Similar Language Detection

Naveen B(140050034), Chanukya(140050043), Uday K(140050048),
Raghuveer U(140050050)
Guide: Ganesh Ramakrishnan
Mentors: Abhishek Sethi, Ajay A. Verma

IIT Bombay

29/04/2017

Outline

- Introduction
- Project Statement
- Classification into groups
- Classification within groups
- Feature Engineering
- Classifier Selection
- Tuning Hyper-parameters and Ensemble method
- Results
- Observations and Inferences
- Future Work and Improvements
- References and Data sets

Introduction

- Discriminating between similar languages is the whole purpose of the project. Discriminating between disparate languages is trivial in the sense that it has well known methods for solving.
- But the same methods used for disparate languages doesn't yield good results for discriminating between similar languages and this project address this problem.
- We have used a hierarchical method, we first divide all the languages into few groups and next we label the languages in each group.
- The first step is Word-Frequency method with tf-idf-qf weighting and second method uses SVM's.

Project Statement

- The training data consists of sentences labeled with 13 languages which are Bulgarian (bg), Macedonian (mk), Bosnian (bs), Croatian (hr), Serbian (sr), Czech (cz), Slovak (sk), Spanish (esar), Peninsular Spanish (eses), Brazilian Portuguese (ptbr), European Portuguese (ptpt), Indonesian (id), Malay (my) and a unknown language(xx),
- Each language has 18,000 sentences for training set and 1,000 sentences for testing.
- Labelling a query sentence with correct language is the problem.
- We divide the training set for each language into 16,000 and 2,000 sentences for training and validation data respectively
- We first convert all sentences from Uppercase to Lower case and remove all the special characters.

Classification into groups

- From the training data, for each language we first parse through all the sentences and find the frequencies of all the words present.
- Among all the words and their corresponding frequencies generated for each language, we choose the top N words based on their frequency.
- Define x^l for each language l as vector $\in R^N$, $x^l = 1000$-tuple of frequency of the chosen words.
- Now given an input statement s we define $x^l(s)$ for language l as follows
- $x^l(s) \in R^N$ first initialised to all zeros
- $x^l_i(s) = 1$ if the i^{th} word of the l^{th} language is present in the input sentence.
- Label for statement s is label(s) = $argmax_{l \in 1...13}$(dotproduct of $x^l(s)$, x^l).
- The intuition is clear, we choose top words from each language and if a top word is found in the query sentence, we add its frequency to the score and the language with maximum score is label.

Problem and tf-idf

- From the confusion matrix after first stage, it is clear it is not appropriate since it needs lot of groups to make group accuracy high and 'pt-Br' and 'eses' cannot be put to same group as 'eses' does not contain any 'pt-Br' while 'pt-Br' has lot of 'eses'
- Solution we thought of is to use tf-idf scoring i.e. the x^l for each language has log(1+1/fl)*log(1+ft) and $x^l(s)$ has f(t,q)*log(1+1/fl) where fl = fraction of languages which contains the word and ft = frequency of the word in the language and f(t,q) is frequency word in query (f(t,q) is normalised).
- The above idea worked and the intuition is the term log(1+1/fl) decrease the weight for words occurring in all languages. We choose N = 5000 clear from graphs and confusion matrix and group accuracy is greater than 99 percent.

Results and groups

- The languages bg, cz, mk, sk are already classified with accuracy greater than 99 percent and the other languages has less accuracy between 70-90 percent and we group them as follows bs,hr,sr esar,eses id,my ptbr,ptbt.
- Note that languages are classified into corresponding groups with accuracy close to 100 percent which is we intended.
- Now each group classification is a different problem and we intend to use ensemble methods with SVMs. For each group we have ensemble of 5 SVMs.

Classification within groups

- Our basic idea for feature modeling for the SVM was through word and character n-grams. Character 1,2-grams when enumerated are very less, so they don't make good features. We chose character n-grams with n $\in$ 3,4,5,6,7,... and word n-grams with n $\in$ 1,2,3,.... We needed to model an efficient feature vector from the n-grams with an appropriate dimension.
- Since character and word n-grams capture different properties, we felt instead of clubbing them in the same feature vector of an SVM, training an ensemble of SVMs over the pure n-gram features is better.
- SVMs for different n-grams as listed above were trained and the overall ensemble accuracy was noted for incremental subsets of 3,4,5,6,7... for characters and 1,2,3,... for words. From the plots we observe that the optimal features would be character 3,4,5-grams and word 1,2-grams, a partial reason also being overfitting for character n-grams with n $\geqslant$ 4

Feature Engineering

- Even after choosing the features as above, the number of features are very high and need to be reduced. For capturing the important features, we score the features using tf-idf scoring with

$$tfidf(t, D) = tf(t, D) \times log(^{N}/_{df(t,D)}) \tag{1}$$

- We rank the features according to the tf-idf scores and pick the top n features. n is chosen by plotting the validation accuracies. The accuracies get saturated for n $\geqslant$ 20000
- Given a query, the feature vector is a boolean vector that captures the presence of the n-ranked features. We represent the feature vector in a sparse way as a dictionary

Classifier Selection

- We chose the weak learners as logistic regression, and support vector classifications with the error norms and the regularizations varying between L1 and L2. By plotting validation accuracies, we decided to use support vector classifier with L1 norm and L2 regularization.

Classifier	**accuracy**(%)
L2-regularized logistic regression	86
L2-regularized L2-loss support vector classification	88
L2-regularized L1-loss support vector classification	88
L1-regularized L2-loss support vector classification	85
L1-regularized logistic regression	77

- Since SVM uses a margin to classify the languages and considering the fact that the languages in our case are similar, SVM seemed more intuitive to be used.

Tuning Hyper-parameters and Ensemble method

- We tune the hyper-parameters(no. of features and the constant C of SVM) for the classifiers using validation accuracies

C	**accuracy**(%)	**No.of features**	**accuracy**(%)
1	87	5000	86.7
0.1	88	10000	88.07
0.01	88	20000	88.4
0.001	85	30000	88.42

- Finally we arrived on the following setup with C=0.01:

Language group	**No.of features**	**Char n-grams**	**Word n-grams**
bs, hr ,sr	20000	3,4,5	1,2
eses, esar	30000	3,4,5	1,2
id, my	20000	3,4,5	1,2
ptbr, ptbt	30000	3,4,5	1,2

- We tested majority voting and mean confidence for the ensemble method and through validation accuracies of 87.47 and 88.48 respectively, we chose mean confidence

Results

- The results for the language classes are as follows

Language group	Validation Accuracy(%)	Test Accuracy(%)
bs, hr ,sr	88.48	89.6
eses, esar	89.55	90.1
id, my	98.82	98.6
ptbr, ptbt	91.72	92.4
bg	99.4	99.6
cz	99.4	99.3
mk	99	99.8
sk	99.5	99.6

Observations and Inferences

- Test accuracy is calculated for 1000 test examples for each language completely not known to the trained model
- The validation and test accuracies are in consensus as in the above table
- Although the languages bg,mk can be grouped and similarly cz,sk we didn't group them as the initial classification had very high validation accuracy.
- For the group id, my after the initial classification itself the validation accuracy for each language was high as seen in the confusion matrix. But we grouped them for a better accuracy and obtained an increase
- Since the grouping accuracy was very close to 100% we now compare the validation accuracies of both the stages for each language as shown below

Observations and Inferences

Language	Stage-1(%)	Stage-2(%)
bs	70	84.5
hr	83	88.8
sr	90	95.6
eses	90	88.8
esar	75	91.7
id	93	98.5
my	96	98.7
ptbr	87	91.1
ptbt	87	94.1
bg	99.4	99.4
cz	99.4	99.4
mk	99	99
sk	99.5	99.5

Future Work and Improvements

- The accuracy of eses has fallen due to grouping with esar but the grouping is necessary so as to increase the accuracy of esar which actually increased drastically. More state of art ensemble methods and classifiers can be tested and also the intial grouping can be done faster with smaller bag of words by better tf-idf-qf scoring which we couldn't do due to the limited scope of this project
- We need to appreciate the fact that the accuracy of bs and esar have drastically increased due to grouping. Though the accuracy of bs increased very well, the overall accuracy for the language group bs, hr, sr is still low and it should be improved by different techniques.

Contributions

- Raghuveer U - Coding for Language grouping, SVM modeling
- Chanukya G - Coding for Language grouping, Feature Engineering for SVM
- Uday K - Implementation Design and Analysis in both stages, Usage of SVM
- Naveen B - Tuning parameters from validation accuracies

Thank You!

References and Data Sets

- We will be working on the data set that was provided in the competition. There are 18000 training samples for each language `https://github.com/Simdiva/DSL-Task/tree/master/data/DSLCC-v2.0`.
- Reference : `http://cs229.stanford.edu/proj2015/335_report.pdf`