Language Classification Experimentation via Maximum Entropy and Generative Noisy Channel N-Gram Modeling

Nikita Markovych nikimar1@umbc.edu
Harshil Dhariwal zz86656@umbc.edu
Dhaval Patel up08495@umbc.edu
Gopinath Achuthan rx44815@umbc.edu

Abstract

Classification problems in natural language processing and machine learning involve identifying the category of data using prior training data with some known categories. A well-known use for classifiers is language identification, which can provide useful context for NLP or other tasks once a corpus language is established. Maximum entropy modeling involves assigning weights to arbitrarily established features in order to discriminatively model data. Proper selection of features and good training data are paramount for effective use of said model. Noisy channel generative classification involves repeated Bayesian modeling of probabilities to reduce noise and retain greater model accuracy given iteration over a training set. The purpose of this research will be to explore best practices for modeling language classification via various forms of Noisy Channel N-grams and Maxent Modeling.

1 Introduction

The motivation of this project problem is to work with various methodologies and establish best practices for classifying several universal dependency languages. The problem is multifaceted due to the varied merits and possibilities for proposed models. Maximum entropy models are very effective discriminative classifiers but feature selection can be difficult and is largely dependent on subjective choices. Furthermore, a sufficiently large training set is necessary for effective modeling. N-gram modeling and Bayesian generative classification are dependent on various hyperparameters which may best fit data using different values across various languages. We will attempt to classify some number of languages at the same time, be it 3 or 5 or 10, yet we are largely uncertain how effective our models will be as more languages are looked at. Thus, initially, we are considering English, Russian, French, Spanish, and Italian universal dependency sets. We may combine multiple corpora of the same language for a more robust vocab in Maxent modeling. Furthermore, the means of dealing with out of vocab words are especially relevant to language classification where large numbers of out of vocab words will be encountered as one compares foreign languages. A simple maximum likelihood estimation model will not likely be sufficient given the incomplete vocabulary of training sets, yet Laplace may be too crude and assign an excess probability to foreign language as unknown words. Backoff and interpolation, as well as other ways of dealing with n-grams, will have to be looked at through all proposed models will be analyzed for best fit. The sheer volume of parameters that can be tweaked can also make effective modeling difficult. The proposed solution is to create several model proposals and tune them via gradient descent. One will look at F1 scores and first tune models across some two languages, later scaling up the language classifier to handle more languages while it is still effective. Limitations of our classification methods will thus be established as well as finding the empirical best modeling practices.

2 Identifying Approaches

At this time we shall first describe our approach to solving the simpler problem of single language classification. Note that all model F1, precision, and recall scores were calculated as a macro average among 6 languages unless otherwise noted. We adopted a few of the known solutions in order to build the best model possible such that we can later attempt a multi-class classification model building upon this basis.

2.1 Laplace Noisy Channel

The first model we created was a Laplace noisy channel classification model. In this model, one used Bayesian inference to find the label by multiplying the prior probability by the likelihood of some observed sentences. The formula for this can be summarized as

$argmax(P(y|x) * P(x)$

Here we estimate the probability of sentence y given some label and given prior probability of said label. We built a model with 6 languages, those being French, English, German, Spanish, Italian, and Dutch. The laplace probability of languages given some sentences was calculated by using several universal dependency training corpora which we are submitting in CoNLL-U format. We first calculated the probability of unigrams by counting up word type instance counts. Each sentence had ¡BOS¿ and ¡EOS¿ tags appended to it. We then added a laplace constant to all such words as well as creating a ¡UNK¿ token with a count of laplace. We then would calculate the probability of a unigram by dividing the specific token's count by total count of work tokens + (laplace constant * V) where V is the total count of word types. We checked that our probability thus added up to one and that all normalization was correct. We also initially allowed for varying cutoffs at which point a word token count would be ignored in unigrams and labelled as unknown in bigrams.

With our model, an OOV cutoff of 15 and laplace constant of .1 actually created a very accurate model with Precision of 0.9803387044134281 Recall of 0.9872388809192215 and F1 Score of 0.9837766934138821. However, upon further realizing that we would be using our model to evaluate and classify short twitter sentences, we realized that a high cutoff was not the best decision and we removed the OOV cutoff functionality, leaving it at 0. If we remove rare words, this assigns more influence to common words such that it may be easier to classify a language from a robust corpus seeing as it will have lots of evidence of said common words. In this use case, such a model would be very effective where so much weight is given to common words. Furthermore, rare usage of foreign words mixed in with some language corpus would not corrupt our model. However, with short tweets, one might only have that rare word present in order to identify some tweet. Tweets are very much not a robust corpus that can benefit from simplifying the training text. In fact, a means of working with noise from internet spelling errors and vernacular through better internet training models as opposed to universal dependency data taken from non-twitter mediums might have been helpful. Regardless, our model actually used bigram laplace probabilities for calculation of sentence probabilities for each language label. Thus, we will further describe the laplace maximum likelihood estimate of each bigram. We calculated the probability of each sentence by dividing it into bigram word pairs and then multiplying their probabilities in a Markov Chain. Their probabilities were estimated as, for some words (x,y) in that order,

$p(y|x)$ = token count per wordpair (x, y) + laplace constant divided by unigram count of x + laplace constant $\times$ count of types for normalization.

We again confirmed that our bigram probabilities added up to one for a few contexts x. The model later evaluated language class by first calculating prior language probability. This was simply accomplished by dividing the instances of a sentence in some language by the count of all instances of sentences in all languages. We then utilized a development set as input for training the models followed by running the models on a test set. The development sets would be iterated over creating bigrams and unigrams and the likelihood of some bigram would be assessed such that probability of some bigram we did not observe in training would be laplace constant divided by unigram probability count of word y if word y from word pair (x,y) exists in our unigram vocabulary + the aforementioned laplace*V. If word Y was not present in the unigram, we used the single laplace constant divided by count of tokens which was used for unigram normalization. Thus we multiplied estimated probabilities from the development or test set per bigram word pair to reach estimated sentence probability. After this, we would add up the logarithm of these nonzero sentence probabilities per each language and multiply them with the prior probability per language. This is the aforementioned argmax equation as we would then classify each sentence as the language with the highest sum of log(p(sentence—label)) +

log(p(label)). We are able to do so due to logarithms identities. Using this model we tested dev sets on various laplace constants with the new lack of out of vocab cutoff. This resulted in slightly worse accuracy for models and our best models were for laplace constant of .1 or .2 with the same result of Precision being 0.9735010741316729, recall being 0.9785422190341646 and F1Score being 0.9760151372134757. We tested laplace constants from .1 to .9. We then used the laplace constant of .1 which was evaluated as the best fit on dev data to evaluate performance on test data from those 6 universal dependency language sets. We found that the result was Precision of 0.9216475969779904, recall of 0.962572808650951, and F1Score of 0.9416657556188366.

2.2 Bigram Katz Backoff model

We then proceeded to create our own bigram katz backoff model. This model worked by taking some probability from each bigram and unigram by subtracting a constant less than 1 from bigram and unigram counts and then redistributing that probability as necessary among all discovered 0 probability bigrams and unigrams in the development set. How this worked is one first had probability of the bigram when the bigram exists in our data being calculated as count of (x,y) word pair occurrences - bigram backoff constant divided by unigram count of (y).

This is exactly the same as laplace probability without a laplace constant or extra normalization for said laplace constant. Of note is that we calculated the redistributed probability as 1 - summation of all such katz backed off bigram probabilities given some context x. We thus created a list of a(x) values based on word x per x,y pair which would be multiplied in the Backoff step. Backing off to a unigram, if the unigram y context exists even if bigram (x,y) does not exist, we would multiply by the backed off probability a(x). Note that if x does not exist but y does exist, we would use the a(¡unk¿) value for a(x). Unknown probability was calculated by assigning an unknown unigram token with a count of one and calculating unigram probability normally. Unknown bigrams had a 0 probability and were always backed off. If count of y ¿ 0 we would take a(x) * count of y - unigram backoff constant divided by number of word tokens for normalization. If count of Y is 0 we would backoff one more time this time redistributing the unigram probability. This was accomplished by finding the constant which is one minus the sum of all (unigram count - unigram backoff constant) / token count probabilities. We then redistributed this probability equally to all discovered dev set unknown unigrams types after first aggregating all unknown types. Once we aggregated unknown types we were able to divide the redistributed unigram probability constant normalized and divided among them. With this backoff model we tested it out and saw very accurate single language classification such that we did not feel the need to play with katz backoff constants too much. We had the following results on the development set with backoff unigram constant .5 and backoff bigram constant of .5: Precision of 0.9873689170618313, Recall of 0.9930225955790527, and F1Score of 0.9901876414351766. With unigram backoff .3 and bigram backoff of .1 we had Precision of 0.9824439537110345, Recall of 0.990375266882618, and F1Score of 0.9863936671916576. We finally ran one last dev set test with unigram backoff of .1 and bigram backoff of .3 with the results of Precision: 0.9885983109124509, Recall: 0.9933143497382311, and F1Score: 0.9909507193257626. We then ran it on the test set with backoff unigram and bigram constants of .5 and got a result of Precision as 0.9716045640704526, Recall as 0.984300777464 and F1Score as 0.9779114638051326. We finally upon realizing the .1 .3 result was slightly better for our dev set ran those constants on the test set and got the result of Precision: 0.9725033589149076, Recall:0.9868233234575854, and F1Score: 0.9796110116318264 which was slightly improved. We thus saw that with consistent reproducible results our backoff model was more effective than our laplace model. We decided to test our multi-language classifier using a similar model and merely modifying it to output the two highest log linear probabilities instead of the single highest one as the label. We did not bother with full grid testing nor gradient ascent especially considering that some aspects of the model are not very convenient for gradient ascent. We would be training our models on multiclass data and wished to focus on that considering we had a very effective model already with not much variance as we modified backoff constants.

2.3 MaxEnt and Laplace Modelling

We furthermore experimented with using the scikitlearn modules for maxent modeling and laplace modeling in order to classify single language labels. We were not sure if we would have the flexibility in adapting them to multiple language classification which takes in the number one and number two most likely language but we did want to experiment with other modules that we did not build ourselves in case we find an effective means of modeling using the toolkit. We were ultimately unable to create a multi-language classifier with sickitlearn but we did succeed in single language classification of the Wortschatz Leipzig corpora with a high degree of accuracy using both a laplace and maxent model on the German, English, French, and Italian branches of said corpora. We will not go into too much detail about these models as we did not end up using them for the ultimate goal of our project and we are also not fully aware of their implementation. We are aware of the basic principles for laplace models but we do not know if they used a noisy channel along with their laplace model. We are furthermore unsure of their implementation of maxent. The results were as follows for laplace 1.5:

	Precision	Recall	F1
English	1.00	1.00	1.00
French	1.00	0.98	0.99
German	0.99	1.00	1.00
Italian	0.99	0.98	0.99

Above are the results as follows for Laplace:

	Precision	Recall	F1
English	1.00	0.98	0.99
French	1.00	0.98	0.99
German	1.00	0.98	1.00
Italian	0.96	1.00	0.98

Above are the results as follows for MaxEnt:

3 Concluding Approaches

Having established several workable accurate models for single classification, we decided to procure data and create some form of a multi-language classifier that discovers the top two most likely languages for corpora or sentence. This was motivated in part by previous research on code-switching, translation of code switched to text, sentiment analysis of code switched text, and other applications. A lot of research was preoccupied with sentiment analysis and translation which interpret the meaning of said code-switched texts and are interesting applications. We did not, however, find much data to use for research of code-switched text nor did we know of a standard best practice for analyzing code-switched text. We thus decided to focus on a simple problem of language classification but find the best models possible with imperfect data. We also forayed into methodologies for procuring corpora and data which were quite interesting if not always fruitful. The limitations of our methodologies and research as novices of natural language processing applications provided a lot of insight for future approaches to problems. One takeaway is that for creating this sort of corpus, outsourcing data for annotation is one of the best solutions. We were looking at twitter data and used some methods to improve our chances of finding multilingual tweets but we still had many difficulties finding data within our own means. We could have simply procured a large number of tweets with a certain hashtag or in a certain area that we presume might be more likely to feature bilingual speech. This would create a lot of junk data to sift through but would also allow for large-high-quality data once parsed by outsourced labor. Our methodology for procuring data was more complicated although somewhat inspired by the methods used in the In Proceedings of the Second Workshop on Language in Social Media research paper. One method they used was having bilingual research group members and friends outsource and find twitter accounts that frequently tweeted bilingual tweets.

4 Challenges faced

Due to time constraints, we did not make much use of this method but we did find a few select tweets from frequent bilingual accounts. We could have queried all tweets from such users and parsed them by outsourcing them for labeling as multilingual or not. We would have been able to create a large corpora of multiple code-switched language combinations.

Another methodology we attempted was a simple query using twitter API based on geocode and language. Our reasoning was that a foreign language in some country might likely be mixed with the native language of said country. However, testing this methodology, we found about 3 tweets in a 5 km radius of some location in southern California classified as Spanish. They were wholly Spanish with no English. Expanding our search to a greater radius may have taxed twitter API or may have taxed our ability to process data without crowd-sourcing resources. We found no tweets within a 5 km radius of a random point in New York that were Spanish. This was using the twython module. Using another module, we were able to find a vast amount of English tweets in Madrid but they were in a JSON format with many excess data unrelated to their language content and thus difficult to parse. They were furthermore predominantly tweets of English tourists with no Spanish content upon a cursory glance. Going over the data of over 16000 tweets was not feasible for us. We also had delays and difficulties getting approval for twitter API use and because our queries were limited as a free development account, we were not easily able to procure all of the robust data we desired. We thus sought ready-made code-switched corpora and found some references to a CoMAprend corpus which contains most of the languages we intended to look at in a single corpus, those languages being Spanish, French, German, English, Italian. We were unable to find this corpus for download as well as looking at a few other corpora but ultimately failing to find data. We went back to the initial plan of querying twitter data and parsing it ourselves such that internet data filled with hashtags and emojis could be correctly interpreted by our models despite the difficulties posed by such data. We ultimately were able to find a blog post by twitter engineers that discussed their methodology and standards for language classification. We discovered that criteria for a tweet language being labeled as unknown included the criteria that it may be equal parts two languages. Unfortunately, unknown language tweets also featured no language and only emojis, URL's, hashtags, and otherwise ambiguous language text.

We furthermore read that tweets of a certain language would not be labeled unknown if only a little bit of the tweet was in another language. We, therefore, decided due to time and query constraints to pursue data that contained Spanish and English language. We would have to process this data extensively which as mentioned was not fully possible with our resources. We however had a list of twitter ids the Twitter engineering team had identified during their review of language identification effectiveness within their system. Thus, we had a limited corpus already prepared for analysis and extraction which we queried using the twurl utility installed via ruby gems. We were able to read out tweets from the JSON output standard into an output useful for us although this took some code on our part. We were then able to aggregate these tweets into a text file separated by tweet id and then tweet sentence for verification of tweet download success. We needed to leave ids for easy restart of timed out twitter API queries and this process still took several hours. In the end, we had approximately 6500 Spanish tweets and approximately 5500 unknown tweets to process for bilingual Spanish English data. Later we parsed out just the tweet sentences into CSV files such that they words were comma-separated and sentences were line separated. This did not conclude our parsing because tweets contained emojis and a lot of other non-language instances, as well as containing concatenated no space language in the form of twitter hashtags. We found a python module based on a three trillion word corpus known as word segment which we were able to use along with wildcard parsing to separate hashtags into words as well as cleaning up some punctuation, emojis, and other junk. We were not able to remove URL tokens with this python module although given time we would have found an even better methodology for parsing data into readable English and Spanish. Lastly, we had to remove some now empty comma-separated values manually since in all this time we were able to only find a few Spanish English tweets. Given the small size of the data this last parsing step was done manually since it would have been inefficient to create a program for it, although for future implementations and quality of life changes we would have moved forward with code for this final parsing step. Regardless, finding bilingual tweets that we later removed empty values from was accomplished by a command-line program we made that allowed us to select bilingual tweets from the unknown and Spanish tables, concatenating them into a CSV file. We are not Spanish speakers and we had some troubles with parsing this data ourselves. One corpora of 76 tweet sentences we produced was created after looking through approximately 1800 lines of our table. Another team member later reviewed it and found only 36 lines that were Spanish-English as per his standards. Some of these sentences contained only three words one of which was a URL. This obviously threw off our model for this particular sentence which was unlikely in any language and had its probability determined far more by which language was more likely to have unknown tokens than anything else. We concatenated 4 sentences from a set of 7 really good sentences one team member found at random without the program in order to create our final test set. We also tested on those seven unambiguously Spanish English sentences and found that our model correctly identified all but one.

We will discuss our bilingual models at greater length later. Regardless of imperfect data, this has been a useful analysis of methodologies for procuring corpora data and moving forward we are aware of several methods that would have been very effective provided that we hired annotators. We could potentially create many multi-linguistic corpora that do not as of yet exist to the best of our knowledge so this attempt proved a useful learning experience on creating our own data.

5 Results

Since our model contains only Spanish English data with one possible outcome, the use of precision and recall was somewhat redundant due to false positives and false negatives being equal while true positives were also always Spanish- English. However, testing the final data set of 40 items gave us .425 precision-recall and F1 score. Testing it on the good set had a 6/7 precision-recall and F1 score. One can easily run a test on the 76 word set as well or a set one produces oneself in our utilities. Remember to remove extra commas since some blanks will exist after our multiple parsing algorithms. Our model was not very successful. However, of note is that we printed which languages were identified and found that sentences of sufficient length and lacking ambiguity in terms of misspelling and otherwise were easily identified with great accuracy. Data we were uncertain was bilingual was not as effectively identified. Given better processing of URLs and typos and a better corpus, we would easily have been able to identify two languages. Also of note is that due to the small amount of data, we only used universal dependency data for development and training. We backed off with .5 and .5 constants. We worked with a model trained on very different and sub optimal data and found great success with that data which was parsed effectively. Prior to removing extra blanks, we had much worse results of .2 precision because empty spaces were more likely to be dutch as per our training corpora. We were able to greatly improve performance with a little processing and could see great gains in creating corpora and models moving forward. Due to time constraints and data procurement difficulties we did not create our own interpolation, interpolated backoff mode, or goodwin-turing backoff model. However, given time and fewer data troubles we could have tweaked models at great length and eventually found the most effective model. Our laplace model was not used because it was less effective than the backoff model for single language classification and we, therefore, did not have high hopes for it.

6 Some additional notes

Please refer ReadMe files created extensively throughout the code for help running it and understanding the intent.

7 Future Scope

In the future, we would like to have tested data of many multilingual sets, not just Spanish and English for creating the best possible model as a standard which can then be used for other purposes than simple language identification. Some of our reasoning for our research pursuits will follow.

8 Conclusion

For the purposes of understanding research and model context, a definition of code-switching is necessary. Code-switching is defined as the practice of alternating between two or more languages or varieties of language in conversation or written language. Building models that accurately work with multilingual data can be difficult since they may engage in switching on a sentence-by-sentence basis or even a word by word basis (Bergsma, McNamee, Bagdouri, Fink, Wilson. 2012). This could greatly hamper the effectiveness of bigrams that rely on neighbors within one word of each other to calculate probability. This could pose difficulties for many other models as well.

One methodology we had considered but were not able to implement was looking at the probability of words being within a certain range of each other. We also hoped that methods like backoff or interpolation would be effective since they attribute probabilities to smaller level n-grams such that one is able to compensate for words being used in an unusual bilingual context not present in monolingual training corpora. One other aspect of research we wanted to delve into was specifically using a monolingual model training corpora which are imperfect for training code-switching identification to still accurately identify bilingual test sets. This is due to the sparsity of code-switched language data at least in the domain of open-source data. We had a lot of difficulties procuring good data and believed training the most effective models possible would be sufficient for building models with imperfect data. Regardless this has been a fascinating endeavor into the realms of both creating and parsing corpora from imperfect internet data and building our own effective models for it. We were not fully able to realize that goal due to limitations on data yet we had great success with monolingual data classification. Also of note is that on our small corpora we frequently identified one of the two languages correctly. This introduces another issue in that Latin languages might have more in common and skew our classification method even with the presence of English words in such tweets. It would be interesting research moving forward to deal with that issue although it seemed to happen less frequently than data issues such as sentence length, non-linguistic URL data, or erroneous language data that our parsing did not fully clean up. Thus our research has shown possible difficulties and solutions of modeling multiclass language, possible difficulties and some solutions with parsing data from an internet medium, and possible difficulties and solutions for procuring data that can be pursued further.

9 References

Anil Kumar Singh and Jagadeesh Gorla. 2007. Identification of languages and encodings in a multilingual document. In Proceedings of ACL-SIGWAC's Web As Corpus3, Belgium.

ARTEMENKO O., MANDL T., SHRAMKO M. et WOMSER-HACKER C. (2006), "Evaluation of a Language Identification System for Mono- and Multilingual Text Documents", in Proceedings of the 13th Annual Workshop on Selected Areas in Cryptography,Dijon, France.

Shane Bergsma, Paul McNamee, Mossaab Bagdouri, Clayton Fink, and Theresa Wilson. 2012. Language identification for creating language-specific twitter collections. In Proceedings of the Second Workshop on Language in Social Media, pages 65–74, Montreal, Canada, June. Association for Computational Linguistics.

Suraj Maharjan, Elizabeth Blair, Steven Bethard, and Thamar Solorio. Developing language-tagged corpora for code-switching tweets. In The 9th Linguistic Annotation Workshop held in conjuncion with NAACL 2015, page 72, 2015.