Native Language Identification using Probabilistic Graphical Models

Garrett Nicolai[1], Md Asadul Islam[1*], and Russ Greiner[1]
[1]Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada
[*] E-mail: mdasadul @ ualberta.ca

Abstract— **Native Language Identification (NLI) is the task of identifying the native language of an author of a text written in a second language. Support Vector Machines and Maximum Entropy Learners are the most common methods used to solve this problem, but we consider it from the point-of-view of probabilistic graphical models. We hypothesize that graphical models are well-suited to this task, as they can capture feature inter-dependencies that cannot be exploited by SVMs. Using progressively more connected graphical models, we show that these models out-perform SVMs on reduced feature sets. Furthermore, on full feature sets, even naïve Bayes increases accuracy from 82.06% to 83.41% over SVMs on a 5-language classification task.**

Keywords— NLI, Machine Learning, SVM, Bayesian Methods, TAN

I. Introduction

Year-by-year, it is becoming more important to know one of a small number of languages if one wishes to succeed professionally. In particular, the number of people who speak English as a second language is growing faster than any other language. As more people devote their time and effort to learning the language, it is also becoming more important to be able to identify difficulties that may arise in the learning of English.

Native Language Identification (NLI) is the task of identifying the native language (L1) of a writer of an essay written in a second language (L2). The task has seen interest mainly in the past ten years, with a concentration of Machine Learning algorithms such as Support Vector Machines (SVMs) being applied to the problem. Probabilistic Graphical Models (PGMs) provide an alternative method of tackling the issue, and provide certain benefits over SVMs. Where SVMs can predict the L1 of the writer, PGMs can also infer likely feature groupings, given observed evidence, allowing second language teachers to adapt curricula to a student's particular needs.

Our paper is organised as follows: Section II describes the important work related to the NLI task; Section III briefly describes the data used to learn and test our models; Section IV describes the features that have been chosen to learn our models; Section V provides the methods that we use to build our graphical models; Section VI gives the results of our experiments, and Section VII provides some conclusions and areas for future work.

II. Related Work

Koppel *et al.* [9] are one of the first groups to seriously consider the problem of NLI. Using a feature set that includes function words, character n-grams, errors, and rare part-of-speech (POS) bigrams, the authors train linear SVMs to classify English essays into five different L1s, obtaining 80.2% accuracy. Tsur and Rappoport [13] further analyse the importance of character bigrams, obtaining 65.6% across the same five L1s. Our approach takes inspiration from these papers, but we approach the problem differently. Aside from using PGMs instead of SVMs, we use a slightly different set of features. As in these papers, we originally considered only 200 features, but sort them by information gain, rather than frequency. Furthermore, we gradually increase the number of accepted features, and allow content words, as well as function words. We also use the most informative POS bigrams, instead of rare ones, and have our own implementation of spelling errors.

Wong and Dras [14] abandon SVMs in favour of a maximum entropy learner to classify essays across seven different L1s. Furthermore, the authors are more concerned with syntactic than lexical features, introducing rules generated by syntax trees as features to their model. By adding syntax rules to the feature set of previous work, they obtain around 80% classification accuracy. Although our features are mostly lexical, Wong and Dras consider using more than just 200 features for some of their feature sets. They also demonstrated that the rare bigrams of Koppel *et al*. [9] did not contribute to a classifier's accuracy, and motivated our decision to use more common and informative part-of-speech bigrams.

Bergsma *et al.* [1] tackle a slightly different task than the one that we are investigating. Using SVMs, the authors analyse scientific papers, and classify them according to three criteria: whether or not the first author's L1 is English, the gender of the first author, and whether the paper is a journal or workshop paper. This task is different from ours, but uses many of the same features that we consider, such as word n-grams and POS n-grams, as well as other features such as syntax tree rules.

Tomokiyo and Jones [12] also investigate the not-quite-NLI-task of native speaker identification, but rather than using SVMs, the authors use a naïve Bayes classifier. They consider both words and POS as features to their classifier for two types of transcribed speech: spontaneous speech and read text. We also consider words and POS, but on written text, which can be significantly different from transcribed speech, and our task is different from the task of identifying whether or not a speaker is a native speaker. However, this work encourages us as we also use PGMs, albeit for the NLI task.

Graphical models have found uses in an area similar to NLI: text classification. Like NLI, text classification aims to use features of the text to separate documents into several classes, based on their content. However, unlike NLI, text classification is often able to look for a small set of keywords that are highly indicative of their class: business documents rarely discuss baseball, unless it is about the business side of the sport. Essays can be about various topics, yet still have the same L1.

Lam and Low [10] use Bayesian networks to perform text classification, learning the structure of their network from their data, and obtain an F-measure of 0.53. Although this task is different from ours, we use the same method as the authors of representing our documents: binary indicators of the presence or absence of features. Khor and Ting [7] also use Bayesian networks for the task of text classification, discovering that Bayesian classifiers perform admirably at the task of separating conference papers by topic, obtaining 90% accuracy when compared with human experts performing the same task.

III. Data

Our data set is the TOEFL 2011 Corpus of Non-Native English [2], which contains 9900 essays evenly distributed across eleven languages and eight essay prompts. However, to make the task more manageable, we have reduced the data set to five languages: Chinese, French, German, Japanese, and Turkish. We thus have 4500 essays across these fives languages.

These languages were chosen to represent a set of languages that contained both linguistically related languages, such as French and German, culturally tied languages such as Chinese and Japanese, and Turkish: a language that is not culturally or linguistically related to any of the other languages. The data set was split into 90% for training and 10% for testing.

IV. Features

Our features were selected from a set determined in previous work [11]. These features have been shown to improve classifier accuracy for the NLI task when using SVMs, and have thus been chosen as features for PGMs. All features are binary; if a document contains the feature, the value is set to 1, otherwise it is 0. The features are chosen from the training set only, and any features encountered in the test set that were not in the training set are ignored.

A. Word Unigrams

If asked for the smallest unit of meaning in languages, many people would suggest the word. Long works are constructed of paragraphs, which are made up of sentences, which in turn are composed of words. While there are smaller units of meaning, words do contain much information.

Furthermore, writers from different linguistic backgrounds may prefer words that match ones in their first language, or make common spelling mistakes that can be captured at the word level. Unfortunately, the set of all words across all documents is rather large, and document vectors composed of words are very sparse. We perform some feature reduction, which we describe further in Section V.

B. Character Bigrams

Even smaller than the word, characters can provide information about the tendencies of an author. Certain linguistic backgrounds may prefer certain spellings, either more reflective of the spelling of their L1, or more representative of the sound of the word. English is a very orthographically dense language, that is, one letter may be associated with many different sounds, and one sound is not necessarily tied to a single letter. Consider the sound 'ə', which is the first sound in “about” in fast speech. In this case, it is represented with an “a”, but in “burn”, it is represented by a “u”, in “woman”, by an “o”, and so on. Similarly, “a” can represent different sounds, such as the ones in “cat” and “arm”.

If a language usually uses one character for one sound, a native speaker may default to that letter if he is unsure of the spelling of a word. Character unigrams are not particularly informative: the letters in English have a given distribution amongst words, and changing the frequencies slightly contains little information. Character bigrams, on the other hand, provide context for when the letters are incorrectly used. If a document contains the bigrams “hi” and “in” for the spelling “thin” when the writer meant “then”, it provides more information than if we know that “i” was used instead of “e”.

C. Part-of-Speech Bigrams

More information is contained in a word than just its meaning. Every word also carries grammatical functionality. Using annotated files generated by the Stanford Parser [8] for previous work, we were able to construct POS bigrams from the essays. As with characters, POS can be very informative of the writer’s L1. Languages vary considerably from one to another. Closely related languages tend to have similar grammatical structures, but unrelated languages have different syntactic rules that can be captured by POS. Consider the following example from German: *Obwohl es warm ist, will er nicht draußen gehen*, which literally translates as “Although it warm is, wants he not outside to go”, and means “Although it is warm, he does not want to go outside to play”. We can see that after the comma, we have a verb followed by a pronoun, which is a very awkward construction in English, but is perfectly fine in German. This feature may be indicative of German as L1.

D. Spelling Errors

Although some aspects of spelling mistakes are captured by the character bigrams and word unigrams, we also have a feature that explicitly checks for spelling errors. Essays are evaluated by *ASpell*[1], and the spell-checker gives a list of recommended correct spellings. We choose the top recommendation as the correct spelling, and compare it with the incorrect spelling. The two spellings are aligned using

1 http://aspell.net

M2MAligner [5]. Consider the example misspelling of "computer" as "kompyuta". The aligner would provide the following alignment:

```
c o m p u  t e r
| | | | /\ | \/
k o m p y u t a
```

From this alignment, we get three error alignments: c-k, u-yu, and er-a. We use these alignments as features for our classifier.

V. Methodology

The following section describes our methods for learning graphical models to classify essays by the first language of their writer. First, we describe methods used to reduce the feature set to a manageable number of input features. Secondly, we describe various structure learning methods that were applied to determine the best graph.

A. Feature Reduction

As mentioned in Section IV, our original feature set was very large, containing approximately 50,000 features. Learning a graph on all 50,000 features would have been prohibitive, so we tried to reduce the dimensionality of our feature set.

There exist in the NLI literature several methods of reducing the number of features. Koppel *et al*. [9] and others ignore content words in essays, and only consider function words. Function words can be considered the "building blocks" of languages. They have little meaning themselves, but are necessary to build grammatically correct sentences. Function words belong to a small number of POS classes, such as prepositions, pronouns, determiners, and conjunctions. Some examples of these words include "for", "I", "the", and "and".

By reducing our word set to a list of 295 function words, and only considering features that appear in these words, we reduce our feature set from 50,000 to approximately 5000 features.

Furthermore, we use information gain on our feature sets to further reduce the number of features. Previous work [11] showed that when you only want to use a subset of features, information gain can be helpful in selecting useful features from a larger set. The equation used for information gain is provided in equation 1, where H is the Entropy, defined by equation 2.

$$IG = \sum_{i=1}^{2} H(word = i) - \sum_{c=1}^{5} \sum_{i=1}^{2} H(word = i | class = c) \quad (1)$$

$$H = -P(x) \times \log(P(x)) \quad (2)$$

B. Structure Learning

Even with a reduction to 1000 features and one class variable, the number of potential Directed Acyclic Graphs (DAG) that can be constructed is super-exponential, and we do not know which is the correct one.

We decided to build several different types of graphs in an attempt to maximize our chances of finding one that approximates the true I-Map for the data, which are described in subsections 1) through 4).

1) Naïve Bayes

The very simplest method of constructing a graph is to assume that all features are independent of each other. This graph gives us no dependencies, but the next simplest method, to assume that features are independent if the class variable is known, at least provides a starting point. As we have 50,000 features, naïve Bayes was attractive, with its minimalist approach.

Khor and Ting [7] also used the naïve Bayes classifier as a baseline for text classification. While our task is not text classification, the challenges are similar. We use naïve Bayes as our simplest classifier, and as a baseline against which more connected graphs can be compared.

2) Tree Augmented Naïve Bayes

Following the work of Chow and Liu, [3], we investigate the Tree-Augmented naïve Bayes (TAN) classifier. The TAN tree starts as a Naïve Bayes classifier, where all nodes are assumed independent of each other, given the class variable. A spanning tree that maximizes mutual information between nodes is then constructed.

The number of arcs in a TAN network is twice as many as in a naïve Bayes, but is still manageable. All of our features are binary, and thus, the number of parameters in the network is also approximately doubled. A TAN is able to encode some dependencies in the data. Furthermore, TAN networks scale well with the number of features, and should be able to be constructed in a reasonable amount of time. A network with more dependencies may also be more prone to over-fitting the training data.

3) K2 Hill-Climbing

Although a TAN network models some dependencies in the data, it is still a graph with very few arcs, and may be missing key relationships between nodes. Consider the graph in Fig. 1. The character bigram "ck" is likely related not just to one word, but to many, such as *black, kicked*, as well as others that are not shown. Likewise, each of those words is related to n-1 character bigrams, where n is the number of letters in the word.

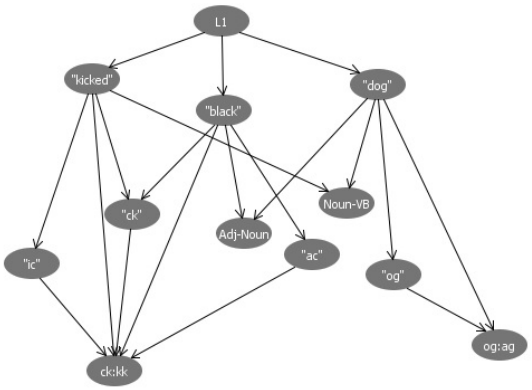

Figure 1. One of many potential graphs for a small number of features

Every word also has a likelihood of being tagged as a particular part-of-speech, and is then likely to be associated with any POS bigram that contains that POS. For example, in the sentence "The man kicked the black dog", *black* is an adjective, and is thus tied to the *Adj-NN* POS bigram. Similarly, *dog* is a noun, and would be tied to all POS bigrams that contain a noun. Likewise, words can be associated with several spelling errors. It is easy to see that one node may have relationships and dependencies with many others.

For this reason, we consider the K2 Hill-Climbing algorithm, as provided by Weka[2]. K2 uses an ordering of the variables to determine the best assignment of parents to nodes, using a particular scoring function. We have chosen the Bayes score, represented in equation 3, where N_{ij} is the number of documents where a node's i^{th} parent takes its j^{th} value, and N_{ijk} is the number of documents where a node's i^{th} parent takes its j^{th} value, and the node takes its k^{th} value. $\Gamma(x)$ is the gamma function, which extends the factorial (in $\Gamma(x) = (x-1)!$ for an integer x). Given standard assumptions, this function represents the probability of a structure (marginalized with respect to the parameter values).

$$Q_{Bayes(BS,D)} = \prod_{i=0}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(N'_{ij})}{\Gamma(N'_{ij} + N_{ij})} \prod_{k=1}^{r_i} \Gamma \frac{(N'_{ijk} + N_{ijk})}{\Gamma(N'_{ijk})} \quad (3)$$

The K2 algorithm iteratively adds parents from the previous nodes in its ordering, up to the maximum specified. It chooses the parents based on which ones lead to the greatest increase in the score of the network. As we are unsure of the true ordering of the variables, we start the K2 algorithm with a random ordering, with the exception that the class variable is always the first variable in the ordering. Furthermore, we also require that each variable is dependent on the class.

Weka ensures that all variables lie within the Markov blanket of the class variable. The network is initialized as a naïve Bayes network, and edges are added to the network.

2 http://www.cs.waikato.ac.nz/ml/weka/

4) Support Vector Machines

As an alternative to graphical models, we consider SVMs. First proposed by Cortes and Vapnik [4], SVMs compute a high-dimensional hyper-plane that separates the data, using a small number of instances, known as Support Vectors, to determine the location and orientation of the hyperplane. We use a package for multi-class classification that is extended from the *SVM-Light* Package [6]. We use a linear kernel.

VI. Results

Our data set contained 4500 essays that were divided into a 90% training set and 10% test set. We learned naïve Bayes networks, TAN networks, and more-fully connected Bayesian networks on the training data, as well as SVMs, and performed inference on the test data. We compared to the Bayesian networks. Beyond testing the accuracy of our learners, we also run an ablation study on one of our networks, and compare the results to a similar ablation study using an SVM. We wanted to determine whether the same features were contributing to the network as to the SVM.

A. Using Function Words for Feature Reduction

Section V describes how we use function words in an attempt to reduce the dimensionality of our feature set. Table I shows results for networks built using only function words, compared with their equivalent networks using all potential words. All of these networks use 295 word features, and 200 of each of the other three feature sets, as selected by information gain.

It is quickly apparent that systems using only function words do not perform as well as those that allow any words. Naïve Bayes gains 5.6% accuracy when all words are considered, TAN gains 6.96% accuracy, the 5-parent K2 network gains 8.3%, and the SVM gains 2.69% accuracy. Using function words, we see that naïve Bayes out-performs the other two Bayesian networks, but when all words are considered, naïve Bayes is outperformed by TAN, which in turn is out-performed by a network that allows even more potential parents. For the next set of experiments, we only consider networks allowing all words from the word feature set, as the trend of this set out-performing the function word classifiers continued as the networks grew larger.

B. Growing the Networks

In the previous subsection, all of the networks contained 895 nodes. However, there may be information in the other features that were not included in the network, and thus, we increase the size of the network in increments. First, we double the number of nodes allowed from each feature set; that is we choose the 400 most informative word unigrams, as well as 400 character bigrams, 400 POS bigrams, and 400 error alignments. We then double the size twice more, allowing 800 of each feature type, and then 1600. However, this tops out for some feature sets -- the character bigrams feature set only

contains 1098 items, while the POS set only contains 1418 items.

TABLE I. ACCURACY OVER 895 FEATURES

Classifier	Function Words	All Words
Naïve Bayes	55.16	60.76
TAN	54.48	61.44
K2 (Max 5 Parents)	53.36	61.66
SVM	55.16	57.85

Thus, while the classifier that allows 800 of each feature type has 3200 total nodes, the classifier that allows 1600 of each will not have 6400, but rather 5716. The results are presented in Fig. 2.

Generally, we see that as the number of features increases, the accuracy of the classifier also increases. As can be expected, the increases in accuracy slow down as more features are introduced. However, the TAN network still appears to be out-performing the SVM, as the increases are leveling off. If this trend continues, it is expected that a TAN would still out-perform the SVM for feature sets for which it is very computationally expensive to build the TAN. Furthermore, as more features are included, the SVM eventually passes the naïve Bayes classifier, but the TAN classifier continues to perform better than the SVM. Somewhat surprisingly, the classifier constructed with K2 does not seem to benefit much from an increase in features, but this may be partially due to the restriction on the number of parents.

Although K2 adds the parents that most increase the score of the classifier, K2 is restricted in the nodes that it can choose as parents, while TAN is not. The parents of a node for K2 must come from the previous nodes in the ordering supplied to the algorithm, and if the ordering is less than optimal, so, too, will be the network. One final experiment was considered, where no feature reduction was performed. Due to time constraints, only the naïve Bayes model and the SVM were able to be compared. The naïve Bayes model achieved 83.41% accuracy, while the SVM obtained 82.06%. Following the results of our other experiments, it is not inconceivable that the TAN model would improve upon the naïve Bayes, further improving upon the SVMs.

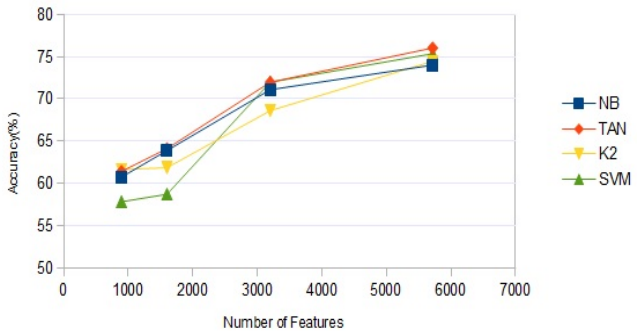

Figure 2. 10-fold cross validation accuracy, as number of features increase

C. *Ablation Study*

To help understand which factors most contribute most to our classifiers, and whether this is constant across the various learners, we conduct an ablation study on the models created with 800 of each feature type. The results are presented in Fig. 3.

While it appears that the feature sets influence the models in a similar way, with word unigrams having the most impact, followed by POS bigrams, character bigrams, and errors, there are a few differences. The accuracy of the PGMs seems to be more dependent upon the word unigrams, while SVMs show similar falls in accuracy for POS bigrams, with larger drops for the other features than the drops observed for the PGMs.

VII. DISCUSSION AND FUTURE WORK

We have learned probabilistic graphical models for classifying essays by the native language of their writer. Typically, this has been a task for machine learning methods such as SVMs or maximum entropy learners, but graphical models have certain innate advantages over the machine learning methods. As well as being able to take advantages in dependencies in the data that cannot be expressed by linear SVMs, graphical models can perform inference tasks that are beyond SVMs. If a user makes a set of mistakes, a graphical model can infer the likelihood that he will make other mistakes; it can also predict errors that differ between L1s . An SVM can make no such inference. This ability is of great importance to second-language teachers, who often need to adapt their teaching to suit their students. While this ability of the graphical models is important, we were more concerned with the construction of the models. We used four feature sets: word unigrams, character bigrams, part-of-speech bigrams, and errors alignments to learn four models: naïve Bayes, TAN, K2, and linear SVM. Using feature selection to allow graphs to be built in reasonable time, we found that the naïve Bayes and TAN models consistently out-perform SVMs when given the same features to learn a model.

Our models only consider five different L1s, but the literature has models that differentiate between as many as

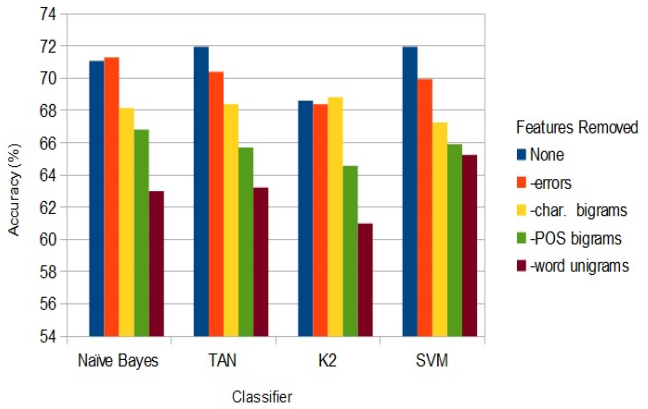

Figure 3. Ablation study for models with 3200 features

eleven. Future work can consider these other languages, and the increase in complexity that accompanies them. The feature sets that we used are only a subset of those commonly used to tackle the NLI task. It is possible that other features will further improve the accuracy of the classifier, and will do so when only a small number are considered. Furthermore, other methods of feature selection need to be considered. One disadvantage of the use of graphical models is the amount of time and resources required to build them. If we are limited in the number of features that can be used, we need to be sure that we are using the best ones available.

Acknowledgments:

The authors would like to thank Dr. Grzegorz Kondrak and Sheehan Khan for their advice and support on the research conducted in this paper. This research was partially funded by a scholarship from the National Science and Engineering Research Council of Canada (NSERC), as well as a scholarship from Alberta Innovates Technical Futures (AITF).

[1] Shane Bergsma, Matt Post, and David Yarowsky. Stylometric analysis of scientific articles. In Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 327–337. Association for Computational Linguistics, 2012.

[2] Daniel Blanchard, Joel Tetreault, Derrick Higgins, Aoife Cahill, and Martin Chodorow. Toefl11: A corpus of non-native english. Technical report, Educational Testing Service, 2013.

[3] C Chow and C Liu. Approximating discrete probability distributions with dependence trees. Information Theory, IEEE Transactions on, 14(3):462–467, 1968.

[4] Corinna Cortes and Vladimir Vapnik. Support-vector networks. Machine learning, 20(3):273–297, 1995.

[5] Sittichai Jiampojamarn, Grzegorz Kondrak, and Tarek Sherif. Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion. In Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference, pages 372–379, 2007.

[6] Thorsten Joachims. Making large scale svm learning practical. 1999.

[7] Kok-Chin Khor and Choo-Yee Ting. A bayesian approach to classify conference papers. In MICAI 2006: Advances in Artificial Intelligence, pages 1027–1036. Springer, 2006.

[8] Dan Klein and Christopher D Manning. Accurate unlexicalized parsing. In Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1, pages 423–430. Association for Computational Linguistics, 2003.

[9] Moshe Koppel, Jonathan Schler, and Kfir Zigdon. Determining an author's native language by mining a text for errors. In Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, pages 624–628. ACM, 2005.

[10] Wai Lam and Kon-Fan Low. Automatic document classification based on probabilistic reasoning: Model and performance analysis. In Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on, volume 3, pages 2719–2723. IEEE, 1997.

[11] Garrett Nicolai, Bradley Hauer, Mohammad Salameh, Lei Yao, and Grzegorz Kondrak. Cognate and misspelling features for natural language identification. In The 8th Workshop on Innovative Use of NLP for Building Educational Applications, 2013.

[12] Laura Mayfield Tomokiyo and Rosie Jones. You're not from'round here, are you?: naïve bayes detection of non-native utterance text. In Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies, pages 1–8. Association for Computational Linguistics, 2001.

[13] Oren Tsur and Ari Rappoport. Using classifier features for studying the effect of native language on the choice of written second language words. In Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition, pages 9–16. Association for Computational Linguistics, 2007.

[14] Sze-Meng Jojo Wong and Mark Dras. Exploiting parse structures for native language identification. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 1600–1610. Association for Computational Linguistics, 2011.