COMP 551 Project 2: Language Classification

David Garfinkle
260515841
david.garfinkle "at" mail.mcgill.ca

Parker King-Fournier
260556983
parker.king-fournier "at " mail.mcgill.ca

Kabir Rajput
withdrawn

***Abstract*—Language identification has many far-reaching applications, especially in the textual domain. This report evaluates two methodologies for identifying five possible languages of text: Slovak, French, Spanish, German, and Polish. Specifically, the construction, implementation and performance of Naïve Bayes and Decision Tree Classifiers will be discussed.**

***Index Terms*—decision trees, feature construction, feature selection, language classification, machine learning, most likely estimation, naive bayes, natural language processing**

I. Introduction

The classification of data into language classes is a well known problem in the world of machine learning. There are many applications in both textual and audio speech applications, such as document retrieval or real-time interpretation. The goal of this study is to focus on the textual domain by classifying scrambled sentences (like Boggle or Scrabble) into either Slovak, French, Spanish or Polish using techniques common to machine learning. For this task, a Naive Bayes classifier and a Decision Tree were chosen.

II. Related Work

The treatment of completely unordered characters as input for a language classifier is relatively untouched in the literature. However, a few reviews of the Language Identification task are still notable. N-grams and word frequency are reviewed by Grothe et al., as well as inverse-document weights which review the idea of adjusting the probability of certain features within a class given their total occurrence in the whole training set. Grothe et al., discuss the limitations of a variety of models applied to the textual language identification task. To be discussed later, the preservation of words in training the classifiers was unnecessary as the test set was comprised of space separated characters. As a result, any method relying on more than one character could be eliminated. It is apparent that methods relying on characters and language class would be more useful. In Reference to *A Comparative Study on Language Identification Models*, all three approaches were insufficient for the purpose of this study. However, the Frequent-Word model inspired similar Frequent-Character and Frequent-Language models that were adopted in the form distributions.

Further approaches to language classification were discussed in *Reconsidering Language Identification for Written Language Resources* (Hughes et al. 2006). This article outlines aspects of language classification that are still "ripe for further investigation". To do this, Hughes et al. (2006) present common approaches and methods to language classification to provide a background for work that has already been completed in this area. Of particular note was the mention of *detection of the character encoding* , and *statistical methods* to classify text. The notion of character encoding, while not directly applicable to the study presented in this report, inspired thought about languages and their unique characters. Just as a unique encoding may indicate a certain language, it is possible that the presence of certain characters that are never, or rarely found in other languages, may indicate a certain language. An example illustrating this is $\tilde{n}$, which is found almost exclusively in Spanish.

The article supplied valuable information in regards to the shortcomings of modern approaches. Hughes et al. (2006) mention that Sparse or Impoverished Training Data remains an outstanding issue in language classification. This was noted due to the possibility that the training data, or subsets of it, supplied for this report may be of insufficient size (less than several thousand words). With a supplied training set of size 276,517, it is possible that this is true.

The independent research summarized in this section provided background information on methods, their pros and cons, and their applicability this study. Digesting this information established a starting point and bearing from which to start to formulate a solution.

III. Problem Representation

Both the training and test data for this study was collected by classmates for a previous project. Because of a lack of regulation in data collection the data supplied for this study was "dirty", requiring some preprocessing to enhance the data's characteristics.

The training data was given in the form of two comma-separated-values (csv) files referred to from now on as X_{Train} and Y_{Train}. X_{Train} contained a unique identification number for each data entry in the first column, and a string of characters in the second column. Y_{Train} contained the same unique identification number for each entry in the first column, but in the second column contained a number representing the language that the corresponding entry in the second column of X_{Train} was written in. To standardize representation and consolidate the data, a parser was written that, for each data point, created an utterance with the form:

Language	% Occurrence
Slovak	5.1
French	51.1
Spanish	25.3
German	13.4
Polish	5.1

Fig. 1. Percentage occurrence of each language. Calculated from X_{Train} and Y_{Train} and rounded to the nearest 0.1%

$$U_i = \text{[Identification number, (Text, Language)]} \quad .$$

This allowed a compact representation and allocated a place in memory for each data point which aided in further use of the data.

The nature of the test data further guided the preprocessing of data. Firstly, the test data ignored all positioning of characters and used uniform white space to separate them. To mitigate this effect against our actual word-separated training data, we removed all white spaces and focused solely on characters. Secondly, since the use of capital letters is at intuitively independent of any one language (though, depending on the corpora such information may be relevant), we converted all characters to lower case. These pre-processing steps were applied to every instance in our training data as well as test sets.

These important observations about the training and test data were useful in the process of feature selection and feature design. Since the test set contained only characters, it was apparent that the features chosen should be constructed based on attributes not dependent on words written in column two of X_{Train}.

An obvious feature of the data revealed itself to be the percentage occurrence of each language (Fig. 1). Although not a particularly clever representation of the data, this feature was chosen because of its potential use in deciding between languages of differing abundance. Figure 1 shows that the distribution of languages significantly skewed: if on were deciding whether a query is in exclusively French or Slovak it would be advantageous to know that, per the training data, it is about ten times more likely that a query is in French.

The remaining features selected were functions of the characters found in column two of X_{Train}, the first of which being character distributions. The probability, p_c, of each character, c, occurring was estimated using the Maximum Likelihood Estimate (MLE)

$$\hat{p}_c = \frac{\hat{c}}{N_T}, \tag{1}$$

where $\hat{c}$ is the number of occurrences of character c and N_T are the total number of characters in X_{Train}. This feature was chosen with hopes that the rarity of a letter may be used in determining how important that characters is to language classification: abundant characters are more likely to be found in every language, and less likely to be indicative of a language. The character frequencies can also be used to normalize data. The probability that a string, s, which contains characters c_1, ..., c_n with low abundances belongs to a give language, l, will be evaluated as a combination of the abundances of each c_i in language l. This would yield a very low "score" that results from being skewed by the underlying character distributions of the training set. Normalizing each frequency of c_i in language l by the overall frequency of each character eliminates an unnecessarily low probability being assigned to s and could potentially increase classification accuracy.

The second character-based feature strove to further characterize the abundance of characters over the whole training data set. This feature, q_c, the percentage of data points containing the character c was again estimated using the MLE

$$\hat{q}_c = \frac{\hat{U}_c}{N}, \tag{2}$$

where $\hat{U}_c$ is the number of data points containing the character c and N is the number of data points.

Knowing all values of p_c and q_c could be useful: if a character has a low value of p_c but a high q_c value, one might infer that the character is found in many languages, but is uncommon in all. Conversely, if a character had a high value of p_c and a low q_c value it could be conceivable that this character, which shows up abundantly in the utterances it is found but is otherwise rare, is indicative of a certain language classification.

Perhaps a more intuitive method for determining significance of characters in each language is to find $p_{c|l}$, the probability of character c occurring in language l. The distribution of characters in a language, denoted d_l, was calculated to create a distinct representation of each language in terms characters. Classifying s, a string of characters, using this metric then becomes a question of which language distribution best fits the distribution of characters found in s, denoted d_s . Alternatively this is denoted

$$L(s) = l^* \ s.t. \ S(d_{l^*}, d_s) = \max_{\forall l \in L} S(d_l, d_s), \tag{3}$$

where $L(s)$ is the language classification of string s and $S(d_l,d_s)$ is a function measuring similarity between distributions.

Lastly, based on the distributions d_l, $\forall l \in L$, sets were constructed of characters that were either never or only found in a certain language (referred to as Not-sets and Only-sets). These sets were constructed by in order to quickly narrow down the possible languages that a string s could be classified as.

Lastly, sets of characters were constructed that were either never or only found in a certain language. These sets were created by using the distributions d_l, $\forall l \in L$, to establish an alphabet, a_l, for a given language then using set theory to establish the sets. For example, the set of characters that are *only* found in Slovak would be

$$a_{Slovak} - \bigcup_{\forall l \in L \ except \ Slovak} (a_{Slovak} \cap a_l), \tag{4}$$

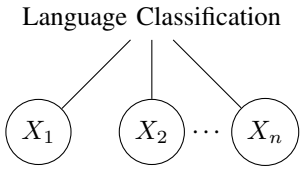

Fig. 2. A visual representation of a Naïve Bayes classifier. Each X_c represents the number of times a character, c, appears in the training data.

and the set of characters *not* found in Slovak would be

$$\bigcup_{\forall l \in L} (a_l - a_{Slovak}). \tag{5}$$

These sets were constructed to quickly narrow down the possible languages a string *s* could be classified as.

IV. Algorithm Selection and Implementation

As mentioned in the introduction, two classification techniques were used in this study. The first, linear classification algorithm used was a Naïve Bayes Classifier (Fig. 2) . The Naïve Bayes was chosen as a result of the knowledge gained from the feature selection described in the previous section. Due to the multiplicative properties of probabilities under the Naïve Bayes assumption any, and character, *c*, in a string, *s*, with a probability $p_{c|l'}$ = 0 will send the resulting value to 0, eliminating the possibility that *s* will be classified in *l'*. This characteristic allows the classifier to quickly narrow down the possible languages that a string, *s*, can be classified as. As mentioned in the previous section, the probability distribution for the Naïve Bayes Classifier was estimated from data using the method of Most Likely Estimation (MLE) as opposed to adhering to an arbitrary distribution. A Maximum Likelihood Estimator is shown to be a *consistent* estimator, meaning that the expected value of the estimator is the estimand in question and that, as sample size increases, the variance in the estimator tends towards 0. The size of X_{Train} and Y_{Train}, and the ability of the MLE method to accurately "learn" a hidden distribution from data were fundamental attributes that led to the use of MLE in this study.

A second, non-linear classification algorithm was implemented in the form of a Decision Tree. Before implementing the top-down recursion that would construct the decision tree it was necessary to find tests that would partition the total alphabet, a_{total} into respective language alphabets as efficiently as possible. For this purpose, the Not-sets and Only-sets were used to eliminate or confirm possible languages, respectively. After these tests were determined, a node of the tree was assigned the test that yielded the most information gain.

The implementation of the Decision Tree consisted the standard top down recursion as well a few implementation-specific features which will be listed to provide insight to those looking to implement the algorithm themselves. When first training the tree the algorithm arbitrarily chose a test given that one or more tests had the same information gain, resulting in whole branches of the tree being labeled with whatever test is the default tie-breaker. This phenomenon happened primarily when one or more test "split" the data into one list of size zero and one list of size *N*, for *N* data points. This partitioning of data yields no information gain, and as a result the algorithm was modified to detect and report this with a special value (-2). Nodes of the decision tree labelled with this value are seen as "dead ends" at which no more information can be extracted and a heuristic-guided guess must be made.

	Slovak	French	Spanish	German	Polish
Slovak	9709	932	267	294	49
French	659	104983	4710	2505	155
Spanish	913	7932	33302	3709	82
German	293	4254	739	24254	112
Polish	306	552	231	340	9931

Fig. 3. Confusion matrix

Both algorithm implementations were trained originally using the entirety of X_{Train} and Y_{Train} but were further validated to improve optimization. The parameters, or assumed distribution, of the Naïve Bayes Classifier was selected for by validating the algorithm both on subset and the entirety of X_{Train} and Y_{Train}. From this, the classifier's optimal parameter setting was the character distribution as approximated by the method of Most Likely Estimation. Similarly, the optimal depth of the Decision Tree was also found. Originally to be carried out via cross validation, the observation that the tree consisted of a small number (6) of tests determined that the maximum depth of 6 be chosen.

V. Testing and Validation

To increase accuracy in the classifiers, the data was split such that 80 percent was set aside for training, with the remaining 20 percent used for testing. Cross validation was carried out with this partitioning scheme. The accuracy, a_k, for each classifier, *k*, was calculated as

$$a_k = \frac{p}{p+n}, \tag{6}$$

where *p* and *n* are the number of positives and negatives, respectively. The results are summarized by a confusion matrix in Figure 4.

Unfortunately, towards the end of the project our third teammate withdrew from the class and we were unable to pick up on the rest of the work, the Decision Tree was never able to be fully implemented. The Decision Tree trains properly on the training data, but the classifier function remains unimplemented at the time of submission.

VI. Discussion

Both classifiers were chosen for their apparent advantages in classification, however the performance of these classifiers

Classifier	% Accuracy
Naïve Bayes	82
Decision Tree	-

Fig. 4. % Accuracy of classification methods, rounded to the nearest percent.

exposed disadvantages of each approach. Specifically, the Naïve Bayes Classifier was chosen for the fact that multiplication of 0-valued probabilities quickly rules out "incorrect" classifications. As mentioned above, this property carries a double edge, as the use of the method of Maximum Likelihood Estimation to train a Naïve Bayes Classifier is susceptible to learning "errors" in the data set on which it is trained. These "errors" often the result from an incomplete representation of the underlying distribution due to small samples size. For example, if training set X'_{Train} contained no examples of the use of the character *w* in French, a Naïve Bayes Classifier trained on X'_{Train} would never classify any string, *s*, containing *w* to be written in French. The consistency of the method of Most Likely Estimation implies that confidence in estimates increases as sample size does. While the training data given was not small, the distribution of languages caused some subsets to quite small. According to Fig. 1, 5.1% of training examples were in Slovak and another 5.1% in Polish. X_{Train} consisting of 276,517 examples implies that for Slovak and Polish there were approximately 7,277 data points with which to estimate the character distributions for each language. The low sample size of subsets of training data contributes significantly (in)accuracy of a Naïve Bayes Classifier. This can be avoided by applying Laplace smoothing to parameter estimates according to the formula

$$\hat{\theta}_{Laplace} = \frac{\hat{x} + 1}{N + 1}, \tag{7}$$

where $\hat{\theta}_{Laplace}$ is the smoothed estimand of θ, $\hat{x}$ is the relevant number of occurrences of the variable in question and *N* is the total number of elements. The ability of the Naïve Bayes Classifier to infer parameters dependent on unobserved latent data could be improved by the use of the Expectation Maximization (EM) algorithm. The EM algorithm alternates between creating a function for the expectation of the log-likelihood (E-step), using the current parameter estimates, and maximization (M-step), where parameters maximizing the expected log-likelihood found on the E-step are calculated. This re-estimation of parameter values diminishes the effect of hidden variables on the accuracy of the estimation.

A Naïve Bayes Classifier is elegant in its simplicity, however the simplicity comes with a price in the form of the strong assumption of independence between features. If this assumption is incorrect, the accuracy of the classifier can be severely affected. There is evidence to suggest that such an assumption does not hold in language, which can be intuitively understood by considering how the frequencies of *q* and *u* are correlated in the English language. Because *q* is frequently followed by *u* in most English words it is reasonable to conclude that in string $s = x_1, ..., x_n$

$$P(X_i = u) < P(X_i = u \mid X_{i-1} = q), \tag{8}$$

showing a dependence between characters in a given language.

To further test the independence of the features used one could carry out an independence test, such as χ^2, between each pair of features. Furthermore, the degree of independence between features could be incorporated in the Naïve Bayes Classifier by quantifying independence as a confidence score which would scale the classifiers resulting outputs accordingly.

As mentioned before, the Decision Tree implemented yielded no results. For the sake of the discussion section, this method will be evaluated theoretically, considering carefully the advantages and disadvantages of this approach as a single classifier, and in combination with the Naïve Bayes Classifier. The choice to implement a Decision Tree was inspired by there being only 5 language classifications. One would expect this to cause the resulting tree to be relatively small, and thus computationally inexpensive. Contrary to this insight, the tree exhibited less than optimal results. Despite the low complexity the tree, its small size limited its ability to accurately make decisions. Because of the limited number of tests used to partition the data-space the effectiveness of each test in partitioning the data is paramount.

The main limitation of the implementation of this Decision Tree was the quality of tests used. As mentioned above, the tests used sometimes split the data into sets of size 0 and size *N*, implying redundancy. The tests, namely the Not- and Only-sets, were limited in their ability to evenly divide the data-space. Tests associated with the Only-sets could potentially assign a classification in one decision, but in practice were rarely evaluated as true and therefor often redundant. Each test associated with a Not-sets was able to eliminate at best 1 language from the possible labels, whereas the evaluation of a perfect test would in theory be able to eliminate about half of the languages. Features which characterize a subset of languages rather than single languages offer one solution for more effective tests. Constructing such features could be done by again performing independence tests to determine which languages show correlations. Feature construction could also be guided by research in linguistics, particularly the relationship between languages (Grefenstette, 1995). Understanding the composition of various language families may help in defining a distance between two languages. One could then construct a tree using tests that partition data based on a threshold distance value. This may be more effective in partitioning data into subsets, implying a higher gain in information. Significant improvements are expected in the performance of this algorithm with the use of tests more effective at partitioning the data-space.

In comparing the Naïve Bayes and Decision Tree Classifiers a redundancy was observed. Both classifiers utilize the notion that possible languages can be eliminated by observing which characters can *not* be found in a certain language. The Naïve Bayes represents this by assigning $\hat{p}_{c|l} = 0$ for all cases which

a character, *c*, is not found in any text written a language, *l*. The Decision Tree follows the same logic by using explicitly define Not-sets. Because both $\hat{p}_{c|l}$ and the Not-sets were constructed using X_{Train} and Y_{Train}, both classifiers will have different but equivalent representations of characters which can be used to eliminate language possibilities. This lack of variety in feature selection between the classifiers implies that they will treat queries similarly and are expected to perform thus.

Differentiation between these algorithms thus falls in the classification methods used at terminal nodes of the Decision Tree that are associated with an incomplete classification. Suppose that a tree has evaluated a query, returning that the query could either be in French or Slovak. In this scenario, a MLE Naïve Bayes Classifier would assign a language label according to which language yielded the highest probability of containing the query, whereas a Decision Tree could be modified to utilize features such as language distribution to make an educated language assignment. In the above scenario this may be advantageous as the training data shows that French is 10 times more abundant than Slovak. It should be noted that this assumes that the distribution of languages in the training data is representative of the real, hidden distribution. Calculating language distribution by using the method of MLE strengthens confidence in this assumption due to the consistency of the method.

The limitations of both classifiers were largely attributed to limitations in feature selection and construction of the data. This led to a lack of confidence in the independence assumption of the Naïve Bayes Classifier, lack of quality tests used in the Decision Tree and caused the two algorithms to characterize the data, and thus perform, similarly.

VII. Statement of Contributions

While originally undertaken by a team of 3, unfortunate circumstances led to Kabir Rajput withdrawing from the team as of October 22, 2017. David Garfinkle and Parker King-Fournier divided the remaining work, with Parker King-Fournier implementing the majority of the Decision Tree Classifier which was then finished by David Garfinkle, who also implemented the Naïve Bayes Classifier. The project report was written by both team members. Parker King-Fournier wrote the majority of the Introcdution, Related Work, Problem Representation, Discussion and Statement of Contribution sections David Garfinkle used his work with the classifiers to write the sections on Testing and Validation, and Related work. We hereby state that all the work presented in this report is that of the authors.

References

[1] B. Hughes, T. Baldwin, S. G. Bird, J. Nicholson, and A. Mackinlay. "Reconsidering language identification for written language resources". In *5th International Conference on Language Resources and Evaluation (LREC2006)*. 2006.

[2] G. Grefenstette. "Comparing two language identification schemes". In *Proceedings of Analisi Statistica dei Dati Testuali*. 1995.

[3] L. Grothe, E. W. De Luca, and A. Nürnberger. "A Comparative Study on Language Identification". In *Proceedings of the Sixth International Language Resources and Evaluation (LREC2008)*. 2008.