler. According to the Figure 3, the results show that the difference of the SMO (Unigram) algorithm is very small between Null and WordFromFile.

Fourthly, based on Figure 4, by comparing the accuracy, precision, recall and F-Measure of the SMO (best argument) algorithm and the SMO (StringToWorldVector) algorithm, we found that using the stemmer and other preprocessing tools in StringToVector first in the experiment can indeed improve our dialect classification accuracy.

After these experiments, the quality of the data was found to be improved. The six country datasets were not consistent in length after the processing of Python script. Although all data lengths were forced to be the same, the confusion matrix indicates that the density of each data item was not consistent. In the next experiment, data balancing will be considered from the beginning of data collection and other balancing techniques will be tried.

```
=== Confusion Matrix ===

   a   b   c   d   e   f   <-- classified as
 527  25  20  14  17  27 |   a = BE
  49 482  29  31  17  22 |   b = FR
  30  54 447  47  10  42 |   c = SW
  38  38  56 454  22  22 |   d = LU
  33  20   9  14 547   7 |   e = CA
  30  37  35  40   6 482 |   f = IT
```

References

2021. Varieties of french.

Holmes G. Donkin A. and Witten I. H. 1994. Weka: a machine learning workbench. *Proceedings of ANZIIS '94 - Australian New Zealnd Intelligent Information Systems Conference*, pages 357–361.

Latifa Al-Sulaiti, Noorhan Abbas, Claire Brierley, Eric Atwell, and Ayman Alghamdi. 2016. Compilation of an arabic children's corpus.

Abdullah Alfaifi and Eric Atwell. 2016. Comparative evaluation of tools for arabic corpora search and analysis. *International Journal of Speech Technology*, 19(2):347–357.

AOO Alshutayri and Eric Atwell. 2017. Exploring twitter as a source of an arabic dialect corpus. *International Journal of Computational Linguistics (IJCL)*, 8(2):37–44.

Areej Alshutayri, Eric Atwell, Abdulrahman Alosaimy, James Dickins, Michael Ingleby, and Janet Watson. 2016. Arabic language weka-based dialect classifier for arabic automatic speech recognition transcripts. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pages 204–211.

Kevin Bouge. Download stop words - kevin bougé.

Arlot Celisse. 2010. A survey of cross-validation procedures for model selection. *Statistics Survey*, 4:40–79.

Tarmom T. Teahan W. Atwell E. 2020. Compression versus traditional machine learning classifiers to detect code-switching in varieties and dialects: Arabic as a case study. *Natural Language Engineering*, 26(6):663–676.

hH1sG0n3. 2020. machine learning - why is it wrong to train and test a model on the same dataset?

Donatella Merlini and Martina Rossini. 2021. Text categorization with weka: A survey. *Machine Learning with Applications*, 4:100033.

Rebecca Posner. 2019. *French Language — Origin, History, Grammar, Speakers*.

Snowball. The lovins stemming algorithm.

Frank E. Hall M. Trigg L. Holmes G. Ian H. Witten. 2004. Data mining in bioinformatics using weka. *Bioinformatics*, 20(15):2479–2481.