THE UNIVERSITY OF CHICAGO

NATURAL LANGUAGE PROCESSING FOR NAMED ENTITIES WITH WORD-INTERNAL INFORMATION

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

BY
SONJIA WAXMONSKY

CHICAGO, ILLINOIS
DECEMBER 2011
# TABLE OF CONTENTS

ABSTRACT ................................................................. iv

ACKNOWLEDGEMENTS ............................................... v

LIST OF FIGURES .................................................. vi

LIST OF TABLES .................................................... viii

Chapter

1 INTRODUCTION .................................................... 1

2 WORD ORIGIN RECOGNITION FOR NAMED ENTITIES .......... 3
   2.1 Related works in speech and language processing ............ 3
   2.2 Related works in life sciences and ethnicity classification ..... 9
   2.3 Methods ...................................................... 10
      2.3.1 Cluster-adaptive sampling ................................... 12
      2.3.2 Spectral clustering ......................................... 12
   2.4 Experiments with Supervised classification ................... 13
      2.4.1 NGA Place Name Data ...................................... 14
      2.4.2 Classifiers ............................................... 15
      2.4.3 Experiments ............................................... 20
      2.4.4 Hand annotation .......................................... 24
   2.5 Cluster-adaptive sampling .................................... 25
      2.5.1 Cluster-Adaptive Sampling Implementation ............... 25
      2.5.2 Experiments ............................................... 27
   2.6 Spectral Clustering .......................................... 30
   2.7 Word origin detection of US Surnames ....................... 31
      2.7.1 Experiments ............................................... 36
      2.7.2 Error analysis ............................................ 36
   2.8 Conclusion .................................................. 39
3 GRAPEHE-TO-PHONEME CONVERSION USING WORD ORIGIN INFORMATION ......................................................... 42
3.1 Related Works .................................................................................................................................................. 42
  3.1.1 Automatic speech recognition .................................................................................................................. 43
  3.1.2 G2P conversion for TTS .......................................................................................................................... 46
3.2 Sequitur: G2P conversion with joint sequence models .................................................................................. 47
3.3 Data .............................................................................................................................................................. 48
3.4 Methods .......................................................................................................................................................... 48
  3.4.1 Step 1: Word origin modeling .................................................................................................................. 49
  3.4.2 Steps 2 and 3: G2P Reweighting of Sequitur output ................................................................................ 51
3.5 Experiments .................................................................................................................................................... 53
  3.5.1 Character-based word origin modeling .................................................................................................... 54
  3.5.2 Self-training for word origin modeling .................................................................................................... 57
  3.5.3 Graphone-based word origin classification ............................................................................................. 59
3.6 Unsupervised word origin clustering ........................................................................................................ 59
3.7 Conclusion ....................................................................................................................................................... 60

4 NAMED ENTITY RECOGNITION FOR DISEASES .......................................................................................... 64
4.1 Related Works ................................................................................................................................................ 65
  4.1.1 Disease tagging with Conditional Random Fields ...................................................................................... 65
  4.1.2 MetaMap ................................................................................................................................................... 66
  4.1.3 Other methods for disease tagging ........................................................................................................... 67
  4.1.4 Gene and protein named entity tagging with morphology ........................................................................ 67
4.2 Data sets for disease mentions ..................................................................................................................... 69
4.3 Methods .......................................................................................................................................................... 70
  4.3.1 Formation of a disease term list from SNOMED ....................................................................................... 70
  4.3.2 Linguistica toolkit ...................................................................................................................................... 71
  4.3.3 Morphological analysis of SNOMED terms with Linguistica .................................................................. 73
  4.3.4 CRF model and features ........................................................................................................................... 76
4.4 Experiments .................................................................................................................................................... 78
4.5 Conclusion ....................................................................................................................................................... 82

5 CONCLUSION ................................................................................................................................................... 84

REFERENCES ...................................................................................................................................................... 86
ABSTRACT

In this thesis we present three projects that focus on natural language processing of named entities in text data, specifically for scenarios and domains where rare and out-of-vocabulary (OOV) words are problematic. First, we present work on the discovery of the language of origin of a named entity. This has applications in speech and language processing tasks, such as text-to-speech synthesis, since language of origin can help predict pronunciation of OOV words. Word origin recognition has also been studied for demographics and life sciences as a component in the collection of ethnicity data. Previous research has applied supervised machine learning methods to automate this task, but this requires a set of hand-labeled training data for each language represented in the model. Hand-labeled data may be expensive to acquire, and additionally the set of origin languages may not be known *a priori*. We consider how active learning can be applied to minimize the amount of annotated data needed to train a successful supervised model. We also apply word origin modeling to grapheme-to-phoneme (G2P) conversion of US surnames, using both supervised and unsupervised approaches.

Finally, we present work in biomedical text mining, where we examine named entity tagging of disease mentions in biomedical text. We extract morphology information from disease terms to be included as features in a Conditional Random Field model. We also show how biomedical disease terms can be decomposed into their component stem parts. Morphology information is acquired with the Linguistica toolkit for unsupervised learning of morphology, which has the advantage that a hand-segmented training set is not required for feature extraction.
ACKNOWLEDGEMENTS

I would like to thank my advisor John Goldsmith for his continuous support, guidance, and encouragement in writing this dissertation. With his help I was able to find what I find to be truly unique and interesting research problems. I would also like to thank members of my committee, Karen Livescu, Andrey Rzhetsky, and Janos Simon, for their input and attention. I am grateful to Gina Levow, my Master’s thesis advisor, and to Anne Rogers, former director of graduate studies, for the role that each played in helping me reach this milestone.

I would like to think my fellow students for their encouragement, advice, and friendship, especially Leandro Cortés, Siwei Wang, Sravana Reddy, Xueyuan Zhou, Ankan Saha, and Irina Matveeva. I am especially grateful for Sravana’s contributions to Chapter 3 of this dissertation, and for discussions with Xueyan and Ankan on the machine learning details of my work.

I would like to take this opportunity to remember Dinoj Surendran who was a valued source of friendship and support while we were students together. During my first few years of graduate school Dinoj gave me advice and inspiration when I often needed it. He was a brilliant and curious scientist, and I wish I had the chance to discuss this dissertation with him.

Finally I would like to thank my parents Raymond and Virginia Waxmonsky whose endless support made this and so many of my dreams possible. And I thank my husband Peter Pawinski for simply everything he’s done for me over the last five years. I could not have done this without the three of you.
LIST OF FIGURES

2.1 Word origin clustering algorithm from [Bose and Sarkar 2009]. In experiments the number of clusters \( N \) is 5, and thresholds \( th_1 \) to \( th_5 \) are \( (0.4, 0.35, 0.3, 0.25, 0.2) \). Alignments are scored using an algorithm that compares character alignments for that pair with translation probabilities learned in the TP-table.

2.2 Cluster-adaptive sampling (CAS), from [Dasgupta and Hsu 2008]. Pseudocode from [Dasgupta and Langford 2009]. \( p_{v,l} \) is the proportion of points in subtree \( T(v) \) at vertex \( v \) having label \( l \). Procedure \( select(P) \) controls hierarchical sampling by choosing the subtree from which a point will be chosen randomly for labeling.

2.3 Accuracy results for supervised classification using Witten-Bell and Additive Smoothing. Results are on six-fold cross-validation with varying training set size.

2.4 Accuracy results for supervised conditional models. Results are on six-fold cross-validation with varying training set size. For comparison the best language model classifier is also displayed: Witten-Bell smoothing over 5-grams.

2.5 Accuracy results for SVM classifiers. “Numeric SVMs” use counts of character n-grams as features. We count all n-grams of order 1 (unigram) through order \( n \), with \( n = 3, 4, 5 \). “Binary SVMs” use binary features (presence vs. absence). For comparison, we also include our best Witten-Bell and Sequential MaxEnt models from previous experiments.

2.6 Accuracy results for cluster-adaptive sampling (CAS), for sample sizes 25-400 (top) and 25-2000 (bottom). Random sampling results are also shown. All data points are averaged over 100 runs.

2.7 Accuracy results for cluster-adaptive sampling (CAS), on held-out development set of 1000 place names. Results are averaged over 100 runs per training set size.

2.8 Accuracy results for cluster-adaptive sampling (CAS) with spectral clustering, for sample sizes 25-500. Two non-spectral systems are included from Figure 2.6: Stand-alone CAS and CAS-LM.

2.9 Accuracy results on US Census data set, for random and cluster adaptive sampling (CAS). Results are averaged over 100 runs per labeling budget size. For series “CAS-MaxEnt”, a circle indicates a significant improvement over “Random Sampling-MaxEnt” based on unpaired t-test at the 5% significance level.
2.10 Accuracy per language class, for 3,000 surname US Census data set. The six most frequent language classes are shown: British Isles (37.6%), German (20.7%), Italian (8.6%), Spanish (6.4%), Slavic (4.9%), French (4.8%). Other (10.4%) includes points that are not in one of these six classes but were assigned a class label. All data points are averaged over 100 runs.

2.11 Percentage of labeled points from each of the six most frequent language classes. “Other and Unlabeled” shows percentage from all other classes excluding these six, plus points that were left unlabeled. Results are averaged over 1000 runs per labeling budget size.

2.12 Conditional entropy of language label given cluster, for Place name and US Census hierarchical clustering trees. Triangle, circle, and square on y-axis mark entropy of unclustered Census, Top-six Census, and Place Name data sets respectively.

2.13 Conditional entropy of language label given cluster, for Place name and US Census hierarchical clustering trees, for pruning size of up to 4,000 clusters.
LIST OF TABLES

2.1 Source and target languages in NEWS shared task on transliteration. Data size is approximate. Examples taken from [Li et al. 2009]. ........................................ 5

2.2 Language classes represented in 8-class set of 10,000 place names from the National Geospatial-Intelligence Agency (NGA). Mayan and Spanish are separated by hand from Mexico country file; for other classes a single language label is assigned by country. ...................................................... 14

2.3 Features used in sequential linear model. $c_i$ represents character at position $i$ relative to the current character position. Features are cumulative, that is the 5-order model contains all features shown above. ........................................ 18

2.4 Confusion matrix for CAS-LM with a sample size of 500 after 6-fold cross-validation. Cell $(i,j)$ shows percentage of $i$ classified as $j$. Bolded diagonal shows accuracy for each class. Accuracy for entire data set is 86.2%. ........... 21

2.5 Hand annotation on test set of 6000 names, after training on 500 names. Column ‘% of data’ gives percentage of 6000 names that belong to this class. Cell $(i,j)$ shows number of points in language $i$ classified as language $j$ and, in parenthesis, percentage of $i$ classified as $j$. Bolded cells on the diagonal show accuracy for class $i$. Overall accuracy is 82.2% ............... 24

2.6 Language classes discovered via hand-labeling of 3000 U.S surnames. In some cases languages are grouped together to find a single language class; grouped languages are then shown in the third column. ........................................ 32

2.7 Confusion matrix for US Census data set. Cell $(i,j)$ shows percentage of points in language $i$ classified as language $j$. Bolded cells on the diagonal show accuracy for class $i$. Overall accuracy is 59.8%. Combined accuracy of 6 most frequent classes listed above is 66.2%. .......................... 33

3.1 G2P word accuracy on US Census test set, by method of word origin modeling (rows) and method of re-weighting G2P output from language-specific models (columns). 67.81% is the baseline accuracy for a single language-independent G2P model trained on the entire training set. Values in parenthesis show difference from baseline. ..................................................... 53

3.2 G2P word accuracy for various weighting methods using a character-based word origin model. For Method B, $\sigma = 0.5$ based on tuning on held out data set. ................................................................. 55

3.3 Examples of names in test set where Baseline and Method B systems output different phonemic transcriptions, and where Method B returns the gold-standard pronunciation. Language labels shown here (Italian, Slavic, Spanish) are computed as $\arg \max_l P(l|w)$ using the character-based word origin model. Phoneme symbols are from an Arpabet-based alphabet, as used in the CMU Pronouncing Dictionary. .................................................. 56
3.4 G2P word accuracy for various weighting methods using a character-based word origin model with self-training. For Method B, $\sigma = 0.55$.  

3.5 G2P word accuracy using a graphone-based word origin model. For Method B, $\sigma = 0.5$ ($\sigma = 0.45$ for "All - Cheat").  

3.6 Accuracy results using unsupervised word origin clustering, with number of clusters $k = 5, 10, 20$ and Methods (A) and (B) of weighting model specific results. As before, accuracy on a language specific baseline is 67.81%; numbers in parenthesis show the absolute change in accuracy from this baseline.  

3.7 G2P word accuracy using an unsupervised character-based word origin model. For Method B, $\sigma = 0.65$ based on tuning on held-out data set.  

3.8 G2P word accuracy using an unsupervised character-based word origin model, with accuracy shown for $k = 20$ clusters labeled as A through T by decreasing training set size. Rightmost column shows three most frequent language classes represented in each cluster, with percentage of training set size shown in parenthesis. Language abbreviations are: 'Br':British, 'Gr':German, 'It':Italian, 'Sl':Slavic, 'Sp':Spanish, 'Fr':French.  

4.1 Sample sentences from Arizona Disease Corpus (AZDC) corpus. Disease mentions are shown in bold.  

4.2 Examples of signatures learned by Linguistica from Tom Sawyer, from [Goldsmith 2001].  

4.3 Subset of suffixes discovered via initial signature analysis of disease word list. Count is the number of words analyzed with this suffix. Ranking is by count.  

4.4 Example of signatures learned from Snomed disease word list.  

4.5 Examples of words where further stem segmentation is possible. The initial signature analysis is shown on the left, and results of further stem segmentation are shown on the right.  

4.6 Examples of morpheme features extracted from the AZDC corpus by segmentation of stems into sub-stems. Column “Example token” shows tokens where this feature occurs. Symbol '+' indicates the location of the stem-suffix split in the initial signature analysis.  

4.7 Feature sets for CRF NER model. Third column shows the token contexts used for each features, relative to the current token. For example, $t_{-1}t_0$ for Part-of-Speech indicates that the current and previous POS tag are combined to a single feature.  

4.8 Results of 10-fold cross-validation on AZDC corpus.  

4.9 Results of 10-fold cross-validation on AZDC corpus, incorporating Linguistica feature sets.  

4.10 Results on CALBC data set, for classification of diseases only.  

4.11 Results on CALBC data set, for classification of diseases and genes/proteins. The first column indicates whether one multi-class NER model was trained (Multi), or if two separate single-class models were trained (Single).
CHAPTER 1
INTRODUCTION

In this thesis we present three projects that focus on natural language processing of named entities in text data. We consider tasks and domains where rare and out-of-vocabulary (OOV) words are especially problematic, and consider how we can recover from data sparsity by extracting character- and morpheme-level features from an OOV word.

First, in Chapter 2 we present work on the discovery of the language of origin of named entities. By language of origin we refer to the language from which a name is derived, which may be different from the language in which it is used. As an example, human readers of English can recognize that the names Pawinski and Schubert have different non-English (Polish and German) origins. The problem of word origin recognition has been previously studied in two domains: speech and language processing, where language of origin can assist in prediction of pronunciation for OOV words, and also demographics and life sciences where ethnicity data is relevant but often unavailable in real-world data sets.

Previous research has applied supervised machine learning to automate the word origin recognition task. These supervised methods require a training set where names are labeled by language, and so the set of language classes must be known before training begins. This training data may be expensive to acquire, and in a practical application the classes of languages may not be known a priori. We apply active learning to minimize the amount of manual annotation needed to train a successful supervised model. Specifically, we apply cluster-adaptive sampling (CAS) as presented in [Dasgupta and Hsu 2008]. Our goal is to determine if, for the same number of labeled names, a data set acquired via CAS can train a more accurate supervised classifier than one trained from data chosen at random for labeling. We apply this method to two data sets: one of geographic place names and another of surnames selected from US Census data. The place name set is composed using a predetermined list of languages, while the US Census data set is created by selecting names at random from a single list for subsequent manual labeling. One contribution of this work is to consider the word origin problem for data where variation in language origin occurs naturally, as in US census data.
In Chapter 3 we present an application of word origin modeling to grapheme-to-phoneme (G2P) conversion, a step in text-to-speech synthesis and automatic speech recognition where an orthographic representation of a name is translated to a phonemic representation. Motivated by the fact that word origin affects pronunciation, our goal is to determine whether we can improve G2P conversion by training on language-specific data sets of grapheme-phoneme name pairs, rather than a single data set where word origin is unknown. A trivial approach to do this is to segment G2P training data by word origin using methods discussed in Chapter 2 and then train multiple language-specific G2P models. However, we find that this can actually decrease performance with our US Census data set since word origin modeling is imperfect and the resulting language-specific training sets are much smaller, leading to problems of data sparsity. To overcome this we train one language-independent model in addition to multiple language-specific models and use word origin posteriors to smooth results. We investigate multiple approaches to smoothing and word origin modeling, and also present an unsupervised approach where no hand-labeled word origin data is required. With smoothing, both the supervised and unsupervised methods are able to improve on a single language-specific G2P model.

Our third project, presented in Chapter 4, is in the area of biomedical text mining. This domain has a large, diverse, and growing technical vocabulary which adds a level of complexity to information retrieval and text mining tasks. Names of genes and proteins are regularly introduced into the vocabulary and, additionally, natural lexical variation in spelling and morphology mean that existing concepts appear in new and unseen forms. We examine named entity tagging of disease mentions in biomedical text, which is necessary to map terms to a database for information retrieval purposes. We show how morphology features can be extracted from disease terms and included as features in a Conditional Random Field model. We also show how biomedical disease terms can be decompounded into their component stem parts. Morphology information is acquired with the Linguistica toolkit for unsupervised morphology discovery, which has the advantage that a morphologically segmented training set is not required for feature extraction. Our results show that structured morphological features can improve performance when other word-internal features are not present; however, that benefit is generally lost as other word-internal features, such as simple character n-grams, are added.
CHAPTER 2
WORD ORIGIN RECOGNITION FOR NAMED ENTITIES

As readers we are able to associate foreign names with their language of origin, even in the case of words we have not seen before. As an example, many English speakers could recognize without prior knowledge that Makowski and Masiello are Polish and Italian surnames. This task, which we refer to as word origin recognition, has been automated as a machine learning problem, and current supervised methods perform very well.

Word origin recognition has applications in multiple fields. Within speech and language processing it has been studied for transliteration, or the character-level translation of out-of-vocabulary (OOV) words between alphabets, and to text-to-speech (TTS) generation when dictionary pronunciation is unavailable. Word origin recognition also has been studied for life science fields, such as public health and demographics, where ethnicity information is of interest to researchers but not always reliably available. Here we review works in various fields where a language or ethnicity is assigned to a data point.

We then present our proposed work in word origin recognition. The majority of works presented here are in a supervised machine learning framework, where language-specific training data is available to label unseen names. This assumes that the set of possible language labels is known in advance, which requires some prior knowledge about the data. Our goal is to address the word origin problem when the set of language labels is not known a priori, or when training data is not readily available. We assume that manual labeling is available but costly; as such we focus on active learning solutions which use both labeled and unlabeled data to improve clustering.

2.1 Related works in speech and language processing

Word origin influences the relationship between orthography and pronunciation. In transliteration, knowledge of the language of origin in a word can affect character choice in the target language. In Hindi-to-English transliteration, the character ‘क’ (‘ka’) is often transliterated as ‘q’ in Arabic-origin words (Qatar, Sadiqah) and ‘k’ in Hindi-origin words (Kapoor, Panchkula). Likewise for
TTS processing, word origin can influence the pronunciation of unseen words. For example *sch-* is likely to be pronounced as /sh/ for German-origin names (*Schoenenberg*) and /sk/ for Italian-origin words (*Schiavone*), as noted in [Llitjós 2001]. Below we review works in speech and language processing that seek to take advantage of these language-specific patterns.

**Word origin recognition on place names**

[Chen and Maison 2003] test source-channel models with various smoothing algorithms for language identification on place names. Data is obtained from the National Geospatial-Intelligence Agency (NGA)\(^1\), where extensive lists of place names are available organized by country. The best results are from a maximum entropy (ME) n-gram model smoothed with a Gaussian prior [Chen 2000]. This model uses *cross-model* smoothing, where n-grams are smoothed toward a data set of all languages combined, rather than toward shorter n-grams. This system showed 70.0% accuracy on a 48-way test set. Kneser-Ney (KN) smoothing performed comparably, at 69.1%.

Source channel models are compared with maximum entropy (ME) models that directly model conditional probability of the language given the input. These ME models use binary features that indicate if an n-gram from the training data is present in the input word; all n-grams up to a certain length can be included, and the authors do not specify what order of \( n \) they used. In experiments on a five-language test set, source-channel models are shown to have slightly higher accuracy than ME models, 86.8% vs. 85.6% for the best case in each category. This work also shows that a place name data set can be successfully applied to a person-name task.

This work uses a unique approach to clustering country data to build language training files. A minimal set of “basis” countries is selected that best represents the set of languages in the evaluation set. Selection is done by creating an initial 90-class language model, from 90 country files, and then iteratively removing the country with the greatest error. An intermediate set of 48 country files is chosen, each of which is assumed to represent its own *pseudo-language*. Other countries that can be closely grouped with one member of this set of 48 are added to the data set, specifically \( C' \) is grouped with basis country \( C \) if \( p(C'|C) > 0.70 \). These groupings include Germany/Austria, Brazil/Portugal, Egypt/SaudiArabia/Yemen, and Bosnia-Herz./Croatia/Yugoslavia.

This clustering approach has two undesirable consequences. First, minority languages such as Catalan cannot be included in the model since there is no corresponding country file. Additionally,

\(^1\) [http://earth-info.nga.mil/gns/html/namefiles.htm](http://earth-info.nga.mil/gns/html/namefiles.htm)
Table 2.1: Source and target languages in NEWS shared task on transliteration. Data size is approximate. Examples taken from [Li et al. 2009].

<table>
<thead>
<tr>
<th>Source</th>
<th>Target</th>
<th>Size</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>Chinese</td>
<td>30K</td>
<td>Timothy → 帝莫西</td>
</tr>
<tr>
<td>English</td>
<td>Japanese Katakana</td>
<td>25K</td>
<td>Harrington → ハリントン</td>
</tr>
<tr>
<td>English</td>
<td>Korean Hangul</td>
<td>7K</td>
<td>Bennett → 베테트</td>
</tr>
<tr>
<td>Japanese name (English)</td>
<td>Japanese Kanji</td>
<td>20K</td>
<td>Akihiro → 秋宏</td>
</tr>
<tr>
<td>English</td>
<td>Hindi</td>
<td>15K</td>
<td>San Francisco → सैन फ्रांसिस्को</td>
</tr>
<tr>
<td>English</td>
<td>Tamil</td>
<td>15K</td>
<td>London → ロンドン</td>
</tr>
<tr>
<td>English</td>
<td>Kannada</td>
<td>15K</td>
<td>Tokyo → トーキョー</td>
</tr>
<tr>
<td>English</td>
<td>Russian</td>
<td>10K</td>
<td>Moscow → Масква</td>
</tr>
</tbody>
</table>

data from a multi-lingual list (e.g. Belgium) is removed from the training set since a uniform language label cannot be assigned. It’s interesting to note that the model selects Brazil, Mexico, and Peru as a basis for Spanish and Portuguese speaking countries, but Mexico and Peru are placed in separate clusters. We presume this is because of the presence of Amerindian language place names, in which case there would be no “pure” Spanish data set in the model.

**Named Entity Workshop, Shared Task in Machine Transliteration**

Word origin classification appears in the 2009 Named Entity Workshop (NEWS) Shared Task [Li et al. 2009], which evaluates machine learning approaches to transliteration. Training data is in the form of word pairs, where each pair includes a term in the source language and its transliterated form in the target language. Eight language pairs are distributed (Table 2.1). Two papers from the NEWS use word origin to improve transliteration results, which we describe below.

[Khapra and Bhattacharyya 2009] classifies words as Indic or Western by building two character-based n-gram language models. An initial set of 3000 words is labeled by hand as Indic or Western, and then a larger set is bootstrapped by automatically labeling all remaining training data. Test data is classified as Indic or Western and then passed to separate CRF models for transliteration. For word origin classification, the authors report 97% accuracy on this test set. We note that the decision to treat origin classification as a binary problem (Indic vs. Western) creates a simpler problem where higher accuracy would be expected compared to the multiclass case.
Algorithm 1 Classify the Corpus

$C_1 = \text{All pairs in training data}$

for $i = 1$ to $N$ do
  Set Alignment threshold, $th_i$ for Class $C_i : th_i \leq th_{i-1}$
  while size of Class $C_i$ does not decrease do
    Compute translation probabilities (TP)-table for $C_i$, using IBM model 1
    for each parallel word pair $<I_j, E_j>$ in $C_i$ do
      Compute character Alignment of word pair
      Compute Score of Alignment using TP-table
      if $Score < th_i$ then
        Move the word pair to class $C_{i+1}$
      end if
    end for
  end while
end for

Figure 2.1: Word origin clustering algorithm from [Bose and Sarkar 2009]. In experiments the number of clusters $N$ is 5, and thresholds $th_1$ to $th_5$ are (0.4, 0.35, 0.3, 0.25, 0.2). Alignments are scored using an algorithm that compares character alignments for that pair with translation probabilities learned in the TP-table.

[Bose and Sarkar 2009] also use word origin information in transliteration. This work does transliteration through character alignment based on IBM model 1, a method of word alignment for machine translation. To allow for variation in alignment probabilities for words of different origin, five different transliteration models are trained, each from its own partition of the training data. Partitioning occurs in a preprocessing step, where training data is clustered into five classes in an unsupervised manner. The objective of clustering is to find classes with consistent character alignments (pseudocode is shown in Figure 2.1). To do transliteration, named entities in the test set are assigned to one of these five classes based on n-gram similarity and then transliterated with the appropriate model. The authors do not provide examples of the resulting five clusters so we are not able to determine how well each cluster corresponds to a unique language. However we note that the training data for this task often includes multiple languages other than the target and source language for the transliteration task. For example the English-Hindi set includes multiple Arabic- and French-origin words.
Other works for word origin recognition

[Chen et al. 2006] uses word origin information to improve letter-to-sound (LTS) generation in speech, specifically on person names. This work uses syllable-based letter clusters (SBLC) as units to build n-gram language models for each of the candidate languages. Letter-based models were also considered, and it was found that SBLCs improved performance for English and German but not for French and Portuguese. Combining both letter- and syllable-based methods with AdaBoost shows further improvements, with a final accuracy of 78% on a test set that includes these four language classes.

[Surana and Singh 2008] presents a method for named entity transliteration that uses methods from [Singh 2006], an earlier work that studies the task of classifying the language and character encoding of a whole text document (rather than a single word). [Singh 2006] compares character n-gram distributions between a reference and input document, and the input document is assigned the language and encoding of the most similar reference document. Multiple measures of similarity are studied, all of which use character n-gram features. The author finds the best results with mutual cross entropy (MCE), the sum of the cross entropy of each distribution with the other. For probabilities \( p \) and \( q \) MCE is computed as:

\[
MCE = \sum_x (p(x) \times \log q(x) + q(x) \times \log p(x))
\]

The later work on transliteration, [Surana and Singh 2008], uses mutual cross-entropy to classify words as Indic or Western; accuracy results for the word origin subcomponent are not reported.

[Llitjós and Black 2001] and [Llitjós 2001] also use word origin information to improve TTS pronunciation of unseen person names. Here letter language models (LLM) output the probability of a language given the word, \( P(L|input) \), for each of 26 languages that were chosen by hand by inspecting the training data. Classification follows the methods presented in [Cavnar and Trenkle 1994] for language and subject classification of an entire document. This work computes:

\[
P(L|input) = \frac{\sum_{\text{input-trigram}} C(\text{trigram})}{\sum_{\text{input-trigram}}' C(\text{trigram}')} P(\text{trigram}|L)
\]

where \( C(X) \) is the count of \( X \) in the input word. This models uses only character trigrams as features, as this was found to be most successful in earlier works. These works are [Vitale 1991],
a letter-to-sound (LTS) system that is similar to [Llitjós 2001] but relies heavily on hand-crafted rules, [Church 2000] which predicts stress in loan words in English, and [Sproat et al. 1999] which classifies “non-standard” words in text such as numbers, abbreviations, URLs, etc. [Llitjós 2001] does not directly test the word-origin component. Rather, language probabilities are included in LTS decision tree models for word pronunciation, and classifiers that use word origin information show a 7.6% increase in accuracy over those that do not.

[Bhargava and Kondrak 2010] applies Support Vector Machines (SVMs) to word origin recognition using character n-grams as features. Experiments use two data sets: a list of international soccer player names and a list of Chinese, English, and Japanese surnames. Experiments show that SVMs outperform language models; details of language model smoothing is not provided. An additional experiment applies word origin to machine transliteration for a set of English and Hindi names. Results show a small drop in accuracy which the authors attribute to reduced training data after separation into two language-specific sets.

[Hagiwara and Sekine 2011] applies word origin detection to transliteration. Here Western names in the Latin alphabet are translated into a phonetic representation in Japanese. This work is unique in that it does not assume a prior set of labeled languages and classes. Instead string substitutions from the source to target language are conditioned on a latent variable $z$, which corresponds to sets of target and source word pairs with similar transliteration characteristics which may naturally relate to gender as well as word origin. Expectation Maximization is used to compute $z$ to maximize its likelihood given the transliteration training set. Performance is evaluated by applying transliteration to an information retrieval task, where the goal is to rank target word $t$ from source $s$ based on model estimations of $P(t|s)$. Experiments compare four systems:

1. A baseline alpha-beta model based on [Brill and Moore 2000] that does not use any language origin factors,
2. A model that conditions transliteration on the single most likely language of origin,
3. A model that weights probabilities based on language class probabilities, and
4. A model based on latent classes.

Models 2 and 3 estimate $P(lang|s)$ using a 3-gram language model with Witten-Bell smoothing. Results show that Model 4 using latent variables is comparable to soft-clustering in Model 3, and both outperform 1 and 2. Two data sets are used for evaluation, one taken from lists of personal
names and another from Wikipedia articles, where phrases such as “of Germany” in the first sentence of the article are used as language indicators. Each set is composed by building and then merging language-specific name lists; the first set uses three European languages and the second uses five.

2.2 Related works in life sciences and ethnicity classification

In Section 2.1 we present works that study the word origin problem to improve speech and language understanding. Here we look at research in health sciences, demographics, geography, marketing, and other life sciences. These works classify a person record, not a textual named entity, and labeling is not for language but rather for ethnicity, a broader category that includes race, religion, birthplace, and nation of origin. Ethnicity classification is not strictly a problem of labeling unseen names, since common first and last names can be recognized and mapped to an ethnicity via lookup on reference lists. In fact many works in this section use only name lists and do not apply any sub-word information (prefix, suffix, n-grams) as done in Section 2.1. Sub-word features are leveraged in some works, but in these cases word origin labeling is only one component in a larger ethnicity classification system.

[Mateos 2007] presents a survey of 13 works in the fields of public health, population genetics, linguistics, and statistics that study name-based ethnicity classification. Data is taken from sources that include hospital records, birth and death registries, and census records. Most of these works use no sub-word features, however two from the UK, [Cummins et al. 1999] and [Harding et al. 1999], use a software program, Nam Pehchan, which classifies names as South Asian or non-South Asian. When a name is not on a reference list then this software uses name stems, generally the first five characters of the last name. [Cummins et al. 1999] found that Nam Pehchan has a high false positive rate (37%) for detecting South Asian names.

[Mateos 2007] notes that none of these 13 studies attempts to classify an entire population into a complete set of ethnic groups within a country; rather, methods start with a reference list and then classify a target population into a predefined set of ethnicities. Only one of the works [Lauderdale and Kestenbaum 2000] did multi-class classification; this work builds name lists for six different Asian ethnicities (Chinese, Japanese, Filipino, Korean, Asian Indian, and Vietnamese) using U.S. Social Security birth and death records. This work also shows that ethnicity and name origin is not a simple one-to-one map, as some surnames may be associated with multiple ethnicities.
[Mateos 2009] presents a geographic study of the foreign population in Madrid, based on first and last name information extracted from a telephone directory. Data is used to study population segregation using metrics presented in [Massey and Denton 1988]. Last names are classified by ethnicity using methods presented in [Mateos et al. 2007], which uses a last name dictionary and augments this with a text-mining clustering approach to label unseen surnames. In this approach, last names are sorted in alphabetical and reverse-alphabetical order, and then an ethnicity is assigned by choosing a distance-weighted score of the 20 surrounding names in each list. [Mateos et al. 2007] is unique from previous works in that it attempts to classify the entire population of the UK by ethnicity, although surnames with fewer than three occurrences are excluded from the data set. Records are sorted into 178 groups in a Cultural Ethnic and Linguistic (CEL) taxonomy that was organized “in a way that maximizes each group’s homogeneity along the four dimensions of human origins (geography, religion, ethnicity and language) .”

[Ambekar et al. 2009] presents an ethnicity classification system for person names. An HMM model is presented which allows for transitions in ethnicity label between the first, middle, and last name. This work also organizes names into a hierarchy of ethnicities in order to overcome data sparsity and to allow for classification at different levels of granularity. Data is obtained from publicly available sources including Wikipedia’s extensive network of article classification which can give clues to the ethnicity of a subject (e.g. Canadians of Welsh Descent).

### 2.3 Methods

Our goal in this project is to label unseen named entities by language of origin when a gold-standard training set is incomplete or unavailable, and when we do not have prior knowledge of the set of languages represented in a data set. This may be the case when immigration patterns or other phenomena introduce unseen names into a population, and when transliteration of names from non-Latin scripts introduces spelling variations. We assume that labeling is time-consuming and expensive, and so our question is how a limited labeling budget can be spent to best classify a population of named entities by language of origin.

First we test methods on data sets built from country place name files, as done in [Chen and Maison 2003]. Next, we apply our methods to lists of person names using surname lists made available by the US Census Bureau. For sampling and evaluation, we will label data manually by inspection and with references such as the *Dictionary of American Family Names* [Hanks 2003],
Algorithm 2 Cluster-adaptive sampling for clustering

Input: Hierarchical clustering tree $T$ of $n$ data points

For each node $v$ in $T$ maintain:
(i) majority label $L(v)$
(ii) empirical frequencies $\hat{p}_{v,l}$ for each label $l$
(iii) confidence interval $[p_{lb}^{v,l}, p_{ub}^{v,l}]$ on lower and upper bounds of $p_{v,l}$

Initialize: pruning $P = \text{root}; L(\text{root}) = l_0$

for $t = 1, 2, 3, \ldots$ until budget runs out do
  $v = \text{select}(P)$ ▷ select node from pruning
  Pick a random point $z$ in subtree $T_v$
  Query for $L(z)$, class label of $z$
  Update empirical counts and probabilities $\hat{p}_{u,l}$ for all nodes $u$ on path from $z$ to $v$
  Choose best pruning and labeling $(P', L')$ of subtree $T_v$
  $P = (P \setminus \{v\}) \cup P'$ ▷ update pruning and class labels
  $L(u) = L'(u)$ for all nodes $u$ in $P'$
end for

for each $v \in P$ do
  for each point $z$ in subtree $T(v)$ do
    $L(z) = L(v)$ ▷ copy labels from node to leaves
  end for
end for

Figure 2.2: Cluster-adaptive sampling (CAS), from [Dasgupta and Hsu 2008]. Pseudocode from [Dasgupta and Langford 2009]. $p_{v,l}$ is the proportion of points in subtree $T(v)$ at vertex $v$ having label $l$. Procedure select($P$) controls hierarchical sampling by choosing the subtree from which a point will be chosen randomly for labeling.

online immigration records, and search engine results. We choose a population where references will be available for evaluation; however we reiterate that our goal is to develop methods that can be applied when such resources are limited or unavailable.

Below we present our methods to label a population of names. We first present Cluster Adaptive Sampling (CAS), a published approach that exploits a hierarchical clustering of a data set to guide active sampling so as to improve data partitioning and classification. We then present our method to incorporate spectral clustering into the CAS framework.
2.3.1 Cluster-adaptive sampling

[Dasgupta and Hsu 2008] presents cluster-adaptive sampling (or hierarchical sampling), an active learning approach that uses the cluster structure of the input data to infer a complete class labeling. This method starts with a hierarchical clustering of the input in the form of a binary tree, and the final output is a pruning of this tree, i.e., a subset of nodes \( v_1, \ldots, v_m \) such that the subtrees rooted at these nodes are disjoint and together cover all data points. Each node in this pruning corresponds to a cluster that is assigned a single class label. The algorithm proceeds iteratively; at each iteration points are queried to be presented to a labeler. The objective is to find a pruning with “pure” subtrees that have a consistent class label. To achieve this, “impure” subtrees are replaced with purer child subtrees based on information acquired from labeling. Various options are presented for sampling, and it is suggested to sample so as to favor impure subtrees where the class labels are most uncertain. Pseudocode is shown in Algorithm 2.

This hierarchical sampling method does not require prior knowledge of the label set, and the number of classes \( k \) is not set in advance. Experiments show that this method is able to identify rare classes that may be excluded if points are queried randomly for labeling. In one experiment this method identifies a class that covers 0.014% of the input data set; here hierarchical sampling discovers all seven classes in 880 queries versus 8000 for random sampling.

2.3.2 Spectral clustering

We will also consider spectral clustering methods, which detect structure in the data by decomposing a pairwise affinity matrix \( A \), where \( a_{ij} \) is some measure of similarity between points \( i \) and \( j \). This corresponds to a similarity graph representation of the data, where each point is a vertex, \( A \) represents edge weights, and a greater weight \( a_{ij} \) indicates greater similarity between points \( i \) and \( j \). Our steps are as follows:

1. For \( n \) data points, form the \( n \times n \) similarity matrix \( A \).

2. Construct the normalized graph Laplacian \( L \) from \( A \).

3. Find the \( k \) largest eigenvectors of \( L \), \( x_1 \ldots x_k \) and the corresponding eigenvalues \( \Lambda = \lambda_1 \ldots \lambda_k \). Construct the \( n \times k \) matrix \( X = [\lambda_1 x_1, \ldots, \lambda_k x_k] \).
4. **Do hierarchical clustering on matrix** $X$, treating each row as a point in a $k$-dimensional space. Output is a clustering tree $T$.

5. Perform Cluster-Adaptive sampling with hierarchical clustering tree $T$.

6. Label entire data set using Stand-alone CAS or supervised classification (CAS-LM or CAS-MaxEnt).

The similarity matrix $A$ is computed as a $k$-nearest neighbor (knn) graph, where vertex $v_i$ is connected with $v_j$ if $v_i$ is among the $k$ nearest neighbor of $v_j$ or $v_j$ is among the $k$ nearest neighbors of $v_i$.

Similarity matrix $A$ is computed from character n-grams using the same feature matrix used in CAS. To normalize $A$ in Step 2 we use the symmetric graph Laplacian:

$$L_{sym} = I - D^{-1/2}AD^{-1/2}$$

(2.1)

After $T$ is constructed in step 5 we label points in the same manner as standard (non-spectral) CAS. In our experiments we can compare spectral and non-spectral hierarchical clustering as input to CAS.

### 2.4 Experiments with Supervised classification

In this section we present experiments with supervised classification of word origin, where gold standard training data is available and the language classes in training and test sets are known in advance. [Chen and Maison 2003] presents a comparison of conditional and generative models for this supervised task, however this work uses a very large data set, with 1.7 million names in total and individual language sets ranging in size from 5,404 to 154,510 unique names. In contrast we anticipate much smaller training sets, on the order of a few hundred or thousand, since we specifically wish to address the situation when training data is not available in advance and cannot be easily acquired. For this reason we present our own comparison of some supervised classifiers appropriate for the active learning framework. In later sections these supervised classifiers will be used as a black box to extend labels acquired during active learning to an entire data set that is initially unlabeled.
<table>
<thead>
<tr>
<th>Language</th>
<th>Source Country</th>
<th>Count</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>UK</td>
<td>5000</td>
<td>Arlingham, Cronton, Corby</td>
</tr>
<tr>
<td>Russian</td>
<td>Russia</td>
<td>2000</td>
<td>Lychnoye, Novoberezovo, Yeliseyevo</td>
</tr>
<tr>
<td>Arabic</td>
<td>Saudi Arabia</td>
<td>1000</td>
<td>Buwayqi, Subaykhan, Misayllan</td>
</tr>
<tr>
<td>Mayan</td>
<td>Mexico (Yucatan)</td>
<td>900</td>
<td>Choyob, Citilcum, Xtuc</td>
</tr>
<tr>
<td>German</td>
<td>Germany</td>
<td>500</td>
<td>Brosen, Wulferstedt, Ihmert</td>
</tr>
<tr>
<td>Japanese</td>
<td>Japan</td>
<td>500</td>
<td>Omagari, Matsugasaki, Tobata</td>
</tr>
<tr>
<td>Polish</td>
<td>Poland</td>
<td>50</td>
<td>Wygielzow, Smrek, Biedaszki</td>
</tr>
<tr>
<td>Spanish</td>
<td>Mexico</td>
<td>50</td>
<td>Bernardo, Luna, Crucero</td>
</tr>
</tbody>
</table>

Table 2.2: Language classes represented in 8-class set of 10,000 place names from the National Geospatial-Intelligence Agency (NGA). Mayan and Spanish are separated by hand from Mexico country file; for other classes a single language label is assigned by country.

2.4.1 NGA Place Name Data

Experiments in this section are done with lists of geographic place names. Data was obtained from the National Geospatial-Intelligence Agency (NGA), where extensive lists of place names are available organized by country. Place names include cities and towns; administrative regions such as provinces and territories; streets, roads and highways; and hydrographic features such as rivers, lakes, and bays. NGA files include some terms written in a non-Latin character set; however we filter out these terms for our experiments. We also remove accents and exclude names with special characters, as their presence can greatly simplify the language recognition problem. NGA files also include latitude and longitude coordinates that are used for preprocessing of some files.

We construct a single data set using place names from eight language classes, as shown in Table 2.2. Mayan and Spanish sets are both taken from the Mexico country file; first we select a list of names from then Yucatan region of Mexico, where the majority of place names are Mayan in origin, and then separate this set by hand into Mayan and Spanish sets. For the other six language classes (English, Russian, Arabic, German, Japanese, and Polish) we choose a “pure” country file where this language is the national and historical language. The UK file is also filtered by longitude and latitude to approximately exclude Gaelic and Welsh place names from Scotland and Wales. Language priors are chosen by hand and range from 0.5% to 50% of the entire 10k set to examine how our supervised and semi-supervised methods perform at both rare and frequent classes.

From examining NGA data we find that multi-word place names often include language-specific word that would make language labeling trivial. For example, *Saint* and *Bay* in *Saint Ninians Bay* are strong cues of an English place name; *San* in Spanish is another example of such a word. These terms appear repeatedly in NGA files and would greatly facilitate supervised classification. Since our goal in this project is to focus on word-internal information and the classification of unseen words, we remove multi-word places names from our training and test sets, thus filtering these repeating language-indicator words.

All countries show in Table 2.2, including Mexico, were discovered to be “language pure” in [Chen and Maison 2003], meaning that each country list is assumed to represent a single *pseudo-language*, and each pseudo-language can be mapped to an appropriate natural language. In reality we recognize that our language sets may not be 100% “pure”, however using place name lists provides us with a large and automatically labeled multi-language data set. In Section 2.4 we examine human performance for language classification on this eight-class set.

Supervised classification experiments are done with 5-fold cross validation. For each fold 2000 names are used for training, 6000 for test, and 2000 are held out for parameter tuning. We use training set size $N$ ranging from 100 to 2000. When $N < 2000$, $N$ items are selected items from the shuffled training set and the remaining $(2000 - N)$ items are discarded, and test and held-out sets are unchanged.

### 2.4.2 Classifiers

Here we present both generative source-channel and conditional models for language classification. All models use character-based features, and we assume that all data is written in a common character set.

**Source-channel n-gram models**

As in [Chen and Maison 2003] we build source-channel language models that use character n-grams to estimate $P(w|L)$, the probability of a word $w$ given a language $L$. The language of origin of $w$ is then chosen as:

$$L = \arg\max_L P(L|w) = \arg\max_L P(w|L)P(L) \quad (2.2)$$
While conventionally used to estimate a distribution over sentences or sequences of words, language models can be applied to sequences of characters for our current problem. For word \( w \) composed of characters \( c_1 \ldots c_n \), we express the probability that \( w \) appears in language \( L \) as:

\[
P(w|L) = p_L(c_1) \times p_L(c_2|c_1) \times \cdots \times p_L(c_n|c_1 \ldots c_{n-1})
\]  

(2.3)

where probabilities \( p_L \) are estimated from the training data. N-gram language models are widely used and make the Markov assumption that \( l_i \) only depends on the \( n-1 \) most recent letters:

\[
P(w|L) = \prod_{i=1}^{k} p_L(c_i|c_{i-n+1} \ldots c_{i-1})
\]  

(2.4)

Maximum likelihood (ML) estimates of \( p_L(c_i|c_{i-n+1} \ldots c_{i-1}) \) can be obtained directly from training data counts. However some smoothing is necessary to prevent over-fitting to the training set and specifically to accommodate for unseen n-grams.

[Chen and Maison 2003] applies two smoothing techniques: Witten-Bell [Bell et al. 1990] and modified Kneser-Ney (KN) [Chen and Goodman 1998]. Witten-Bell smoothing presents a recursive model that, for order \( n \), linearly interpolates between ML estimates at order \( n \) and smoothed estimates at order \( n-1 \):

\[
p_{WB}(c_i|c_{i-n+1}) = \lambda c_{i-1} c_{i-n+1} p_{ML}(c_i|c_{i-n+1}) + (1 - \lambda c_{i-1} c_{i-n+1}) p_{WB}(c_i|c_{i-n+2})
\]  

(2.5)

Here the higher order model is used with probability \( \lambda c_{i-1} c_{i-n+1} \) and the lower-order model with probability \( 1 - \lambda c_{i-1} c_{i-n+1} \).

Parameters are chosen so that the higher order model is used when \( c_{i-n+1}^{i-1} \) occurs in the training data and the lower order model when it does not. We choose \( \lambda \) such that:

\[
1 - \lambda c_{i-1} c_{i-n+1} = \frac{N(c_{i-n+1}^{i-1})}{N(c_{i-n+1}^{i-1}) + \text{count}(c_{i-n+1}^{i-1})}
\]  

(2.6)

where \( N(c_{i-n+1}^{i-1}) \) is the number of unique words that follow \( c_{i-n+1}^{i-1} \) and \( \text{count}(c_{i-n+1}^{i-1}) \) is the number of times that \( c_{i-n+1}^{i-1} \) is observed in the training corpus. That is, \( 1 - \lambda \) is the probability that a character not already observed in the training data after \( c_{i-n+1}^{i-1} \) is observed in that context.

16
Substituting with Equation 2.5 we have:

\[
p_{WB}(c_i|cC) = \frac{\text{count}(c_{i-n+1} \ldots c_i) + pWB(c_i|c_{i-n+2})N(c_{i-n+1}^{l-1})}{N(c_{i-n+1}^{l-1}) + \text{count}(c_{i-n+1}^{l-1})}
\]  

(2.7)

Modified Kneser-Ney (KN) smoothing is also presented in [Chen and Maison 2003]; however we find that low n-gram counts make the method unfeasible for smoothing in our training data, where a single class may only occur a handful of times. While our training set is artificial, we do expect that in an active learning framework with a limited sampling budget there will be similar problems of data sparsity and rare classes.

As an alternate source-channel method we also consider additive smoothing where a constant \( D \) is added to each n-gram count:

\[
p(c_i|c_{i-n+1} \ldots c_{i-1}) = \frac{\text{count}(c_{i-n} \ldots c_i) + D}{\text{count}(c_{i-n} \ldots c_{i-1}) + D|V|}
\]  

(2.8)

where \( V \) is the vocabulary. Here \( V \) is the observed characters plus start and end symbols. We chose \( D = 0.1 \) based on tuning on our held-out set.

We use the SRILM toolkit to generate LMs for each language class in our place name set.

**Maximum Entropy models**

We next consider conditional Maximum Entropy (MaxEnt) models using character n-grams as binary features. MaxEnt models have the advantage of directly modeling the conditional probability \( P(L|w) \). We represent a word as a vector of binary features, where each feature indicates the presence or absence of an n-gram found the training set. We include n-grams of length one up to \( n = 3, 4, 5 \), that is, for \( n = 3 \) we include character unigrams, bigrams, and trigrams as features. For all n-grams \( i = 1 \ldots F \) in the training set we have:

\[
f_i(w_j, l_k) = \begin{cases} 
1, & \text{if n-gram}_i \text{ present in word } w_j \\
0, & \text{otherwise}
\end{cases}
\]  

(2.9)

where \( l_k \) is the language class label, and

\[
P(l_k|w_j) = \frac{\exp \sum_i \lambda_i f_i(w_j, l_k)}{Z}
\]  

(2.10)
Table 2.3: Features used in sequential linear model. $c_i$ represents character at position $i$ relative to the current character position. Features are cumulative, that is the 5-order model contains all features shown above.

where $Z = \sum_j \exp \sum_i \lambda_i f_i(w_j, l_k)$ is a normalization factor to ensure that probabilities sum to one and $\lambda_i = \lambda_1 \ldots \lambda_F$ are the parameters of the model.

We implement our MaxEnt model with the Mallet toolkit [McCallum 2002]. Mallet applies smoothing with a Gaussian prior [Chen 2000] to prevent against over-fitting to the training set. Tuning is done on held-out development sets.

**Sequential models**

In our previous MaxEnt model we test for the presence or absence of a given n-gram. This fails to account for word length and repeated n-grams within a word. To overcome this we present a sequential conditional model of $P(L|w)$. Here we use “sequential” somewhat loosely; we model the word as a sequence of character positions dependent on a single language tag. Feature functions are defined for a character position rather than over the whole word. Specifically, rather than check for the presence or absence of an n-gram with the word we test for n-gram character contexts at character position $m = 1 \ldots M$ with word $w_j$ of length $M$. These contexts correspond to the presence or absence of an n-gram at a given character position $m$ in the word.

$$f_i(c_m, l_k) = \begin{cases} 
1, & \text{if context } i \text{ is present at character position } m \\
0, & \text{otherwise} 
\end{cases} \quad (2.11)$$

where $l_k$ is the language class label. Table 2.3 shows feature functions for this model.

Then for $w_j = c_i \ldots c_M$:

$$P(l_k|w_j) = P(l_k|c_i \ldots c_M) = \frac{\exp \sum_m \sum_i \lambda_i f_i(c_m, l_k)}{Z} \quad (2.12)$$
where \( Z = \sum_j \exp \sum_m \sum_i \lambda_i f_i(c_m, l_k) \) is a normalization factor.

In practice we can implement this model as a CRF, where a language label is applied to each character position. The result is that instead of a single language label we have a chain of language tags, one for each character in the word. Smoothing on state transitions may allow a single word to be assigned multiple language labels, and so we may want to add the constraint that all language labels be equal within a word. However in practice we find only a small handful of words where such a state transition occurs. For these cases we take the language label of the last character position.

We implement our model with the CRF++ package with \( L^2 \) regularization. Tuning of the smoothing parameter \( c \) is done on the held-out development set.

**Support Vector Machines**

We next apply Support Vector Machines (SVMs) to our classification task. The SVM outputs a binary classifier that labels data points as +1 or -1 based on a separating hyperplane. The SVM seeks to maximize the margin, or distance, between the hyperplane and the nearest training examples. Given training data \( x_1 \ldots x_l \in R^n \), the SVM has the following objective function:

\[
\arg\min_{w, b, \epsilon} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \epsilon_i \quad C > 0
\]

subject to \( y_i (w^T \phi(x_i) + b) \geq 1 - \epsilon_i \) \( \epsilon_i \geq 0, i = 1 \ldots l \)

(2.13)

Here \( C \) is a penalty parameter and \( \epsilon_i \) are the slack variables that allow for some degree of misclassification on training vector \( x_i \). Function \( \phi \) maps \( x_i \) into a higher dimensional space where the separating hyperplane is constructed, and \( K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j) \) is called the kernel function. Use of a kernel function allows for a non-linear hyperplane in the input feature space. We use the radial basis function (RBF) kernel with parameter \( \gamma \):

\[
K(x_i, x_j) = \exp(-\gamma||x_i - x_j||^2), \gamma > 0
\]

(2.14)

We implement SVM classifiers with the LIBSVM pacakge [Chang and Lin 2001]. LIBSVM uses the “one-against-one” or pairwise method for multi-class classification, where, for \( K \) classes, \( K(K-1)/2 \) binary SVMs are constructed and a final class is assigned by majority vote. Penalty
parameter $C$ and kernel parameter $\gamma$ are tuned on held-out development data using the LIBSVM grid-search tool.

As with other supervised models, we use character n-grams as features. We define a numeric feature for each character n-gram in the training set, representing the count of occurrences of that n-gram within the input word. We count n-grams with length 1 (unigram) up to some value $n$; here we use $n = 3, 4, \text{ and } 5$. Initial experiments show that scaling of features does not improve accuracy, and can lower accuracy in some cases, so we leave data unscaled. For comparison we also apply an SVM model that uses strictly binary features indicating presence or absence of an n-gram within the word, as done with Maximum Entropy models.

2.4.3 Experiments

Figures 2.3, 2.4, and 2.5 shows accuracy results for supervised methods described in this section. Figure 2.3 shows results for Witten-Bell and additive smoothing. Results for Witten-Bell smoothing show little variation with n-gram order, although the 5-gram models shows a small but statistically significant 0.2% improvement over the trigram model as we approach the largest training set size. As expected Witten-Bell outperforms add-one-in smoothing, which is shown here as a comparative baseline. However, additive smoothing with an additive weight of 0.1 shows results that are comparable to Witten-Bell.

Figure 2.4 compares MaxEnt and sequential MaxEnt models. We see that our sequential MaxEnt shows better results for all sample sizes, although it does not outperform our best language model for this training set.

Figure 2.5 shows results for SVMs, after parameter tuning. Using either binary or numeric features, SVMs did not outperform our best MaxEnt or language model classifiers. We found that including higher order n-grams in the feature set actually lowered accuracy for smaller training set sizes. Also, we found that SVMs tended to be brittle with respect to parameter tuning. With poor tuning, they naively assigned the major class label to all data points.

Table 2.4 shows a confusion matrix for CAS-LM with a sample size of 500. As expected we see high confusion between English and German which are related European languages. Additionally we see confusion between unrelated language pairs, for example Japanese and Arabic, showing that linguistic distance does not necessarily simplify the language classification task.
Figure 2.3: Accuracy results for supervised classification using Witten-Bell and Additive Smoothing. Results are on six-fold cross-validation with varying training set size.

Table 2.4: Confusion matrix for CAS-LM with a sample size of 500 after 6-fold cross-validation. Cell \((i, j)\) shows percentage of \(i\) classified as \(j\). Bolded diagonal shows accuracy for each class. Accuracy for entire data set is 86.2%.
Figure 2.4: Accuracy results for supervised conditional models. Results are on six-fold cross-validation with varying training set size. For comparison the best language model classifier is also displayed: Witten-Bell smoothing over 5-grams.
Figure 2.5: Accuracy results for SVM classifiers. “Numeric SVMs” use counts of character n-grams as features. We count all n-grams of order 1 (unigram) through order $n$, with $n = 3, 4, 5$. “Binary SVMs” use binary features (presence vs. absence). For comparison, we also include our best Witten-Bell and Sequential MaxEnt models from previous experiments.
Table 2.5: Hand annotation on test set of 6000 names, after training on 500 names. Column ‘% of data’ gives percentage of 6000 names that belong to this class. Cell (i, j) shows number of points in language i classified as language j and, in parenthesis, percentage of i classified as j. Bolded cells on the diagonal show accuracy for class i. Overall accuracy is 82.2%.

### 2.4.4 Hand annotation

As a comparison with supervised methods we hand-annotate one fold of 6000 place names by hand. A human annotator first “trains” on a practice set of 500 place names, meaning that they label each instance and are then shown the gold-standard label when incorrect. The test set of 6000 names are then labeled by hand; the labeler is not shown corrections or additional language information while labeling this set. Results are shown in Table 2.5. Overall accuracy is 82.2% which is lower than all supervised methods presented here after training on 500 labels.

The strong performance of supervised classification compared with human annotation motives our choice to focus on active learning and semi-supervised methods in this project. We see that supervised methods perform well when training data is available. Our goal is to guide sampling and labeling to maximize their performance on new data sets and domains.
2.5 Cluster-adaptive sampling

Next we apply active learning to the language identification problem by implementing Cluster-Adaptive Sampling (CAS) as presented in [Dasgupta and Hsu 2008] and described in Section 2.3.1. We apply CAS to our language identification in two ways, as presented in the original work:

1. CAS is used as a stand-alone tool for labeling an entire data set. After the sampling budget is spent, class labels are propagated from labeled to unlabeled points based on cluster structure of the input hierarchical tree. We refer to this labeling approach as Stand-alone CAS. Supervised methods presented in Section 2 are not applied to Stand-alone CAS.

2. CAS is used as a tool to guide sampling, as a substitute for random sampling, to build a training set for supervised classification. After CAS terminates, a supervised classifier is trained CAS labels; here we apply a language model with Witten-Bell smoothing and a sequential MaxEnt model. In the remainder of this work we refer to these approaches as CAS-LM and CAS-MaxEnt.

In this section we first present some modifications for each of these approaches and then present our results over a range of sampling sizes. Experiments in this section use the National Geospatial-Intelligence Agency (NGA) data set described in Table 2.2 and also used in supervised learning experiments.

2.5.1 Cluster-Adaptive Sampling Implementation

We implement Cluster-Adaptive Sampling as described in [Dasgupta and Hsu 2008] and in Section 2.3.1. CAS requires a hierarchical clustering tree $T$ of the data set. We construct $T$ using the Cluster open-source toolkit. We use character n-grams as features, specifically we find the $M$ most frequent n-grams of orders 2 through 5 in the data set and for each n-gram $i$ create an integer feature $f_i$, corresponding to the count of $i$ in an input word. Here $M$ is some threshold that can be varied to adjust the feature set size. As in [Dasgupta and Hsu 2008] we use average-linkage for hierarchical clustering.

Labeling by inner subtree: Stand-alone CAS

For Stand-alone CAS, we apply a simple extension of the labeling heuristic presented in [Dasgupta and Hsu 2008]. In the original work, the final output of CAS includes a pruning $P$ of the hierarchical clustering tree, where $P$ is a subset of nodes $\{v_q, \ldots, v_m\}$ such that nodes $v_i$ are disjoint and cover all the data. For labeling, the majority label of each node $v_i$ is assigned to all its unlabeled points. In practice we find that nodes in $P$ often contain small “pockets” of pure subtrees that were not included in the final pruning. In these experiments we allow the label of a node that is a descendant of a pruning node to determine the labeling of its leaf points.

Re-estimation of class prior: CAS-LM

One disadvantage of CAS is that, by design, labeled nodes are not sampled at random, and “pure” clusters are avoided during sampling once they are identified as such. As a result priors of frequent classes, which tend to form large pure clusters, may be underestimated, and this can introduce error into generative language models. Here we present a method of re-estimating class priors using CAS output. Our intuition is that if a subtree has been sufficiently sampled by CAS to predict a label for that subtree, then its unlabeled points should contribute to weighting of class priors for the whole data set.

In our method, priors are computed recursively at each node based on estimate from right and left child subtrees. We rely on the concept of an admissible subtree-label pair, as described in [Dasgupta and Hsu 2008]. In general terms, a label $l$ is admissible for subtree $v$ when it can be inferred that $l$ incurs twice as much error as any other label when $l$ is assigned to all points in that subtree. We re-weight class priors as follows: for any subtree $v$ with right and left child subtrees $a$ and $b$, if either $a$ or $b$ have an admissible label then $P_v(l_k)$, the probability of language $k$ in $p$, is re-estimated as:

$$P_v(l_k) = \frac{n_a}{n_v} P_a(l_k) + \frac{n_b}{n_v} P_b(l_k)$$  \hspace{1cm} (2.15)

where $n_i$ is the number of nodes in subtree $i$. If neither $a$ nor $b$ have an admissible label then we make no assumptions about their relative class distributions, and priors are then computed directly from counts at $v$. On average our method of re-weighting class priors improves accuracy 0.24%, which tests as significant at the 5% level using a paired $t$-test.
2.5.2 Experiments

We next present our experiments and results with the Cluster-Adaptive Sampling framework on our place name data set, as shown in Table 2.2. CAS allows sampling of an arbitrary number of points until the sampling budget runs out; we use sample sizes between 25 and 2000. Hand labeling is simulated by returning the automatically acquired country label when a label is queried. We do evaluation over the entire set of 10,000 place names. Points that were labeled during sampling are included in accuracy counts.

We use random sampling as a baseline, where we sample points at random to build a labeled training set for supervised classification. We apply two of our supervised methods from Section 2: a 5-gram language model with Witten-Bell smoothing, and our sequential Maximum Entropy model. Since we are comparing results with active sampling, we again credit labeled points toward overall accuracy.

Figure 2.6 shows results for cluster-adaptive sampling. As a comparative baseline we also show random sampling with the same supervised classifiers. For very small sample sizes, stand-alone CAS is comparable to sampling followed by supervised classification. However as sample size grows past 50, stand-alone CAS is quickly surpassed by our random-sampling baselines. With supervised classification, CAS shows a consistent improvement over random-sampling, and CAS-LM gives the best final accuracy results. After labeling 1000 names, or 10% of the data, CAS-LM shows an accuracy of 91.6%. Random sampling requires over 1600 labels to reach the same level of accuracy.

To further analyze the supervised classifiers generated via CAS-LM and CAS-MaxEnt, we also evaluate on a held-out set of 1000 place names. These names are not included in the hierarchical clustering tree $T$ used as input to CAS. The distribution of language classes follows those in the original set (Table 2.2). For comparison we compare results with random sampling, where training sets are selected from the original 10k set and evaluation is done on the same held-out set of 1000 place names. Results are shown in Figure 2.7. As with Figure 2.6, we see that CAS improves accuracy for of both the LM and MaxEnt classifiers, compared with random sampling. Then, the CAS framework may also be appropriate for scenarios when not all data points are available upfront.
Figure 2.6: Accuracy results for cluster-adaptive sampling (CAS), for sample sizes 25-400 (top) and 25-2000 (bottom). Random sampling results are also shown. All data points are averaged over 100 runs.
Figure 2.7: Accuracy results for cluster-adaptive sampling (CAS), on held-out development set of 1000 place names. Results are averaged over 100 runs per training set size.
Next we present how we incorporate spectral clustering into the cluster-adaptive sampling framework. We follow steps presented in 2.3.2 that apply spectral clustering to build a hierarchical clustering tree $T$ to be used as input to cluster-adaptive sampling.

The similarity matrix $A$ is computed from the same feature matrix used in previous CAS experiments which uses character n-grams of order 2 through 5. The similarity matrix $A$ is computed as a $k$-nearest neighbor (knn) graph, where vertex $v_i$ is connected with $v_j$ if $v_i$ is among the $k$ nearest neighbor of $v_j$ or $v_j$ is among the $k$ nearest neighbors of $v_i$. In our experiments we test $k = 10$ and $k = 100$.

Results are shown in Figure 2.8. First consider Stand-alone CAS without supervised classification: we see that incorporating spectral clustering as described above improves accuracy of Stand-alone CAS at all samples sizes. Additionally, at very small sample sizes, up to 100 data
points or 1% of the data, spectral clustering with stand-alone CAS shows better results than methods that use supervised classification.

### 2.7 Word origin detection of US Surnames

Next we apply our methods to a second data set of person surnames using data made available by the US Census Bureau.\(^4\). Files include the 88,799 most frequent US surnames from the 1990 Census, with frequency counts. Of the names that occur at least 500 times, we randomly select 3,000 for our surname data set.

To facilitate evaluation we label the entire 3,000 name data set in advance. Our goal in labeling is to identify, for every surname, its *most likely language of origin* when it occurs in the United States. Labeling was done primarily using the *Dictionary of American Family Names* [Hanks 2003] and Ellis Island immigration records.\(^5\) Additional resources were used when these sources were inconclusive. Generally this happened because a name was widely introduced to the US by post-Ellis Island immigration, as was the case with many Spanish surnames.

We find that in many cases a single surname cannot be attributed to single language but can be assigned to a set of languages related by geography and language family. For example multiple surnames were discovered that could be ambiguously labeled as English, Scottish, or Irish in origin (e.g. *Tirey* and *Hanlin*). For languages that frequently co-occur, we created a single *language group* to be used as a class label in our experiments. Groups were created for British Isles, Slavic, Scandinavian, and Indic languages. (The Indic group was created because limited information was available to further classify Indian surnames.) Table 1.6 shows the 33 language classes discovered in the US Census data set.

205 of the 3,000 surnames (6.8%) were not assigned any language class, either because language of origin information was unavailable or inconclusive. These unlabeled names are not used for evaluation, leaving a test set of 2795 surnames. When queried during sampling, unlabeled names are still deducted from the sampling budget. In a live system these names may be assigned a label via machine classification, so a complete classification of the data set is still possible.

---


5. Available online at [http://www.ellisisland.org](http://www.ellisisland.org)
<table>
<thead>
<tr>
<th>Language/Language Group</th>
<th>Count</th>
<th>Languages (for Language Groups)</th>
</tr>
</thead>
<tbody>
<tr>
<td>British Isles</td>
<td>1128</td>
<td>English, Scottish, Irish/Gaelic, Welsh</td>
</tr>
<tr>
<td>German</td>
<td>608</td>
<td>German, Swiss German</td>
</tr>
<tr>
<td>Italian</td>
<td>260</td>
<td>-</td>
</tr>
<tr>
<td>Spanish</td>
<td>194</td>
<td>Spanish, Catalan</td>
</tr>
<tr>
<td>Slavic</td>
<td>148</td>
<td>Polish, Czech, Slovak, Croatian, Serbian, Slovenian, Bulgarian, Russian, Ukrainian, Belarusian</td>
</tr>
<tr>
<td>French</td>
<td>145</td>
<td>-</td>
</tr>
<tr>
<td>Dutch</td>
<td>83</td>
<td>-</td>
</tr>
<tr>
<td>Scandinavian</td>
<td>68</td>
<td>Swedish, Danish, Norwegian</td>
</tr>
<tr>
<td>Arabic</td>
<td>24</td>
<td>-</td>
</tr>
<tr>
<td>Indic</td>
<td>22</td>
<td>Hindi, Tamil, Bengali, Gujarati, Punjabi</td>
</tr>
<tr>
<td>Japanese</td>
<td>20</td>
<td>-</td>
</tr>
<tr>
<td>Hungarian</td>
<td>15</td>
<td>-</td>
</tr>
<tr>
<td>Portuguese</td>
<td>15</td>
<td>-</td>
</tr>
<tr>
<td>Chinese</td>
<td>13</td>
<td>-</td>
</tr>
<tr>
<td>Greek</td>
<td>8</td>
<td>-</td>
</tr>
<tr>
<td>Yiddish</td>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>Frisian</td>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>Finnish</td>
<td>5</td>
<td>-</td>
</tr>
<tr>
<td>Armenian</td>
<td>4</td>
<td>-</td>
</tr>
<tr>
<td>Korean</td>
<td>4</td>
<td>-</td>
</tr>
<tr>
<td>Filipino/Tagalog</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>Lithuanian</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>Romanian</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>Navajo</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>Vietnamese</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>Cambodian</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>Hawaiian</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>Powhatan</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>Nahautl</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>Farsi</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>Igbo/Nigeria</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>Ghana (language unknown)</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>Wolof/Senegal</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>Unlabeled</td>
<td>205</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2.6: Language classes discovered via hand-labeling of 3000 U.S surnames. In some cases languages are grouped together to find a single language class; grouped languages are then shown in the third column.
Figure 2.9: Accuracy results on US Census data set, for random and cluster adaptive sampling (CAS). Results are averaged over 100 runs per labeling budget size. For series “CAS-MaxEnt”, a circle indicates a significant improvement over “Random Sampling-MaxEnt” based on unpaired t-test at the 5% significance level.

Table 2.7: Confusion matrix for US Census data set. Cell \((i, j)\) shows percentage of points in language \(i\) classified as language \(j\). Bolded cells on the diagonal show accuracy for class \(i\). Overall accuracy is 59.8%. Combined accuracy of 6 most frequent classes listed above is 66.2%.
Figure 2.10: Accuracy per language class, for 3,000 surname US Census data set. The six most frequent language classes are shown: British Isles (37.6%), German (20.7%), Italian (8.6%), Spanish (6.4%), Slavic (4.9%), French (4.8%). Other (10.4%) includes points that are not in one of these six classes but were assigned a class label. All data points are averaged over 100 runs.
Figure 2.11: Percentage of labeled points from each of the six most frequent language classes. “Other and Unlabeled” shows percentage from all other classes excluding these six, plus points that were left unlabeled. Results are averaged over 1000 runs per labeling budget size.
2.7.1 Experiments

We repeat our experiments with CAS, random sampling, and spectral clustering from the previous section. Figure 2.9 shows results for the US surname set. Comparing supervised classifiers, we see that a Maximum Entropy model (CAS-MaxEnt) shows better results than the generative language model (CAS-LM), which is the reverse of what we observed in the NGA Place Name data set in Figure 2.6. Also unlike place name experiments, spectral clustering noticeably reduces the performance of Stand-alone CAS for the Census data set.

Comparing CAS against random sampling, we see that CAS leads to a small improvement for some sample sizes when used with supervised classifiers to label unsampled points. Significance tests are inconclusive, as we find a significant improvement in some, but not all, sample set sizes. The average increase is 0.3% and 0.2% for CAS-MaxEnt and CAS-LM respectively. In contrast, we see a larger and more consistent improvement with the NGA Place Names data set (Figures 2.6), with an average increase of 1.9% for CAS-MaxEnt and 1.5% and CAS-LM. These differences test as significant for all sample sizes above 25 data points.

Figure 2.10 shows accuracy by language class for the six most frequent classes in this surname set: British Isles, German, Italian, Spanish, Slavic, and French. All other labeled points are grouped as “Other”. We see that the two most frequent classes, British Isles and German, show no improvement with CAS, and in fact German suffers for smaller sample sizes. In contrast less frequent classes, such as Slavic, Italian, and Spanish, do show gains in accuracy at varying intervals.

2.7.2 Error analysis

We propose two factors to explain why CAS is not as effective with the Census data set as with the Place Name set. First, the Census data set contains more classes than the Place Name set (33 vs. 8) and many of these classes have very low counts, with 19 having less than 10 data points. Since the goal of CAS is to direct sampling towards less pure subtrees, nodes that contain these low-count language classes will tend to be preferred in sampling. We can see this in Figure 2.11 which shows the percentage of sampled points drawn from each of the six most frequent language classes, as well as points outside the top six grouped in a single class ("Other and Unlabeled"). Here we see that sampling moves away from the majority British class as the budget increases. The next most frequent class, German, composes 20.7% of overall data but is consistently sampled
at a lower rate. In contrast, “Other and Unlabeled” gradually gains weight over time. Points in this set, however, will make minimal contributions to CAS-MaxEnt accuracy, as unlabeled points are unusable for training, and very infrequent classes will not have enough training samples to be useful for supervised classification. Effectively the CAS labeling budget is directed away from the majority classes, British and German, but not necessarily towards data that improves supervised classifiers.

A second reason for lower performance with Census data is higher ambiguity among frequent classes in this set. Figure 2.7 shows the confusion matrix for the six most frequent classes for supervised classification with the MaxEnt model and a training size of 500 points. Accuracy with this set of six is 66.2%, much lower than the overall accuracy of 86.2% for similar experiments with the eight class Place Name set (Table 2.4). We see that names in these top-six classes still have high ambiguity with each other, with all classes showing frequent confusion with the most frequent “British” class. We note that one source of ambiguity is a tendency towards “Americanization” of surnames, where names are partially altered toward an English/American spelling but still have an identifiable non-English origin that is assigned during labeling. This is especially common in German-origin surnames. For example, our data set includes the names Rohrback and Eisenhour, both labeled as German, which are Americanized forms of the original German surnames Rohrbach and Eisenhauer.

We hypothesize that these two factors, the presence of more low-count classes and higher confusion among frequent classes, have a negative affect on CAS because they reduce cluster purity in hierarchical clustering tree $T$ which is then used as input to CAS. A core assumption of CAS is that it can discover pure subtrees at some level of $T$ and will direct sampling towards less pure regions of the tree. But if $T$ fails to form class pure subtree than the behavior of CAS will be closer to random sampling.

As a means of measuring cluster purity in $T$ we compute the conditional entropy of the language label given the cluster, for some node pruning $P$ that is disjoint and covers all the points in $T$:

$$H(L|P) = - \sum_{t \in P} p(t) \sum_{l \in L} p(l|t) \log p(l|t)$$  

Here $L = l_1 \ldots l_k$ is the set of language class labels and $t_1 \ldots t_m \in P$ are nodes or clusters in the pruning $P$. $p(t)$ is the proportion of nodes in $T$ covered by subtree $t$. We seek to measure $H(L|P)$ independent of the CAS output, and so we cut $T$ at varying levels based on node correlation

37
score between right and left subtress in order to obtain prunings of varying sizes. Correlation is measured as the average pairwise cosine similarity between elements in each subtree, as used in initial hierarchical clusterings. Unlabeled points in Census data are each placed in their own class for the purposes of computing $H(L|P)$.

Figure 2.12 shows $H(L|P)$ for spectral and non-spectral clustering trees for both the Census and Place Name data sets. Figure 2.13 shows the same results for larger pruning sizes. Here we see minimal difference between spectral and non-spectral clusterings for small pruning sizes with Census data. In contrast, with Place Name data spectral clustering does reduce conditional entropy immediately after the root is split, which is how we explain the benefit of spectral clustering for small budget sizes as observed in Figure 2.8.

To analyze the effect of low-count classes in the Census data set, we perform an additional hierarchical clustering that only includes data points in the six most frequent classes listed in Table 2.7 (British, German, Italian, Spanish, Slavic, and French); low-count classes and unlabeled points are removed. This set is shown with the label “US Census-Top six classes only” in Figure 2.12. As expected we see that initial entropy $H(L)$, or conditional entropy when the number of clusters is one, is lower than clustering trees that include all Census data. Coincidentally this Top-six set has approximately the same starting entropy as Place Name data. As pruning size increases from one, conditional entropy for Top-six decreases at a slower rate than Place Name clustering trees, despite the fact that Top-six is the smaller data set that will reach zero uncertainty first. From this we see that the presence of low-count classes is not the sole cause of increased subtree impurity in Census data, and confusion among frequent classes is also a factor.

Qualitatively we recognize that language ambiguity is an inherent problem of clustering and labeling of US surnames. Surnames in the Census data set are discovered to cover a large number of languages, some of which are related to each other by geography, overlapping populations, and linguistic origin (i.e. Spanish and Italian, Dutch and German). In addition US surnames have been altered by Americanization in a way that we do not see with the NGA Place Name data set. The varying results between these two data sets lead us to conclude that cluster-adaptive sampling is most applicable when the boundaries between language classes are readily identified and defined.
2.8 Conclusion

In this chapter we apply Cluster Adaptive Sampling (CAS) to the problem of word origin recognition. We choose this method because it does not require word origin training data or prior knowledge of a set of origin languages; instead both are acquired by sampling of unlabeled data. Two previous works in transliteration, [Bose and Sarkar 2009] and [Hagiwara and Sekine 2011], both apply word origin clustering in an unsupervised manner, i.e. without a word origin training set; however we know of no previous works that applied active learning to this problem. [Chen and Maison 2003] discovers language classes using geographic place name data organized by country-specific files, but this has the limitation that languages that are not well-represented by the place name list of one or more countries will not be discovered as a language class.

We apply CAS to two data sets, one of geographic place names as in [Chen and Maison 2003], and other created by randomly selecting person surnames from US Census data. We discover that CAS benefits subsequent supervised learning with the first set. However with Census data, Americanization and confusion between language classes prevents CAS from returning an improved training set for supervised methods.

We also perform experiments with supervised word origin classification and present a sequential Maximum Entropy model that improves on a non-sequential Maximum Entropy model trained on the same features.
Figure 2.12: Conditional entropy of language label given cluster, for Place name and US Census hierarchical clustering trees. Triangle, circle, and square on y-axis mark entropy of unclustered Census, Top-six Census, and Place Name data sets respectively.
Figure 2.13: Conditional entropy of language label given cluster, for Place name and US Census hierarchical clustering trees, for pruning size of up to 4,000 clusters.
CHAPTER 3
GRAPHEME-TO-PHONEME CONVERSION USING WORD ORIGIN INFORMATION

In Chapter 2 we discuss the motivation for word origin classification and present works that apply word origin recognition to character transliteration and pronunciation modeling. In this chapter we present methods to use word origin information to improve grapheme-to-phoneme (G2P) conversion, specifically for person surnames. Our goal is to show how word origin information can be used to choose training sets for language-specific G2P models, and how output from multiple language-specific models can be combined to improve prediction of the single best pronunciation of a name. We show how we account for low training counts with rare language classes by reweighting output of language-specific and language-independent models. Our methods are a wrapper to the Sequitur G2P toolkit and can be applied to other G2P systems that output posterior probabilities of the phonemic transcriptions give the input character sequence.

As in the previous chapter we use a data set of frequently occurring surnames in the United States. In the last chapter we found that US surnames often are “Americanized” in their written form; the same can be expected of their pronunciation. With this in mind, our goal is not to predict pronunciation of names in the origin language, but rather to model how a name is most likely to be pronounced in standard US English. We use an existing pronunciation dictionary as a gold standard for this task.

3.1 Related Works

In previous works, word origin information has been applied to speech recognition and G2P conversion of proper names. Works presented here that study recognition address two related problems: pronunciation of non-native speakers, and pronunciation of foreign words by native speakers. Only the latter is within the scope of this project. For G2P and monolingual speech recognition, “native” refers to the language of the acoustic model and the phoneme alphabet used.
3.1.1 Automatic speech recognition

[Maison et al. 2003]

[Maison et al. 2003] models pronunciation of foreign proper names to improve English automatic speech recognition (ASR) accuracy. In particular phonemic transcriptions of foreign names are automatically generated and added to the vocabulary of an ASR system for US English. Transcriptions are generated as follows: first, separate pronunciation models are built for eight languages (Spanish, French, German, Italian, Czech, Russian, Hindi and Mandarin Chinese). These language-specific models are built from pronunciation dictionaries of sizes from 12k to 306k transcriptions. Hand-crafted rules are used to map the phonemic alphabet of each dictionary to that of the US English ASR system. Characters are also mapped between grapheme alphabets. Pronunciation models are constructed for each foreign language using maximum entropy n-gram model as presented in [Chen 2003]. Units are “chunks” which are character-phoneme pairs where either the character or phoneme may be null. The joint probability of a letter sequence \( L \) and phoneme sequence \( P \) is computed by summing over all chunk sequences \( C = c_0, \ldots, c_m \) that generate both \( L \) and \( P \):

\[
P(L, P) = \sum_{C \rightarrow (L,P)} P(C = c_0, \ldots, c_m) = \prod_{i=1}^{m} P(c_i | c_1 \ldots c_{i-1}) \tag{3.1}
\]

Foreign pronunciation models are then used to augment US English pronunciation dictionaries for a vocabulary of 24k names. For each name, language identification is applied, using methods of [Chen and Maison 2003], to generate a list of up to \( k \) languages with prior greater than 0.05; for each language up to \( l \) phonemic transcriptions are generated with the appropriate language-specific model. The authors note the best performance with \( k, l \leq 2 \). Three test sets are presented for ASR evaluation:

1. Foreign names spoken by native or fluent speakers of the origin language. This set is constructed by crossing lists of international chess and Go players with a US English pronunciation dictionary. Names where the corresponding foreign-language model assigns low probability (\( \leq 0.2 \)) are discarded from that test set, so that evaluation is not done on names where origin language is highly uncertain. Speakers are instructed to say names in the original, non-anglicized form.

2. Foreign names spoken by US speakers. A subset of names from the first test set is used.
3. US names spoken by US speakers. The authors note that this set contains names of non-
English origin. The source of these US names is not given.

Two baseline lexicons are used. The first is a lexicon where pronunciations are first generated
by a rules-based system and then edited by an expert to allow for anglicization by US speakers.
The second is a lexicon where transcriptions of test set names are automatically generated from a
model trained on an English lexicon. The handcrafted lexicon performs best on all three test sets
including the set of US names only.

Experimental results show that the inclusion of transcriptions from foreign models reduces
error in all three test sets. For the first test set of foreign names and foreign speakers, relative
error is reduced by about 25% for both baseline systems. The authors also note that no single
language class showed an increase in recognition error. For the second set of foreign names and
US speakers, relative error is reduced less, by about 10%. For both foreign name sets, the best
results are shown when transcriptions from the two most likely candidate languages are added to
the lexicon \( k = 2 \). Improvement on the third set of US names is minimal, from 15.3% to 15.0%.
For this set the baseline that was handcrafted for foreign names has lower error than the baseline
with automatically generated English transcriptions. In addition, the handcrafted system without
foreign transcriptions still outperforms automatic G2P output with foreign transcriptions.

[Cremelie and ten Bosch 2001]

[Cremelie and ten Bosch 2001] applies lexical adaption of ASR systems to account for two sources
of error, non-native speech and foreign words spoken by native or non-native speakers. Data is
taken from an automated attendant system and a database of dialing instructions. A Dutch acoustic
model is used, and the native languages of the speakers are Dutch, French, and English.

Experiments show that recognition is improved by adding English and French phonemic tran-
scriptions to the baseline lexicons; as in [Maison et al. 2003] phonemes from other language
systems are mapped to a Dutch phonemic alphabet. The authors note that penalty weighting of
phonemic transcriptions has previously been shown to improve ASR accuracy when multiple tran-
scriptions for a single word are included in the lexicon. This work penalizes individual phonemic
transcriptions using language information in two different ways. First, a language-global penalty is
assigned for all transcriptions from the same language/G2P model. Second, individual name tran-
scriptions are weighted based on the probability of the names language of origin. For word-specific
weighting, the penalty of word $w$ for language $l$ is:

$$w_l = M[1 - \frac{P(l|w)}{\arg \max_{l'} P(l'|w)}] \tag{3.2}$$

Here $M$ is a scaling factor that can be optimized on a tuning set. The method used for language identification is not specified. Recognition for both data sets improves with foreign phonemic transcriptions and further improve with language weighting. The relative benefit of each weighting scheme varies with the data set, which the authors attribute to differences in the acoustic models.

[van den Heuvel et al. 2009], [Reveil et al. 2009], [Reveil et al. 2010]

[Reveil et al. 2009] studies the problem of ASR for proper names and compares affects of the following factors: monolingual vs. multilingual acoustic models, native language of the speaker, the inclusion of foreign phonemic transcriptions. For monolingual ASR, foreign phonemic alphabets are again mapped to the native alphabet. Phonemic transcriptions are generated from existing foreign and native G2P systems; these models are not adjusted for this work. Related works, [van den Heuvel et al. 2009] and [Reveil et al. 2010] apply phoneme-to-phoneme (P2P) transcription to improve recognition. Here phonemic representations are converted into variants that more closely match training data, specifically conversion is done between native (Dutch) and non-native pronunciation.

**Related works in machine transliteration**

A natural correspondence exists between G2P conversion and machine transliteration, as both tasks aim to correctly map words from a source to a target alphabet. In Section 2.1 we presented works that applied word origin recognition to transliteration, including [Surana and Singh 2008], [Khapra and Bhattacharyya 2009], [Bose and Sarkar 2009], and [Hagiwara and Sekine 2011]. [Hagiwara and Sekine 2011] is particularly relevant to experiments in this chapter, as this work also weights output of multiple class-specific transliteration models using estimates of $P(c|s)$, the probability of the class given the source language. Also, in a separate set of experiments, this work discovers latent class variables without word origin supervision. We distinguish experiments in this chapter from [Hagiwara and Sekine 2011] in two ways. First, we use a data set where multiple languages of origin occur naturally, while [Hagiwara and Sekine 2011] creates language-specific lists and
then merges them into a single set. Second we consider methods of smoothing against a language independent model to overcome the problems of data sparsity and word origin recognition error.

### 3.1.2 G2P conversion for TTS

As presented in Chapter 2, [Llitjós and Black 2001] and [Llitjós 2001] use language identification to improve text-to-speech (TTS) pronunciation. The goal is modeling of “pronunciation of foreign proper names in American English”, and unlike [Maison et al. 2003] this work does not specifically account for non-native speakers of English. Data is taken from a Bell Labs list of 50,000 US surnames and 6,000 first names with pronunciations. According to [Llitjós 2001] these lists were developed at AT&T Bell Labs during the 1980s using directory listings. Pronunciations for these names are included in the CMU Pronouncing Dictionary\(^1\). 10% of the this set of 56k names is held out for testing and 90% used for training CART models.

A CART decision tree model is used as the non-language-specific baseline. This model maps letters to phonemes using local context, where a single letter may map to one phoneme, two phonemes, or \( \epsilon \) (null). Training data is acquired from character-phoneme alignments of the training data. Language identity is predicted using character trigram counts for a set of 26 candidate languages. The following language-identity features are included in the CART decision tree model:

1. \( \text{Language}_1: \arg \max_L P(L|\text{word}) \)
2. \( \text{Prob}_1: P(\text{Language}_1|\text{word}) \)
3. \( \text{Language}_2: \arg \max_{L \neq \text{Language}_1} P(L|\text{word}) \)
4. \( \text{Prob}_2: P(\text{Language}_2|\text{word}) \)
5. \( \text{delta-probability}: \text{Prob}_1 - \text{Prob}_2 \)

Evaluation shows that the CART model with language-origin features improves word accuracy by 7.6%, from 54.1% to 61.7%. Results on individual language classes are not reported.

\(^1\) http://www.speech.cs.cmu.edu/cgi-bin/cmudict
3.2 Sequitur: G2P conversion with joint sequence models

Experiments in this report are based on a joint sequence model for G2P conversion, as presented in [Bisani and Ney 2008]. Here joint refers to the fact that the basic model unit is a joint pair of graphemes and phonemes, or a graphone. Graphones are analogous to chunks in [Maison et al. 2003] with the exception that more than one symbol is allowed on the grapheme or phoneme side. The maximum allowable number of symbols per side is model parameter \( L \).

This work presents a data-driven probabilistic model that maximizes the joint probability \( P(\bar{g}, \bar{\phi}) \) of phoneme sequence \( \bar{\phi} = \phi_1 \ldots \phi_N \) and input grapheme sequence \( \bar{g} = g_1 \ldots g_N \). This is done by summing over \( S(\bar{g}, \bar{\phi}) \) the set of all possible graphone segmentations of \( \bar{g} \) and \( \bar{\phi} \) given the set of graphones \( Q \) in the model:

\[
P(\bar{g}, \bar{\phi}) = \sum_{\bar{q} \in S(\bar{g}, \bar{\phi})} P(\bar{q})
\]  

(3.3)

\( \bar{q} = q_1 \ldots q_K \) is estimated with an M-gram model, where \( M \) is also a model parameter.

\[
P(\bar{q}) \approx \prod_{j=1}^{K+1} \left( q_j | q_{j-1}, \ldots, g_{j-M+1} \right)
\]  

(3.4)

Expectation Maximization is used to estimate an initial set of graphones, and Kneser-Ney smoothing is used to estimate the M-gram model. Posterior probabilities are estimated as:

\[
P(\bar{g}) = \sum_{\bar{\phi}} P(\bar{g}, \bar{\phi}) = \sum_{\bar{q} : g(\bar{q}) = \bar{g}} P(\bar{q})
\]  

(3.5)

\[
P(\bar{\phi} | \bar{g}) = \frac{\sum_{\bar{q} \in S(\bar{g}, \bar{\phi})} P(\bar{q})}{P(\bar{g})}
\]  

(3.6)

An open-source implementation of Sequitur is available at:
http://www-i6.informatik.rwth-aachen.de/web/Software/g2p.html.
3.3 Data

As in [Llitjós and Black 2001], the focus of this project is pronunciation of US surnames in US English. For the 50,000 most frequent surnames in the 1990 census\(^2\) we extract those entries that also appear in the CMU Pronouncing Dictionary. According to [Jurafsky and Martin 2000] the CMU Dictionary contains the same Bell Labs pronouncing list used in [Llitjós and Black 2001]. This gives us a set of 45,841 surnames with their phoneme representation, transcribed in the Arpabet symbol set. We divide this set 80/10/10 into train, test, and a held-out development set. We assume that the CMU Pronouncing Dictionary contains the standard pronunciation in US English and use this resource as a gold standard in our experiments. We recognize that US surnames may have multiple valid pronunciations; however for this project we make the simplifying assumption that a name has one best pronunciation.

For word origin classification, we have the set of hand-labeled surnames used the previous chapter. This is a set of 2,795 surnames that are randomly selected from the same US Census surname list, and then labeled by hand for language of origin. 33 different language groups are discovered with the majority having less than 10 data points. In these experiments we use the 12 largest language classes: British, German, Italian, Slavic, Spanish, French, Dutch, Scandinavian, Japanese, Arabic, Portuguese, and Hungarian. Other names are grouped into an “Other class”.

3.4 Methods

Our goal in this project is to improve G2P conversion with word origin information. Our baselines is a single G2P model that is trained on pronunciations of US surnames, without any word origin information. To improve output we train additional language specific models and incorporate the output to re-estimate \(P(\bar{\phi}|\bar{g})\), the posterior probabilities output by a baseline model. At a high level our steps are:

1. Train a supervised word origin classifier with some set of languages \(L\). Use this classifier to compute \(P(l|w)\) for all \(l \in L\) and for all words in the training and test sets.

2. Train supervised G2P models for each \(l \in L\). Each language specific model \(m_l\) is trained on data that is predicted to be from language \(l\). Specifically, \(m_l\) is trained on words with \(P(l|w)\)

\(^2\) Reported at http://www.census.gov/genealogy/names/
greater than some threshold $\alpha$. Languages with very low training counts may be removed from $L$.

3. For each word $w$ in the test set, generate candidate phonemic transcriptions for each language with nonzero $P(l|w)$. Weight transcriptions to choose $\arg\max_{\bar{\phi}} P(\bar{\phi}|\bar{g})$, the most likely phonemic transcription of the word. Output of the baseline language-independent model may also be weighted and included.

The focus on this project is how to implement steps 2 and 3, that is how to use language posteriors to improve on a language-general G2P system. We also consider Step 1, which was addressed in the last chapter; in particular we consider how we can use additional data to improve initial word origin labeling.

3.4.1 Step 1: Word origin modeling

Our labeled set of 2,795 names from the previous chapter gives (1) an appropriate set of languages labels $L$ for US surname data, and (2) an initial training set for a word origin classifier. This classifier can then be applied to the unlabeled names in training and test sets to generate posterior probabilities $P(l|w)$ to be used in G2P re-weighting. Possible extensions are:

**Self-training for word origin classification**

We use the remaining 45k names extracted from the CMU Pronouncing Dictionary for self-training. Specifically, we iteratively label names for language of origin, add names labeled with high confidence to the training set, and then retrain. This is a generalized version of Yarowsky’s bootstrapping algorithm [Yarowsky 1995], [Abney 2004].

**Word origin classification using phonemic information**

We also have available the phonemic transcriptions of the 37k names in our training set that can be used to improve language groupings. One possible approach is co-training as presented in [Blum and Mitchell 1998], with a second word origin classifier trained on phonemes instead of characters. Co-training has the requirement that “each view in itself is sufficient for classification”. To test this, we train two word origin classifiers: the first on characters as done in the previous chapter,
and the second with phonemes in place of characters. For training data we use the subset of our hand-labeled Census data that also appears in the CMU dictionary. This yields 2,646 names with their corresponding gold standard phonemic transcriptions. We train two language model-based classifiers with Witten-Bell smoothing, one using character tokens and another using phonemes. In ten-fold cross-validation, the classifier trained on phonemes shows an accuracy of 57.60%, compared with 67.16% for the classifier trained on characters. A baseline of selecting the most frequent class yields an accuracy of 41.80%. From this we conclude that phoneme information alone is not a strong indicator of word-origin, at least not with this specific data set where pronunciation may have already been adapted for spoken US English.

An alternate approach is to use the graphones representation output by Sequitur, or any other character-to-phoneme alignment of the data. Effectively this gives us a single-view representation of the data that incorporates both characters and phonemes. As an additional experiment we train a third LM classifier using a graphone segmentation of the same data set of 2,646 names. So for example, surname *Brite* is represented as the graphone sequence \((b:B, r:R, i:AY, t:T, e: null)\), where \(b:BR\) is the graphone that pairs character \(b\) with phoneme \(B\). In ten-fold cross validation accuracy is 68.71% which tests as a significant improvement over the character-based classifier with a paired t-test at the 5% confidence interval. This method can be applied to G2P experiments by classifying all names in the training set by language of origin and using these labels to build language-specific G2P models. In this case we only use the subset of the 2,646 names that are also in the G2P training set, as phonemes for test are not available. So less annotated word origin data is available for training.

**Unsupervised word origin clustering**

In the previous chapter we showed how a hierarchical clustering of the input data can be applied to active sampling of a word origin training set. Here we use the same hierarchical clustering tree \(T\) to cluster similar names. We perform two clusterings: first by characters and then by graphones. In either case, we extract a fixed number of clusters from \(T\) by ordering tree nodes by increasing correlation score and then splitting nodes until we have \(k\) clusters, where \(k\) is chosen arbitrarily. That is, we split \(T\) starting at the root until we reach the desired number of clusters \(k\). Clusters with fewer than 100 points are merged into a single cluster. As in Chapter 2, we build \(T\) using cosine as a measure of similarity, and average pairwise distance is used as the distance between two clusters.
This approach to word origin clustering is unsupervised in that we use no annotated training set for word origin, and we make no assumptions about the languages represented in our set of US surnames. After a set of $k$ clusters is formed by cutting $T$, each $k$ cluster is treated as a pseudo-language and we build a $k$-class word origin classifier around this data. A language-specific G2P model is trained from each pseudo-language cluster. From there word origin priors are computed for all points in test and we proceed as described in the next section.

3.4.2 Steps 2 and 3: G2P Reweighting of Sequitur output

We present methods to re-estimate $P(\tilde{\phi}|\tilde{g})$ as initially output by a language-independent G2P model. The phonemic transcription of a word with grapheme representation $g$ is then chosen as $\arg \max_{\tilde{\phi}} \tilde{P}(\tilde{\phi}|\tilde{g})$ where $\tilde{P}$ is the re-estimated posterior probability.

**Method A**

For every word $w$ in the test set we compute $P(l|w)$ for all $l \in L$. Model $m_l$ is trained on all words with $P(l|w) > \alpha$. ($\alpha = 0.7$ for all experiments below.) We re-estimate $P(\tilde{\phi}|\tilde{g})$ as:

$$\tilde{P}(\tilde{\phi}|\tilde{g}) = \sum_{l \in L} P(\tilde{\phi}|\tilde{g}, l)P(l|g)$$

(3.7)

Output from the baseline language-independent model is not used.

**Method B**

Method A has the result that names from infrequent classes will suffer from data sparsity. We introduce a smoothing factor $\sigma$ so that results from the baseline language-independent G2P model are also given weight:

$$\tilde{P}(\tilde{\phi}|\tilde{g}) = \sigma P_{Indep}(\tilde{\phi}|\tilde{g}) + (1 - \sigma) \sum_{l \in L} P(\tilde{\phi}|\tilde{g}, l)P(l|g)$$

(3.8)

In experiments shown below we use the held-out development set to tune $\sigma$. However in practice we find that 0.5 works well as a default value.
**Method C**

Similar to Method B, but $\sigma$ becomes a language-specific parameter $\sigma_l$ that is tuned for each language $l \in L$ on a held-out development set.

$$
\tilde{P}(\bar{\phi}|\bar{g}) = \sigma_g P_{\text{Indep}}(\bar{\phi}|\bar{g}) + \sum_{l \in L} (1 - \sigma_l) P(\bar{\phi}|\bar{g}, l) P(l|g)
$$

$$
\sigma_g = \sum_{l \in L} P(l|g) \sigma_l
$$

We tune each $\sigma_l$ on held-out development set $D$ to maximize accuracy on the development set, weighted by language priors.

$$
\sigma_l = \arg \max_{\sigma'} \sum_{g \in D} P(l|g) I(g, \sigma')
$$

Here $I(g, \sigma')$ is an indicator function equal to 1 when $g$ is correctly transcribed with Method B using $\sigma'$, and 0 otherwise.

**Method D**

Rather than assign a global weight (Model B) or language-specific weight (Model C), we compute a weight for each word in the test set. We do this by treating the language-independent model as its own language class so that the the word-origin classifier also outputs $P(\text{Indep}|g)$. Our set of languages becomes $L' = L + \{\text{Indep}\}$.

$$
\tilde{P}(\bar{\phi}|\bar{g}) = \sum_{l \in L' = L + \{\text{Indep}\}} P(\bar{\phi}|g, l) P(l|g)
$$

This requires that we train a new word origin classifier that recognizes $\text{Indep}$ as a separate language class. For the Sequential Max-Ent classifier, we do this by re-adding all points to the word-origin training set with the $\text{Indep}$ label.
Table 3.1: G2P word accuracy on US Census test set, by method of word origin modeling (rows) and method of re-weighting G2P output from language-specific models (columns). 67.81% is the baseline accuracy for a single language-independent G2P model trained on the entire training set. Values in parenthesis show difference from baseline.

### 3.5 Experiments

Table 3.1 shows word accuracy results for methods presented in Section 3.4, evaluated on the test set of 4,585 names from the CMU dictionary. In all experiments in this section, we use the Sequential-MaxEnt model to compute word origin probabilities $P(l|g)$, as presented in Chapter 2.

G2P re-weighting methods A through D all use language-specific G2P models. The set of languages $L$ comes from hand-annotation of US Census data from the previous chapter. We use all language groups with at least 10 hand-labeled points in our G2P training set. In practice a higher cutoff may be preferable but we include low-count classes to examine their performance in these experiments. Remaining languages are grouped into a single “Other” class. For each language $l \in L$, G2P model $m_l$ is trained on names from the training set with $P(l|w) >= 0.7$. This means that ambiguous names ($\max_l P(l|w) < 0.7$) are not used for language-specific training. No filtering is done from the test set.

Next we present results from each of the last three rows in Table 1.1 corresponding to different word origin models. Tables 3.2, 3.4, and 3.5 show accuracy for a single word origin model, grouped by a name’s most likely language of origin. In these tables column “Train Count” is the size of the training set for language $l$. Accuracy of $l$ is measured over all points in the test set with $\arg \max_{l'} P(l'|w) = l$, regardless of posterior value. Note that Train and Test counts vary in Tables 3.2, 3.4, and 3.5, as they reflect the output of the word origin model used in the corresponding set of experiments.
3.5.1 Character-based word origin modeling

Table 3.2 shows word accuracy results using a character-based word origin model without self-training. “Baseline” shows results for a single language-independent G2P model. First we see that Method A, which uses only language-specific G2P models, has lower overall accuracy than the baseline. We attribute this to data sparsity introduced by dividing the training set by language label. With the exception of British and German, language-specific training set sizes are less than 10% the size of the language independent training set of 37k names. Also, since we do not have a ground-truth language label for all words in our training set, language-specific training sets are not guaranteed to be language-pure or to capture all names derived from a language class in the training set.

Examining results for individual language classes for Method A, we see that Italian and British are the only language classes where accuracy improves. For Italian, we attribute this to two factors: high variation in pronunciation from US English, and enough training data to build a successful language-specific model. In the case of British, a language-specific model removes foreign words but leaves enough training data to model the language sufficiently. German suffers with Method A despite having a larger training set than Italian. We assume that extracting British names is an overall loss for the German model, due to close pronunciation and ambiguity between these two language groups.

Methods B, C, and D, which apply some smoothing of \( P(\bar{\phi}|\bar{g}) \) with the language-independent model, all improve on the baseline. Method B shows the highest gain in accuracy, with 2.2% absolute improvement. However this is at the expense of the Dutch and Scandinavian classes, which show better results with the baseline model than when Method B is applied. We assume this is because names in these classes have close to standard pronunciation and are already well-modeled by a language independent model.

Methods B and C benefits the Slavic group less than Italian, even though we expect that pronunciation in both groups will vary from standard US English. We attribute this to a smaller training set for the Slavic class than for Italian. This in part motivates our experiments in this chapter to consider how word origin can be applied to G2P conversion when language-specific training data is limited for some classes.

As described in section 3.4.1, Methods C and D both use word-level posteriors \( P(l|g) \) to vary the weight with the language independent model. Neither surpass the performance of Method B,
<table>
<thead>
<tr>
<th>Language Class</th>
<th>Train Count</th>
<th>Test Count</th>
<th>Baseline</th>
<th>(A)</th>
<th>(B)</th>
<th>(C)</th>
<th>(D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>British/English</td>
<td>16,171</td>
<td>2111</td>
<td>71.81</td>
<td>73.14</td>
<td><strong>73.90</strong></td>
<td>73.76</td>
<td>73.33</td>
</tr>
<tr>
<td>German</td>
<td>8,360</td>
<td>1109</td>
<td>75.83</td>
<td>74.21</td>
<td><strong>78.18</strong></td>
<td>77.82</td>
<td>76.28</td>
</tr>
<tr>
<td>Italian</td>
<td>3,358</td>
<td>447</td>
<td>61.74</td>
<td><strong>66.22</strong></td>
<td>65.10</td>
<td>65.77</td>
<td>62.86</td>
</tr>
<tr>
<td>Slavic</td>
<td>1,658</td>
<td>232</td>
<td>50.86</td>
<td>49.57</td>
<td>51.72</td>
<td><strong>52.16</strong></td>
<td>50.86</td>
</tr>
<tr>
<td>Spanish</td>
<td>1,460</td>
<td>246</td>
<td>44.72</td>
<td>41.46</td>
<td><strong>47.97</strong></td>
<td>47.56</td>
<td>45.12</td>
</tr>
<tr>
<td>French</td>
<td>1,143</td>
<td>177</td>
<td>42.94</td>
<td>42.37</td>
<td><strong>45.20</strong></td>
<td>44.63</td>
<td>41.81</td>
</tr>
<tr>
<td>Dutch</td>
<td>468</td>
<td>82</td>
<td><strong>70.73</strong></td>
<td>52.44</td>
<td>68.29</td>
<td>67.07</td>
<td><strong>70.73</strong></td>
</tr>
<tr>
<td>Scandinavian</td>
<td>393</td>
<td>61</td>
<td><strong>77.05</strong></td>
<td>60.66</td>
<td>72.13</td>
<td>67.21</td>
<td>70.49</td>
</tr>
<tr>
<td>Japanese</td>
<td>116</td>
<td>23</td>
<td>73.91</td>
<td>52.17</td>
<td><strong>78.26</strong></td>
<td>65.22</td>
<td>73.91</td>
</tr>
<tr>
<td>Arabic</td>
<td>68</td>
<td>18</td>
<td>33.33</td>
<td><strong>38.89</strong></td>
<td>38.89</td>
<td>38.89</td>
<td><strong>38.89</strong></td>
</tr>
<tr>
<td>Portuguese</td>
<td>34</td>
<td>4</td>
<td>25.00</td>
<td>25.00</td>
<td><strong>50.00</strong></td>
<td><strong>50.00</strong></td>
<td>25.00</td>
</tr>
<tr>
<td>Hungarian</td>
<td>28</td>
<td>3</td>
<td><strong>100.00</strong></td>
<td>66.67</td>
<td><strong>100.00</strong></td>
<td><strong>100.00</strong></td>
<td><strong>100.00</strong></td>
</tr>
<tr>
<td>Other</td>
<td>431</td>
<td>72</td>
<td>55.56</td>
<td>54.17</td>
<td>59.72</td>
<td>58.33</td>
<td><strong>61.11</strong></td>
</tr>
<tr>
<td>All</td>
<td>-</td>
<td>-</td>
<td>67.81</td>
<td>67.42</td>
<td><strong>69.99</strong></td>
<td>69.70</td>
<td>68.72</td>
</tr>
</tbody>
</table>

Table 3.2: G2P word accuracy for various weighting methods using a character-based word origin model. For Method B, $\sigma = 0.5$ based on tuning on held out data set.

showing us that while it is intuitive to vary weight $\sigma$ by word or language class, in practice tuning a single global weight (Method B) provides comparable results.

We next look at some sample results from our G2P systems. Table 3.3 shows names where Method B generated the gold standard pronunciation and the baseline system did not. We show examples from Italian, Slavic, and Spanish. For the Italian set we see changes in consonant pronunciation, including grapheme-to-phoneme mapping ($c \rightarrow /CH/$) in Carcione and ($ch \rightarrow /K/$) in Cherico. We also see changes in vowel pronunciation: ($a \rightarrow /AA/$) in Carrara and Ruffalo; ($u \rightarrow /UW/$) in Cuttino and Lubrano; and ($o \rightarrow /OW/$) in Pesola and Romito. The Spanish set also shows changes in vowel pronunciation with Method B, for example ($i \rightarrow /IY/$) appears in Vivona and Zavadil.

Interestingly in Table 3.3 we see the test name Bencivenga is categorized as Spanish but appears with the grapheme-to-phoneme mapping ($c \rightarrow /CH/$), which is indicative of Italian origin. Searching the Spanish training set we find other examples of ($c \rightarrow /CH/$) mappings, indicating that Italian-origin names have been folded into Spanish data. This is not surprising since Spanish and Italian names have high confusion with each other, making it likely that a true Italian-origin name may be included in the Spanish training set. We also find true Slavic-origin surnames in the
Table 3.3: Examples of names in test set where Baseline and Method B systems output different phonemic transcriptions, and where Method B returns the gold-standard pronunciation. Language labels shown here (Italian, Slavic, Spanish) are computed as arg max_l P(l|w) using the character-based word origin model. Phoneme symbols are from an Arpabet-based alphabet, as used in the CMU Pronouncing Dictionary.
Spanish training set (e.g. Tencza and Lazarz), and in Table 3.3 the Slavic name Bednarz is listed as Spanish because it is similarly assigned a high probability of Spanish origin. Effectively our word origin model gave a Spanish G2P training set that is not composed purely of Spanish names, but the resulting G2P system is robust to these word origin errors.

Examining the Slavic set in Table 3.3, we see examples where the gold standard dictionary pronunciation is partially but not completely Americanized. In Jaworowski, we have mappings $j \rightarrow /Y/$ and $w \rightarrow /F/$ (second occurrence of $w$), both of which are derived from the original Polish pronunciation. But for the same name we also have $w \rightarrow /W/$ (first occurrence of $w$) rather than $w \rightarrow /V/$ where the latter is truer to the original Polish. Likewise in Wasik we see the Americanized $s \rightarrow /S/$ rather than $s \rightarrow /SH/$. One of our goals in this project is to capture these patterns of Americanization as they occur in the training set, and independent of how closely the training set approximates origin languages. We specifically study a set of US surnames where origin and degree of Americanization is not known a priori and so must be learned from the training data without external language-specific pronunciation resources.

### 3.5.2 Self-training for word origin modeling

We next examine the results of a character-based G2P system that applies self-training for the word origin component. Iterative self-training proceeds as follows: we start with the hand-labeled training set of 2,795 names, train a word origin model, automatically label the rest of our surname data set for word origin, and add high confidence data points ($P(l|w) > 0.999$) to the training set. After the word origin model converges we build language-specific G2P models as in the previous section. Results are shown in Table 3.4.

We see that Method D specifically gains 1% in accuracy over the previous set of experiments that do not use self-training. However the improvement of the best output is minimal (70.03% vs. 69.99%). We also note that training sets in 3.4 are larger than in Table 3.2 for many language groups, in particular the most frequent classes of British, German, and Italian, indicating that this word origin model has found more high confidence language labels with $P(l|w) > 0.7$. The end result, however, does not noticeably improve G2P conversion.
<table>
<thead>
<tr>
<th>Language Class</th>
<th>Train Count</th>
<th>Test Count</th>
<th>Baseline</th>
<th>(A)</th>
<th>(B)</th>
<th>(C)</th>
<th>(D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>British/English</td>
<td>18,072</td>
<td>2,251</td>
<td>70.68</td>
<td>72.19</td>
<td>72.41</td>
<td>72.41</td>
<td>72.41</td>
</tr>
<tr>
<td>German</td>
<td>9,002</td>
<td>1,133</td>
<td>75.82</td>
<td>72.20</td>
<td>77.93</td>
<td>77.76</td>
<td>78.11</td>
</tr>
<tr>
<td>Italian</td>
<td>3,897</td>
<td>489</td>
<td>60.94</td>
<td>66.46</td>
<td>65.85</td>
<td>67.48</td>
<td>65.24</td>
</tr>
<tr>
<td>Slavic</td>
<td>1,695</td>
<td>234</td>
<td>52.56</td>
<td>46.58</td>
<td>55.56</td>
<td>54.70</td>
<td>54.27</td>
</tr>
<tr>
<td>Spanish</td>
<td>1,258</td>
<td>188</td>
<td>44.68</td>
<td>39.36</td>
<td>46.28</td>
<td>45.74</td>
<td>47.34</td>
</tr>
<tr>
<td>French</td>
<td>983</td>
<td>129</td>
<td>42.64</td>
<td>40.31</td>
<td>44.19</td>
<td>44.19</td>
<td>43.41</td>
</tr>
<tr>
<td>Dutch</td>
<td>310</td>
<td>45</td>
<td><strong>66.67</strong></td>
<td>44.44</td>
<td>60.00</td>
<td>66.67</td>
<td>66.67</td>
</tr>
<tr>
<td>Scandinavian</td>
<td>317</td>
<td>42</td>
<td><strong>71.43</strong></td>
<td>59.52</td>
<td><strong>71.43</strong></td>
<td>69.05</td>
<td><strong>71.43</strong></td>
</tr>
<tr>
<td>Japanese</td>
<td>107</td>
<td>19</td>
<td><strong>78.95</strong></td>
<td>63.16</td>
<td><strong>78.95</strong></td>
<td>73.68</td>
<td><strong>78.95</strong></td>
</tr>
<tr>
<td>Arabic</td>
<td>40</td>
<td>9</td>
<td><strong>22.22</strong></td>
<td>11.11</td>
<td><strong>22.22</strong></td>
<td>22.22</td>
<td>22.22</td>
</tr>
<tr>
<td>Portuguese</td>
<td>20</td>
<td>2</td>
<td><strong>50.00</strong></td>
<td>0.00</td>
<td><strong>50.00</strong></td>
<td><strong>50.00</strong></td>
<td><strong>50.00</strong></td>
</tr>
<tr>
<td>Hungarian</td>
<td>18</td>
<td>2</td>
<td><strong>100.00</strong></td>
<td>0.00</td>
<td><strong>100.00</strong></td>
<td><strong>100.00</strong></td>
<td><strong>100.00</strong></td>
</tr>
<tr>
<td>Other</td>
<td>281</td>
<td>42</td>
<td>45.24</td>
<td>35.71</td>
<td>50.00</td>
<td>50.00</td>
<td><strong>52.38</strong></td>
</tr>
<tr>
<td>All</td>
<td>-</td>
<td>-</td>
<td>67.81</td>
<td>67.83</td>
<td>69.95</td>
<td><strong>70.03</strong></td>
<td>69.97</td>
</tr>
</tbody>
</table>

Table 3.4: G2P word accuracy for various weighting methods using a character-based word origin model with self-training. For Method B, $\sigma = 0.55$.

<table>
<thead>
<tr>
<th>Language Class</th>
<th>Train Count</th>
<th>Test Count</th>
<th>Baseline</th>
<th>(A)</th>
<th>(B)</th>
<th>(C)</th>
<th>(D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>British</td>
<td>16,829</td>
<td>2219</td>
<td>70.44</td>
<td>71.07</td>
<td>72.56</td>
<td>72.33</td>
<td>72.51</td>
</tr>
<tr>
<td>German</td>
<td>8,250</td>
<td>1124</td>
<td>75.89</td>
<td>72.42</td>
<td>77.05</td>
<td><strong>77.49</strong></td>
<td>77.14</td>
</tr>
<tr>
<td>Italian</td>
<td>3,558</td>
<td>481</td>
<td>60.50</td>
<td>64.45</td>
<td>65.90</td>
<td><strong>66.32</strong></td>
<td>64.03</td>
</tr>
<tr>
<td>Slavic</td>
<td>1,657</td>
<td>232</td>
<td>50.86</td>
<td>46.55</td>
<td><strong>52.16</strong></td>
<td>50.86</td>
<td>51.72</td>
</tr>
<tr>
<td>Spanish</td>
<td>1,042</td>
<td>185</td>
<td>47.03</td>
<td>34.05</td>
<td>45.95</td>
<td>47.57</td>
<td><strong>48.65</strong></td>
</tr>
<tr>
<td>French</td>
<td>912</td>
<td>142</td>
<td>45.77</td>
<td>36.62</td>
<td>47.18</td>
<td>46.48</td>
<td><strong>47.89</strong></td>
</tr>
<tr>
<td>Dutch</td>
<td>409</td>
<td>46</td>
<td><strong>69.57</strong></td>
<td>45.65</td>
<td>63.04</td>
<td>67.39</td>
<td><strong>69.57</strong></td>
</tr>
<tr>
<td>Scandinavian</td>
<td>362</td>
<td>63</td>
<td><strong>74.60</strong></td>
<td>63.49</td>
<td>73.02</td>
<td>73.02</td>
<td><strong>74.60</strong></td>
</tr>
<tr>
<td>Japanese</td>
<td>32</td>
<td>19</td>
<td>78.95</td>
<td>57.89</td>
<td><strong>78.95</strong></td>
<td><strong>78.95</strong></td>
<td><strong>78.95</strong></td>
</tr>
<tr>
<td>Arabic</td>
<td>25</td>
<td>9</td>
<td>22.22</td>
<td>44.44</td>
<td><strong>44.44</strong></td>
<td>33.33</td>
<td>33.33</td>
</tr>
<tr>
<td>Portuguese</td>
<td>29</td>
<td>3</td>
<td>66.67</td>
<td>0.00</td>
<td>66.67</td>
<td>66.67</td>
<td>66.67</td>
</tr>
<tr>
<td>Hungarian</td>
<td>13</td>
<td>2</td>
<td>100.00</td>
<td>50.00</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>Other</td>
<td>219</td>
<td>60</td>
<td>53.33</td>
<td>43.33</td>
<td>60.00</td>
<td>60.00</td>
<td><strong>61.67</strong></td>
</tr>
<tr>
<td>All</td>
<td>-</td>
<td>-</td>
<td>67.81</td>
<td>66.02</td>
<td>69.79</td>
<td><strong>69.83</strong></td>
<td>69.79</td>
</tr>
<tr>
<td>All - Cheat</td>
<td>-</td>
<td>-</td>
<td>67.81</td>
<td>69.22</td>
<td>71.23</td>
<td>71.08</td>
<td>69.79</td>
</tr>
</tbody>
</table>

Table 3.5: G2P word accuracy using a graphone-based word origin model. For Method B, $\sigma = 0.5$ ($\sigma = 0.45$ for “All - Cheat”).
3.5.3 Graphone-based word origin classification

Table 3.5 shows results for G2P conversion where Sequitur graphones are used for word origin modeling. In this approach, the graphone-based model is used to categorize names into language-specific G2P training sets. Then a character-based word origin model is used to predict $P(l|w)$ for the test set since phoneme (and hence graphone) information is unavailable for test data. Accuracy results do not improve on our first system (Section 3.5.1) that uses character n-grams exclusively for word origin, and Method A shows a noticeable 1.8% drop in accuracy from the baseline. After smoothing against the language-independent model (Methods B-D), accuracy is comparable to but not better than the systems using character-based word origin modeling.

As described in Section 3.4.1 we have previously found that Sequitur graphones can improve word origin labeling. We then seek to explain why this does not improve subsequent G2P conversion. We attribute this the fact that different word origin models are applied to train and test, and as a result word origin predictions between these two sets may be misaligned. To test this we implement a “Cheat” system where we use graphones to estimate language posteriors for both test and train data. Results are shown in the last row of Table 3.5. We see that the best result of “Cheat” improves on the best result of our previous G2P systems, leading us to conclude that further G2P gains may be found in improved word origin modeling. This can possibly be accomplished with additional features not investigated here, such as first names or other demographic information.

3.6 Unsupervised word origin clustering

Next we present our results for a G2P system that uses pseudo-language classes acquired from unsupervised hierarchical clustering of the training data set; these experiments do not use any word origin information from the hand-labeled US Census data set. We do clustering at $k = 5, 10, 20$ using both character and graphone n-grams as features. Results are shown in Table 3.6. As in previous experiments, Method A does worse than the baseline and Method B improves on it, showing that smoothing with the language-independent model is also useful in this unsupervised system.

We next examine results for the character-based system with $k = 20$ by language class and by cluster. Table 3.7 shows accuracy results by language class. Since this system does not naturally give us a language-labeling of the data we use the labels assigned in Table 3.2; note that these labels were not used at any step in G2P conversion in this section. A noticeable difference with
Table 3.6: Accuracy results using unsupervised word origin clustering, with number of clusters \(k = 5, 10, 20\) and Methods (A) and (B) of weighting model specific results. As before, accuracy on a language specific baseline is 67.81%; numbers in parenthesis show the absolute change in accuracy from this baseline.

These experiments is that the Italian set shows lower accuracy with Methods A and B compared to the baseline; in previous experiments Italian is the one language class that regularly improved with any language-specific modeling. British, German, and Slavic do show improvements. Table 3.7 also shows how each language class is distributed among the \(k = 20\) clusters. We see that the Italian set is fairly well divided among clusters, in contrast with German and Slavic where a single cluster label covers over 40% of names. British names seem to be also well distributed among clusters; however accuracy does not fall as with Italian which we attribute to the fact that more British training data is available.

Table 3.8 shows the same results listed by the cluster label assigned rather than language class. For each cluster we similarly show the three most frequent language classes represented, again using the language labels from Table 3.2. We see that British and German repeatedly appear as the most frequent language in a cluster, and Slavic is the only other language label to cover the majority of a cluster. We also see that at the level of individual clusters, Method B consistently improves on the baseline for all but the smallest clusters. So while the unsupervised word origin model has failed to discover language groupings, the output cluster labels do improve G2P conversion on our test set.

### 3.7 Conclusion

In this chapter we have applied word origin modeling to G2P conversion, using both supervised and unsupervised methods. We see that with our data set of US Surnames, word origin labeling is often ambiguous and name pronunciation is affected by Americanization to varying degrees. We find that
<table>
<thead>
<tr>
<th>Lang. Class</th>
<th>Train Count</th>
<th>Test Count</th>
<th>Baseline Count</th>
<th>(A)</th>
<th>(B)</th>
<th>Cluster Distribution (3 most frequent)</th>
</tr>
</thead>
<tbody>
<tr>
<td>British</td>
<td>16,171</td>
<td>2,111</td>
<td>71.81</td>
<td>65.89</td>
<td><strong>73.24</strong></td>
<td>B(21),A(18),D(10),F(10)</td>
</tr>
<tr>
<td>German</td>
<td>8,360</td>
<td>1,109</td>
<td>75.83</td>
<td>69.70</td>
<td><strong>76.56</strong></td>
<td>A(45), B(12), E(8)</td>
</tr>
<tr>
<td>Italian</td>
<td>3,358</td>
<td>447</td>
<td><strong>61.74</strong></td>
<td>51.90</td>
<td>61.30</td>
<td>G(22), B(14), H(13)</td>
</tr>
<tr>
<td>Slavic</td>
<td>1,658</td>
<td>232</td>
<td>50.86</td>
<td>52.16</td>
<td><strong>52.59</strong></td>
<td>I(42), K(12), A(10)</td>
</tr>
<tr>
<td>Spanish</td>
<td>1,460</td>
<td>246</td>
<td>44.72</td>
<td>41.06</td>
<td><strong>46.75</strong></td>
<td>C(17), G(14), H(14)</td>
</tr>
<tr>
<td>French</td>
<td>1,143</td>
<td>177</td>
<td><strong>42.94</strong></td>
<td>38.42</td>
<td>41.81</td>
<td>A(16), B(16), C(14)</td>
</tr>
<tr>
<td>Dutch</td>
<td>468</td>
<td>82</td>
<td><strong>70.73</strong></td>
<td>58.54</td>
<td><strong>70.73</strong></td>
<td>B(46), A(21), C(8)</td>
</tr>
<tr>
<td>Scand.</td>
<td>393</td>
<td>61</td>
<td><strong>77.05</strong></td>
<td>68.85</td>
<td><strong>77.05</strong></td>
<td>B(36), A(30), H(14)</td>
</tr>
<tr>
<td>Japanese</td>
<td>116</td>
<td>23</td>
<td>73.91</td>
<td>52.17</td>
<td>73.91</td>
<td>I(26), A(20), H(18)</td>
</tr>
<tr>
<td>Arabic</td>
<td>68</td>
<td>18</td>
<td>33.33</td>
<td>22.22</td>
<td><strong>44.44</strong></td>
<td>A(22), D(19), B(18)</td>
</tr>
<tr>
<td>Portug.</td>
<td>34</td>
<td>4</td>
<td>25.00</td>
<td>50.00</td>
<td>50.00</td>
<td>H(27), C(13), A(13)</td>
</tr>
<tr>
<td>Hung.</td>
<td>28</td>
<td>3</td>
<td>100.00</td>
<td>66.67</td>
<td>100.00</td>
<td>C(32), B(18), A(14)</td>
</tr>
<tr>
<td>Other</td>
<td>431</td>
<td>72</td>
<td>55.56</td>
<td>51.39</td>
<td>58.33</td>
<td>B(27), A(20), C(8), D(8)</td>
</tr>
</tbody>
</table>

**Table 3.7:** G2P word accuracy using an unsupervised character-based word origin model. For Method B, $\sigma = 0.65$ based on tuning on held-out data set.

At each row for language $l$, rightmost column shows the 3 most frequent clusters that are represented in the training set of $l$, labeled as A through T as shown in Table 3.8. Values in parenthesis shows percentage of train data for $l$ assigned to each cluster.
Table 3.8: G2P word accuracy using an unsupervised character-based word origin model, with accuracy shown for $k = 20$ clusters labeled as A through T by decreasing training set size. Rightmost column shows three most frequent language classes represented in each cluster, with percentage of training set size shown in parenthesis. Language abbreviations are: 'Br':British, 'Gr':German, 'It':Italian, 'Sl':Slavic, 'Sp':Spanish, 'Fr':French.
G2P conversion using only language-specific models often performs below baseline; however we see 1-2% improvement when we incorporate smoothing between language-specific and language-independent models. We also present a system that uses an unsupervised word origin model, and this system also improves on the language-independent baseline. Further improvements may be found with improved word origin labeling, possibly using additional features not explored here.

Our work varies from previous research in two ways. First, we present methods that are not tied to any specific G2P algorithm, but rather can be applied to any G2P system that outputs posteriors $P(\phi|\bar{g})$. Second, for experiments we choose a data set that naturally contains names of varied and unknown languages of origin, and consider how the output of language-specific and language-independent G2P models can be combined to overcome the problem of language sparsity for rare classes. As in the previous chapter our goal is to apply word origin modeling to problems and data sets where origin languages are not known \textit{a priori} and where language-specific pronunciation resources are limited or unavailable.
CHAPTER 4
NAMED ENTITY RECOGNITION FOR DISEASES

The rapid expansion of published literature in biology and medicine presents challenges for researchers in these fields who need to search and process new information. Names of genes and proteins are regularly introduced into the vocabulary and, additionally, natural lexical variation causes existing concepts to appear in new and unseen forms. Tools for text mining and information extraction can assist in processing of biomedical literature, and this has become a major area of study within natural language processing.

In this chapter we focus on named entity (NE) recognition, the automated process of finding mentions of specific types of entities in text. Named entity recognition in biomedical text processing tends to be separated into two steps: tagging of the term occurrence in a text or abstract, followed by mapping or normalization of the NE mention into a database of known terms. Previous research has focused on protein and gene mentions, and less attention has been paid to the classification and mapping of diseases. In this chapter we study named entity recognition for disease terms, and specifically consider how word-internal features including morphology can be applied in a supervised machine learning approach.

In Section 4.1 we present previous works that focus on disease mention recognition, followed by works that incorporate word-internal features for gene and protein recognition. We then present available data sets for the study of disease mentions in Section 4.2. In Section 4.3 we present our methods for incorporating morphological information into named entity tagging of diseases. Specifically we show how we extract morphology information using the Linguistica toolkit and include this as features in a Conditional Random Field (CRF) model for named entity tagging. Experiments and results are discussed in Section 4.4.
4.1 Related Works

4.1.1 Disease tagging with Conditional Random Fields

[Jin et al. 2006] applies Conditional Random Fields (CRFs) to disease recognition. CRFs [Lafferty et al. 2001] are probabilistic models that learn the conditional probability of a tag sequence $T = t_1 \ldots t_n$ given the observed token sequence $O = o_1 \ldots o_n$. A linear-chain CRF assumes a first-order dependency between tag sequences; in this case the conditional probability of the tag sequence given the observation sequence is represented as:

$$P(T|O) = \frac{e^{\sum_j \sum_i \lambda_i f_i(t_i,t_{i-1},O,j)}}{Z(O)}$$

Here each $f_i(t_i,t_{i-1},O,j)$ is a feature function measuring feature $i$ with tags $t_j$ and $t_{j-1}$ at the current and previous tokens, at position $j$ in the observation sequence. $\lambda_i$ is the learned feature weight and $Z(O)$ is a normalization function:

$$Z(O) = \sum_T e^{\sum_j \sum_i \lambda_i f_i(t_i,t_{i-1},O,j)}$$

to ensure that the distribution sums to 1. In [Jin et al. 2006], the token sequence is a sequence of words in a text, and possible tags are $\{\text{Malignancy, Other}\}$, indicating whether the word is part of a malignancy mention. An example feature function is given as:

$$f_i(t_i,t_{i-1},O,j) = \begin{cases} 1 : & t_j = \text{Malignancy} \land t_{j-1} = \text{Malignancy}\land o_j = "cancer" \land o_{j-1} = "lung" \\ 0 : & \text{otherwise} \end{cases}$$

which indicates if the two-word term “lung cancer” is tagged as a malignancy. CRFs are considered to be well-suited for sequential tagging problems since, unlike generative models such as HMMs, they allow overlapping, non-independent features in the observation sequence.

A range of features are used in [Jin et al. 2006], and are listed in these categories:

- **Word-based** - Occurrence of word in malignancy mention in the training data.
- **character n-gram-based** - Frequency of character n-grams (orders 2-4) in malignancy mentions in training data.
The CRF method showed a considerable improvement in recall over exact string matching. The majority of mentions detected by the CRF tagger but missed by string matching were due to acronyms and lexical variations (e.g., “leukaemia” versus “leukemia”). In addition, a significant subset of these misses is from terms that were not represented in the training lexicon, such as “temporal lobe benign capillary haemangioblastoma”, indicating that the tagger was able to learn to recognize new malignancy terms. False positives detected by the CRF are attributed to inability to correctly detect boundaries (e.g. “neoplasm” instead of “brain neoplasm”) and character overlap with true malignancy terms (e.g., “opl” in “brain neoplasm” and “neoplastic”).

In a final experiment the tagger was applied to 15 million medical abstracts, from the MEDLINE database. It was found that the ratio of new malignancy mentions per abstract was relatively consistent, from 0.183 new mentions per abstract in the first 0.1% of documents to 0.038 new mentions in the last 1%. The demonstrates both the tagger’s ability to detect new and unique mentions, as well as the frequency of new mentions in the MEDLINE dataset. In total 9,153,340 mentions and 580,002 unique mentions were tagged.

[Chowdhury and Lavelli 2010] also applies CRFs to tagging of disease mentions, using a similar set of lexical and orthographic features. More structured morphological features are not applied. The best reported F-measure on the AZDC corpus, described in the next section, is 81.08% although train and test data sets are not distributed. The authors note that featured previously used in other works for gene and protein recognition, such digits and capitalization, were found to be not as effective with disease tagging.

4.1.2 MetaMap

One extensive and publicly available system for biomedical named entity recognition is MetaMap [Aronson 2001], which tags NEs and maps them to terms in the Unified Medical Language System (UMLS) Metathesaurus, an extensive biomedical database made available by the National Library of Medicine (NLM). The MetaMap system discovers noun phrases in the input text using a shallow parser and then generates variants of these phrases that can be matched with the UMLS vocabulary. Variant phrases are generated using information about acronyms, abbreviations, synonyms,
derivation, inflectional and spelling variation. This information is acquired from the SPECIALIST lexicon, an online syntactic lexicon of biomedical and general English.

### 4.1.3 Other methods for disease tagging

[Jimeno et al. 2008] applies MetaMap to diseases and compares this system to two different methods: exact string matching using a dictionary, and a statistical information-theory-based method previously applied to gene recognition [Gaudan et al. 2008]. The third system scores terms based on the information content of its tokens and uses words as its base units. All three systems are evaluated with a corpus of MEDLINE abstracts for both the tagging and normalization subtasks. Results show that the dictionary lookup method alone provides competitive results, and a voting combination of the three methods provides further improvement.

[Névéol et al. 2009] also studies the mapping task on the same data set as [Jimeno et al. 2008] and presents their own statistical system based on the Priority Model of [Tanabe and Wilbur 2006], which has previously been applied to gene and protein classification. This model scores terms based on the frequency of tokens in disease and non-disease noun phrases. Results from the Priority Model are comparable to results from MetaMap with adjusted parameters, with the Priority Model showing higher F-measure and recall. The Priority Model was found to perform better on an input model of non-sentence queries, showing that it is more robust when sentence context is not available.

At the document level, [Srinivasan and Wedemeyer 2003] and [Srinivasan et al. 2004] present applications of disease mentions in IR tasks. The first studies global trends in disease research, and the second discovers a connection in literature between turmeric and retinal diseases. Note that neither of these works do text-level extraction, rather they use manually assigned Medical Subject Headings (MeSH) to create a vector representation of documents.

### 4.1.4 Gene and protein named entity tagging with morphology

Similar to [Jin et al. 2006], [McDonald and Pereira 2005] applies CRFs to tagging of gene and protein mentions. Features include word and spelling (use of capitalization, digits, and punctuation), character n-grams (orders 2-4) and prefix and suffixes, also of length 2-4. Features were included for the current, previous, and next word token. In a second model a feature was added to indicate a term occurred in a gene lexicon; for a multi-word gene term all tokens must appear in
the lexicon entry for this feature to be positive. Use of a lexicon showed a significant improvement in precision and recall.

[Yamamoto et al. 2003] specifically applies morphological analysis to protein name tagging. The authors note that the space character is not a sufficient delimiter in this domain, for example “ERK activator kinase 1” is treated as a single term. For morphological analysis, the authors use “common prefix search”, a technique already applied to non-segmented languages including Japanese. This method segments words according to whole word units from a dictionary lookup, as a result morphemes in this work may not correspond to the most atomic segmentation by linguistic standards. For example, “associated” is treated as a single morpheme, in the phrase “SLP-76-associated substrate”.

HMMs are a common solution to the tagging of multi-word named entities. [Collier et al. 2000] classifies named entities into one of ten predefined classes, including proteins, cell lines, and viruses using an HMM classifier over words. A range of word-level features are included, such as punctuation, digits, and capitals. To overcome data sparseness, classification is done using a linear interpolation of a range of feature models. [Zhou et al. 2004] includes additional features such as POS and head noun, and uses a k-nearest neighbors (k-NN) algorithm to overcome sparsity; a comparison on the GENIA corpus of biomedical abstracts shows improvement over [Collier et al. 2000].

[Shen et al. 2003] adapts an HMM-based NE tagger to the biomedical domain using a range of features including morphology and speech. Suffixes common to organic compounds are identified by hand and grouped together, for example “-cin”, “-mid”, and “-zole” are grouped as being frequent for organic compounds and infrequent in other NE classes. Of the 100 most frequent suffixes, 37 are chosen by hand to be grouped into 23 feature sets; the authors do not specify how the initial set of 100 suffixes is acquired.

[Torii et al. 2004] uses a combination of word-internal and -external features to classify an identified named entity by type, i.e., protein, virus, cell type, etc. For internal features, the authors note that suffix features provide varying levels of information gain for type classification, and that some suffixes may be substrings of others. For example, ine is a common suffix with low information gain, while kine tends to indicate a protein name. Decision list classification is used to identify informative suffixes. Additionally, approximate string matching is used for classification for unseen terms, i.e., the presence of the term “CD28” can be used to classify the unknown
None of the 13 families with cases of male breast cancer appear to be linked, but it is estimated that 92% (95% confidence interval 76%-100%) of families with no male breast cancer and with two or more ovarian cancers are linked to BRCA1.

In the present study, seven mutations in the PDS gene (PDS), the gene responsible for Pendred syndrome, have been found in families of non-syndromic sensorineural hearing loss with EVA.

Table 4.1: Sample sentences from Arizona Disease Corpus (AZDC) corpus. Disease mentions are shown in bold.

<table>
<thead>
<tr>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>None of the 13 families with cases of male breast cancer appear to be linked, but it is estimated that 92% (95% confidence interval 76%-100%) of families with no male breast cancer and with two or more ovarian cancers are linked to BRCA1.</td>
</tr>
<tr>
<td>In the present study, seven mutations in the PDS gene (PDS), the gene responsible for Pendred syndrome, have been found in families of non-syndromic sensorineural hearing loss with EVA.</td>
</tr>
</tbody>
</table>

4.2 Data sets for disease mentions

There is a limited amount of annotated data available for the study of disease named entity recognition in research literature. The data set from [Jimeno et al. 2008], also used in [Névéol et al. 2009], is publicly available and contains 551 MEDLINE sentences annotated with UMLS concepts or concept clusters. [Jin et al. 2006] uses the Penn BioIE corpus (v.0.9) containing 1,442 abstracts, and this data set is also publicly available.

A more recent corpus of disease mentions is the Arizona Disease Corpus (AZDC) [Leaman et al. 2009], a corpus of 793 MEDLINE abstracts covering 2784 sentences that are hand-annotated for disease mentions. Each mention is tagged by its start and end point and associated with a specific UMLS concept ID. Duplicate mentions are allowed within a single sentence. Spans are chosen so as to cover the minimum number of tokens needed to identify the most specific form of the disease. For example, in the first example in Table 4.1, “male breast cancer” is chosen over the phrase “breast cancer” as it can be linked to the specific UMLS concept “Carcinoma of male breast”. We use AZDC for our main experiments in this chapter. AZDC is also the corpus used in [Chowdhury and Lavelli 2010].

We also evaluate our model on the CALBC corpus, a large-scale biomedical corpus with annotations for multiple types of named entities including genes and diseases [Rebholz-Schuhmann et al. 2010]. CALBC is a “Silver standard”, in that annotations are not acquired by-hand but rather from the harmonized results of multiple automatic annotation systems. We use CALBC Phase-I, a
corpus of 50k MEDLINE abstracts annotated by five automatic annotation systems (EMBL-EBI, EMC Rotterdam, NLM, JULIE Lab Jena, and Linguamatics). In this report we use a training set of 15k sentences and a test set of 5k sentences.

### 4.3 Methods

In this project we focus on the tagging phase of named entity recognition. We examine tagging of disease mentions, as this is an area that has received less focus than gene and protein tagging. Compared to genes, disease mentions are less likely to be marked by non-standard spelling, case, and punctuation, and we expect that morphological features will be valuable for improving recognition results.

We follow the model of [Jin et al. 2006] and [Chowdhury and Lavelli 2010], which apply CRFs to disease (or malignancy) mentions. Here we first present our methods for incorporating morphological information with the Linguistica toolkit, and then present lexical features that we also incorporate into our model. Finally, we present our results on the AZDC disease corpus, as well as experimental results on a “Silver Standard” corpus containing multiple named entity types including genes and diseases.

To incorporate morphology, we will use features acquired from Linguistica when applied to a list of known disease terms and a more general biomedical data set. As described in Section 4.1, previous works for gene tagging have used prefix and suffix information, but either a predefined suffix list was built or the feature set included all observed prefixes and suffixes up to a fixed length. By using Linguistica features we can easily include longer prefixes and suffixes without an intractable increase in the size of the feature set.

#### 4.3.1 Formation of a disease term list from SNOMED

To extract Linguistica features, we first build a disease term list from the Systematized Nomenclature of Medicine Clinical Terms (SNOMED). SNOMED is a comprehensive clinical vocabulary managed by an international organization, of which the US National Library of Medicine (NLM) is a charter member. SNOMED is a hierarchical database, and its core elements are concepts, descriptions, and relationships. Concepts are the central entities to SNOMED and are described by one or more Descriptions which provide alternate phrasings of a concept that may appear in natural
text. For example, “myocardial infarction”, “cardiac infarction,” “heart attack,” and “infarction of heart” all describe the same concept.

We use SNOMED to build two disease-related lists. First, we build a list of terms which are descriptions associated with disease concepts; these terms may be composed of one or more words. Next we also build a list of words that appear at least once in any disease term. This is our disease word list that is used for morphological analysis. We follow these steps:

1. Construct Disease Term List by extracting single- and multi-word term labeled as a “disorder” in SNOMED
2. Construct Disease Word List, a list of unique words that appear in any term in the Disease Term List
3. Filter Disease Word List: Remove words that are not “natural English” and that may introduce error into morphological analysis. Specifically we remove words with numbers and special characters, and words composed of a series of upper-case characters indicating an acronym. We also exclude words that are fewer than six characters in length, as our goal is to examine polymorphemic words.

This disease word list is used for morphological analysis with the Linguistica toolkit, which we describe in the next section.

4.3.2 Linguistica toolkit

Morphological analysis can be acquired with Linguistica, a software tool for the unsupervised discovery of morphology [Goldsmith 2001], [Goldsmith 2006]. The goal of Linguistica is to produce an analysis similar to that which would be produced by a human morphologist. We say Linguistica is unsupervised because the only input is a text corpus, and no dictionary or morphological rules are known in advance. As such, Linguistica can analyze languages where such resources are unavailable. In our case, Linguistica can be applied to domain-specific texts where lexical resources are incomplete.

The output analysis of Linguistica has three sets: a set of suffixes, a set of stems, and a set of signatures. The signature is a concept introduced in [Goldsmith 2001]; a signature is a collection of stems and the suffixes that appear with these stems in the input corpus. Examples of signatures
Table 4.2: Examples of signatures learned by Linguistica from *Tom Sawyer*, from [Goldsmith 2001]

<table>
<thead>
<tr>
<th>Signature</th>
<th>Example</th>
<th>Stem Count</th>
<th>Token Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL.ed.ing</td>
<td>betray betrayed betraying</td>
<td>69</td>
<td>864</td>
</tr>
<tr>
<td>NULL.ed.ing.s</td>
<td>remain remained remaining</td>
<td>14</td>
<td>516</td>
</tr>
<tr>
<td>NULL.s</td>
<td>cow cows</td>
<td>253</td>
<td>3,414</td>
</tr>
<tr>
<td>e.ed.es.ing</td>
<td>notice noticed noticing</td>
<td>4</td>
<td>62</td>
</tr>
</tbody>
</table>

discovered from standard English texts are shown in Table 4.2, where we see that the stem *betray* appears with the suffixes *-ing*, *-ed*, and *NULL*, where NULL is the empty string. The signature representation also provides us with a morphological segmentation of analyzed words into its stem and suffix parts.

The Linguistica framework has two components: (1) a set of heuristics that proposes a signature-based morphological analysis of the input, and (2) an evaluation component that decides if modifications from the heuristics should be adopted. The evaluation component is based on the Minimum Description Length (MDL) principal [Rissanen 1989], an analytical approach to statistical model selection based in information theory. MDL proposes that the optimal linguistic analysis is the one that minimizes the length of the compressed data, including the length of the model itself; this is the *description length* of the corpus. For Corpus $C$ and Model $M$, description length is computed as:

$$DL(C, M) = length(M) - \log_2(prob(C|M))$$

Here model $M$ is the sets of stems, affixes, and signatures. Corpus $C$ is represented as the original text file with analyzed words replaced with pointers to component stems and affixes. Initially when the signature set is empty the model is uncompressed. Through iterative application of heuristics and MDL evaluation, a signature set is composed and DL is optimized. Details of these heuristics are described in [Goldsmith 2006].

Linguistica is implemented in C++ as a stand-alone, open-source software package. Reasonable results are possible on files of as little as 5,000 words, and input size can reach up to the tens of millions of words.
4.3.3 Morphological analysis of SNOMED terms with Linguistica

First, we analyze terms to obtain their signature analysis in which each term is assigned exactly one stem and one suffix, where the suffix may be NULL indicating the empty string. Of unique 15,345 words in the disease words list, 10,311 were found to have a signature analysis with Linguistica. Table and 4.3 show a list of the most frequent suffixes, as well as some less frequent stems that are highly indicative of disease terminology.

A frequent occurrence of the signature analysis is that a single stem may be further decomposed into other known morphemes. For instance, *spondylarthritis* is encoded with the signature analysis *spondylarthr + itis*; however as both *spondyl* and *arthr* are discovered as stems so a natural extension would be to further decompose *spondylarthr* into these two subparts. We also include this type of stem segmentation as a feature in our tagger, with the motivation that it may be useful for recognizing disease terms. We next present our methods and analysis for incorporating stem decomposition into the initial signature analysis output by Linguistica. Our problem is analogous to general morphological decompounding, where a compound word is decomposed into component words, but with the extension that we are decomposing morphemes into other morphemes that may not occur as whole uninflected words.

Previous research in morphological decompounding has in part focused on selecting one of a set of possible and semantically distinct segmentations. For example, [Marek 2006] notes that the German word *gelbrot* may be split as *gelb rot* (yellow-red) or *Gel brot* (gel bread). We find that similar examples of semantic ambiguity are difficult to identify in our disease term list. We find only one case where a single stem can be split into Linguistica-discovered sub-stems at different breakpoints: *cholester* may be split as *chol ester* or *chole ster*. Ambiguity instead arises when determining whether a parse should be split at all, for example a reasonable segmentation of *cholester* is to keep the stem whole. (In the case of this example, *chole ster* is a valid decomposition).

Of the 10,311 words that are assigned a signature analysis, 690 may be further decomposed using a greedy approach that splits stems whenever possible based on the set of already discovered morphemes. Examining a list of 100 words selected at random, we find only two examples where over-splitting leads to possible ambiguous interpretation: *read miss ion* from *readmiss ion* and *illeg al ly* from *illegal ly*. Over-splitting is not observed in biomedical-specific words. However, we do note cases where a stem split is semantically correct but the original stem may be a more salient feature for supervised classification. For example the stem *lipoprotein* may be more meaningful as
<table>
<thead>
<tr>
<th>Rank</th>
<th>Suffix</th>
<th>Count</th>
<th>Exemplar</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>al</td>
<td>988</td>
<td>abdomin al, duct al</td>
</tr>
<tr>
<td>2</td>
<td>ic</td>
<td>814</td>
<td>aerob ic, extrahepat ic</td>
</tr>
<tr>
<td>3</td>
<td>NULL (no suffix)</td>
<td>679</td>
<td>lipoprotein, phlebolith</td>
</tr>
<tr>
<td>4</td>
<td>ia</td>
<td>677</td>
<td>croyglobulinem ia, hypercapn ia</td>
</tr>
<tr>
<td>5</td>
<td>us</td>
<td>621</td>
<td>enanthemato us, fibro us</td>
</tr>
<tr>
<td>6</td>
<td>um</td>
<td>468</td>
<td>adenophor um, caloric um</td>
</tr>
<tr>
<td>7</td>
<td>is</td>
<td>423</td>
<td>nasal is, cerebral is</td>
</tr>
<tr>
<td>8</td>
<td>osis</td>
<td>408</td>
<td>endotoxic osis, neurobartonell osis</td>
</tr>
<tr>
<td>9</td>
<td>ing</td>
<td>375</td>
<td>bleed ing, function ing</td>
</tr>
<tr>
<td>10</td>
<td>s</td>
<td>361</td>
<td>insulin s, pancreas s</td>
</tr>
<tr>
<td>11</td>
<td>ed</td>
<td>356</td>
<td>inflam ed, polybrominat ed</td>
</tr>
<tr>
<td>12</td>
<td>itis</td>
<td>331</td>
<td>cholecyst itis, osteoperiost itis</td>
</tr>
<tr>
<td>13</td>
<td>tion</td>
<td>331</td>
<td>epidermidiza tion, lichenifica tion</td>
</tr>
<tr>
<td>14</td>
<td>es</td>
<td>249</td>
<td>cartilag es, lichen oid es</td>
</tr>
<tr>
<td>15</td>
<td>ase</td>
<td>227</td>
<td>allantoic ase, nucleotid ase</td>
</tr>
<tr>
<td>16</td>
<td>ate</td>
<td>216</td>
<td>chlor ate, nicotin ate</td>
</tr>
<tr>
<td>17</td>
<td>oma</td>
<td>187</td>
<td>blast oma, carcin oma, gatrin oma</td>
</tr>
<tr>
<td>18</td>
<td>a</td>
<td>154</td>
<td>myoclonic a, plasmodi a</td>
</tr>
<tr>
<td>19</td>
<td>oid</td>
<td>148</td>
<td>fibromyx oid, retin oid</td>
</tr>
<tr>
<td>20</td>
<td>e</td>
<td>141</td>
<td>autosom e, parasit e</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>matosis</td>
<td>16</td>
<td>adeno matosis, lipo matosis</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>pox</td>
<td>13</td>
<td>chickenpox, rabbitpox</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>plasia</td>
<td>11</td>
<td>fibro plasia, neo plasia</td>
</tr>
<tr>
<td>83</td>
<td>plastic</td>
<td>11</td>
<td>hypo plastic, kerato plastic</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>153</td>
<td>viral</td>
<td>5</td>
<td>filo viral, papilloma viral</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>232</td>
<td>renal</td>
<td>3</td>
<td>extra renal, pre renal</td>
</tr>
<tr>
<td>233</td>
<td>retinol</td>
<td>3</td>
<td>extra retinal, intra retinal</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>325</td>
<td>genesis</td>
<td>1</td>
<td>hypo genesis</td>
</tr>
</tbody>
</table>

Table 4.3: Subset of suffixes discovered via initial signature analysis of disease word list. Count is the number of words analyzed with this suffix. Ranking is by count.
Table 4.4: Example of signatures learned from Snomed disease word list

<table>
<thead>
<tr>
<th>Signature</th>
<th>Example</th>
<th>Stem Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL.s</td>
<td>allergen adenoid cancer disinfectant steroid</td>
<td>283</td>
</tr>
<tr>
<td>ia.ic</td>
<td>atax dermatopath ectop myop neuralg</td>
<td>55</td>
</tr>
<tr>
<td>NULL.is</td>
<td>cevical cranial glandular macular stercoral</td>
<td>50</td>
</tr>
<tr>
<td>ed.ing</td>
<td>heal evolv burn swallo mottl scarr imag</td>
<td>34</td>
</tr>
<tr>
<td>osis otic</td>
<td>cyan hypomelan ket narc osteopor scoli</td>
<td>28</td>
</tr>
<tr>
<td>a osis</td>
<td>histoplasm salmonell protothe toxoplasm</td>
<td>19</td>
</tr>
</tbody>
</table>

To account for this we will include morpheme bigrams as features. That is, for a word \( w \) with segmentation \( w = m_0, \ldots, m_n \), we include bigrams \( m_0m_1, \ldots, m_{n-1}m_n \) as model features; unigram morphemes \( m_0, \ldots, m_n \) are also included in the feature set. Morpheme bigrams may cross the original stem-suffix boundary. For example "adenolymphoma" (Stem: "adenolymph", Suffix: "oma") from Table 4.5 is assigned unigram morpheme features "adeno", "lymph" and "oma" and bigram features "adenolymph" and "lymphoma".

To evaluate the utility of this segmentation of stems into sub-stems, we also measure the expected mutual information (EMI) of Linguistica-based features discovered in the AZDC corpus. Expected mutual information, or information gain, is a measure of reduction in uncertainty of one variable given another variable. Higher mutual information indicates a larger reduction in uncertainty. As presented in [Manning et al. 2008] and [Sebastiani 2002], we can measure expected mutual information of feature \( f_i \) as:

\[
\sum_{f \in \{f_i, \bar{f}_i\}} \sum_{l \in \{l_j, \bar{l}_j\}} P(f, l) \log \frac{P(f, l)}{P(f)P(l)}
\]  

(4.1)

Here \( f_i, \bar{f}_i \) are the events that morpheme feature \( f_i \) does or does not occur for a given token, and \( l_j, \bar{l}_j \) are the events that the token is or is not labeled as part of a disease term. In measuring EMI we consider stems and suffixed obtained from the initial signature analysis, as well as unigram and bigram morphemes obtained from stem segmentation. Table 4.6 shows some of these morpheme features, with EMI scores and example tokens where they appear. In our data set we discover 59 features that are not already represented as a stem or suffix feature from signature analysis. This includes morpheme bigrams that cross the stem-suffix boundary in some tokens (e.g. "epithelial"), and unigram morphemes that only occur in the AZDC corpus as a segment in a compound stem
### Table 4.5: Examples of words where further stem segmentation is possible. The initial signature analysis is shown on the left, and results of further stem segmentation are shown on the right.

<table>
<thead>
<tr>
<th>Stem segmentation</th>
<th>Signature analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>adeno lymph oma</td>
<td>adenolymph oma</td>
</tr>
<tr>
<td>adeno phor um</td>
<td>adenophor um</td>
</tr>
<tr>
<td>adeno tonsill itis</td>
<td>adenotonsill itis</td>
</tr>
<tr>
<td>chole cyst itis</td>
<td>cholecyst itis</td>
</tr>
<tr>
<td>develop ment al</td>
<td>development al</td>
</tr>
<tr>
<td>extra cutane ous</td>
<td>extra cutaneous</td>
</tr>
<tr>
<td>fibro follicul oma</td>
<td>fibrofollicul oma</td>
</tr>
<tr>
<td>fibro follicul oma s</td>
<td>fibrofolliculoma s</td>
</tr>
<tr>
<td>hyper bilirubin emia</td>
<td>hyperbilirubin emia</td>
</tr>
<tr>
<td>hyper cement osis</td>
<td>hypercement osis</td>
</tr>
<tr>
<td>hyper chylomicronem ia</td>
<td>hyperchylomicronem ia</td>
</tr>
<tr>
<td>hyper pigment ation</td>
<td>hyper pigmentation</td>
</tr>
<tr>
<td>neuro toxic osis</td>
<td>neurotoxic osis</td>
</tr>
<tr>
<td>papill oma tous</td>
<td>papilloma tous</td>
</tr>
<tr>
<td>psycho physiolog ic</td>
<td>psychophysiolg ic</td>
</tr>
<tr>
<td>teta nitr ate</td>
<td>tetraneit rat</td>
</tr>
</tbody>
</table>

(e.g. fibro). We also discover morphemes that do occur in the initial signature analysis, but segmentation discovers other instances and increases EMI scores. For example, amyl increase from 0.12 to 1.66 due to its discovery as a sub-stem in amyloidosis after stem segmentation.

#### 4.3.4 CRF model and features

Next we present our feature-based model for named entity recognition of diseases. We apply a CRF model over tokenized words where each sentence is represented as a single sequence. We use the BIO tag set commonly used for named entity recognition: B indicates the start of a named entity term, I indicates an internal token, and O indicates that a token is outside of any named entity. For example, the five token phrase “cases of male breast cancer” from Table 4.1 would have the tag sequence (cases- O, of-O, male- B, breast- I, cancer- I). As a pre-processing step, data is tokenized and tagged for Part-of-Speech (POS) using the Stanford tagger with the left-three-word model.

We create an initial feature set using features motivated by [Chowdhury and Lavelli 2010] and [Hsu et al. 2008], previous works that apply CRFs to disease and gene recognition, respec-
Table 4.6: Examples of morpheme features extracted from the AZDC corpus by segmentation of stems into sub-stems. Column “Example token” shows tokens where this feature occurs. Symbol ’+’ indicates the location of the stem-suffix split in the initial signature analysis.

- **Token**: This feature set includes: (1) Unaltered token, (2) Normalized token, (3) Stemmed token, and (4) Part-of-Speech (POS). The following transformations are performed to acquire (2) Normalized token:
  - Convert upper to lower case
  - Replace Greek letters with single ’G’ character
  - Replace digit characters 0-9 with ’9’

The Stanford tagger, which is used for tokenization, does automatic “Americanization” of British spellings of words. For example, *tumour* is automatically converted to *tumor*. We use the Americanized spelling as input to all feature functions presented here.

- **Orthographic**: Length of word and various additional binary features that are listed in Table 4.7.

- **Character**: Features that use characters sub-sequences of the word. These include suffixes, prefixes, and internal n-grams.

- **Dependency Parse**: Governing token and relationship type, acquired from a statistical dependency parser. We apply the Stanford statistical parser [Marie-Catherine de Marneffe
and Manning 2006] to extract dependency parse information for each sentence. Following [Chowdhury and Lavelli 2010] we only use the following dependency types: direct object, indirect object, noun compound modifier, nominal subject, and passive nominal subject.

After a baseline system is developed, we also add Linguistica features in three different sets:

- **Linguistica signature analysis (LXA-A):** Stem and suffix identified by Linguistica, after processing on our disease word list. If a token has no Linguistica analysis then these features are assigned a value of “Undefined”.

- **Linguistica morphemes, unigram only (LXA-B):** As described in Section 4.3.3, we include morpheme unigrams $m_0 \ldots m_n$. Morpheme bigrams are not included.

- **Linguistica morphemes, unigram and bigram (LXA-C):** Morpheme unigrams and bigrams $m_0m_1, \ldots, m_{n-1}m_n$ are used as features.

### 4.4 Experiments

**Disease tagging with AZDC corpus**

We compare feature sets by performing 10-fold cross-validation on the AZDC corpus. Folds are divided to ensure that a single abstract is not split across two folds. We first examine classifiers that do not use Linguistica features. Results for these baseline systems are shown in Table 4.8.

We first build a “Dictionary look-up” classifier that scans the corpus for terms appearing in SNOMED disease list and tags each match as a disease mention. For this approach supervised classification is not applied, and the only criteria is string matching. The low recall demonstrates why exact matching is not a highly reliable method of detecting disease mentions. Examining results, we find cases where lexical variations results in a near miss. For example, the disease term *Tapeto-Choroidal Dystrophy* occurs in the corpus but not in our disease term list; the associated SNOMED concept is *choroideremia*. We see that one out of three tagged disease mentions discovered by exact match are false positives. Examining results, we find that many are instances where only a substring of an annotated mention is tagged. For example *colorectal cancer* does not appear in our SNOMED disease list, and as a result the token *cancer* is tagged as a disease mention which refers to a more general concept than the actual disease mention.
<table>
<thead>
<tr>
<th>Features</th>
<th>Description</th>
<th>Context</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Token Feature Set</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Token</td>
<td>Unaltered Token</td>
<td>Unigram: ( t_{-2}, \ldots, t_2 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bigram: ( t_{-2}t_{-1}, \ldots, t_1t_2 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Part-of-speech</td>
<td>Unigram: ( t_{-2}, \ldots, t_2 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bigram: ( t_{-2}t_{-1}, \ldots, t_1t_2 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Trigram: ( t_{-1}t_0t_1 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Normalized Token</td>
<td>Unigram: ( t_{-2}, \ldots, t_2 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bigram: ( t_{-2}t_{-1}, \ldots, t_1t_2 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Trigram: ( t_{-2}t_{-1}t_0, \ldots, t_0t_1t_2 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Stemmed Token</td>
<td>Unigram: ( t_{-2}, \ldots, t_2 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bigram: ( t_{-2}t_{-1}, \ldots, t_1t_2 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Orthographic Feature Set</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>InitCaps</td>
<td>1st char. capitalized</td>
<td>( t_0 )</td>
</tr>
<tr>
<td>AllCaps</td>
<td>Has upper-case only</td>
<td>( t_0 )</td>
</tr>
<tr>
<td>MixCase</td>
<td>Has upper- and lower-case</td>
<td>( t_0 )</td>
</tr>
<tr>
<td>SingleLow</td>
<td>Is 1 lower-case char.</td>
<td>( t_0 )</td>
</tr>
<tr>
<td>SingleUp</td>
<td>Is 1 upper-case char.</td>
<td>( t_0 )</td>
</tr>
<tr>
<td>IsNumber</td>
<td>Is integer of f.p. number</td>
<td>( t_0 )</td>
</tr>
<tr>
<td>PunctuationChar</td>
<td>Is punctuation</td>
<td>( t_{-1}, t_0, t_1 )</td>
</tr>
<tr>
<td>LastCharPunct</td>
<td>Last char. is punctuation</td>
<td>( t_{-1} )</td>
</tr>
<tr>
<td>WordLength</td>
<td>Binned: 0, 1, 2, 3-5, 6+</td>
<td>( t_{0} )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>n-gram Feature Set</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Character Prefixes</td>
<td>( n=2,3,4 )</td>
<td>( t_0 )</td>
</tr>
<tr>
<td>Character Suffix</td>
<td>( n=2,3,4 )</td>
<td>( t_0 )</td>
</tr>
<tr>
<td>Word-internal char. n-grams</td>
<td>( n=3,4 )</td>
<td>( t_0 )</td>
</tr>
</tbody>
</table>

Table 4.7: Feature sets for CRF NER model. Third column shows the token contexts used for each features, relative to the current token. For example, \( t_{-1}t_0 \) for Part-of-Speech indicates that the current and previous POS tag are combined to a single feature.
Examining supervised classifiers in Table 4.8, we see progressive improvements as we include orthographic (Ortho.) and n-gram features. Each shows a statistically significant improvement from the preceding row, using a paired t-test between folds at the 5% significance level. Including dependency parse information does not show a significant gain in improvement. We next attempt to use information from the Dictionary look-up tagger as a feature for supervised classifiers. To do this, we use tag all mentions discovered by the SNOMED look-up classifier with a separate set of BIO tags, and include these tags as features in the CRF model. This results in a further improvement in accuracy of 1.5%.

In Table 4.8 we compare results with systems that use Linguistica features (LXA-A, LXA-B, and LXA-C). Here we compare the relative benefit of the n-gram and Linguistica feature sets, both of which use word-internal character information. Recall that feature set n-gram includes all possible prefixes and suffixes of 2-4 characters, as shown in Table 4.7, while Linguistica-based features includes stems and suffixes of any length that have been acquired by unsupervised morphological analysis. We see that gains in absolute F-measure are about 2% when n-gram features are not already included, but the overall gain does not surpass the n-gram feature set alone. Of the three Linguistica feature sets, only LXA-C shows a significant improvement over the classifier with feature Token + Ortho. + n-gram.

Linguistica-based feature sets have the advantage that they uses a disease word list as input, and so only words on this list can be assigned a Linguistica suffix. Words that do not appear on the disease word list are assigned an “Undefined” value. In contrast the n-gram feature set starts with no information about what tokens appear in the disease word list. Then the benefit of the Linguistica-based features may simply that they are binary indicators of a disease-related word. To test this we create an additional feature, IsDiseaseWord, which is a binary flag indicating that a token appeared in our disease word list. We see that IsDiseaseWord improves when added to a fairly full set (Token + Ortho. + n-gram) but inclusion of Linguistica features does show some further benefit.

Combined classification for disease and genes/protein mentions with CALBC corpus

Next we apply our experiments to the CALBC silver standard corpus. We first consider train a tagger for disease mentions only, as done in the previous section. Results are shown in Table 4.10.
<table>
<thead>
<tr>
<th>Feature Sets</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dictionary look-up (SNOMED)</td>
<td>65.38</td>
<td>37.00</td>
<td>47.26</td>
</tr>
<tr>
<td>Token</td>
<td>85.07</td>
<td>60.91</td>
<td>70.99</td>
</tr>
<tr>
<td>Token + Ortho.</td>
<td>84.71</td>
<td>62.34</td>
<td>71.83</td>
</tr>
<tr>
<td>Token + Ortho. + n-gram</td>
<td>81.92</td>
<td>70.81</td>
<td>75.96</td>
</tr>
<tr>
<td>Token + Ortho. + n-gram + Dep. Parse</td>
<td>82.00</td>
<td>70.88</td>
<td>76.03</td>
</tr>
<tr>
<td>Token + Ortho. + n-gram + SNOMED-Tags</td>
<td>82.80</td>
<td>72.81</td>
<td>77.49</td>
</tr>
</tbody>
</table>

Table 4.8: Results of 10-fold cross-validation on AZDC corpus.

<table>
<thead>
<tr>
<th>Feature Sets</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Token + Ortho. + LXA-A</td>
<td>84.31</td>
<td>65.66</td>
<td>73.82</td>
</tr>
<tr>
<td>Token + Ortho. + LXA-B</td>
<td>84.20</td>
<td>65.78</td>
<td>73.86</td>
</tr>
<tr>
<td>Token + Ortho. + LXA-C</td>
<td>84.26</td>
<td>66.09</td>
<td>74.08</td>
</tr>
<tr>
<td>Token + Ortho. + n-gram + LXA-A</td>
<td>81.85</td>
<td>71.75</td>
<td>76.47</td>
</tr>
<tr>
<td>Token + Ortho. + n-gram + LXA-B</td>
<td>81.84</td>
<td>71.53</td>
<td>76.34</td>
</tr>
<tr>
<td>Token + Ortho. + n-gram + LXA-C</td>
<td>82.21</td>
<td>71.75</td>
<td>76.62</td>
</tr>
<tr>
<td>Token + Ortho. + n-gram + IsDiseaseWord</td>
<td>81.95</td>
<td>71.22</td>
<td>76.21</td>
</tr>
<tr>
<td>Token + Ortho. + n-gram + LXA-C + SNOMED-Tags</td>
<td>82.43</td>
<td>72.41</td>
<td>77.09</td>
</tr>
</tbody>
</table>

Table 4.9: Results of 10-fold cross-validation on AZDC corpus, incorporating Linguistica feature sets.

<table>
<thead>
<tr>
<th>Feature Sets</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Token</td>
<td>83.73</td>
<td>60.24</td>
<td>70.07</td>
</tr>
<tr>
<td>Token + Ortho.</td>
<td>83.40</td>
<td>60.81</td>
<td>70.33</td>
</tr>
<tr>
<td>Token + Ortho. + LXA-A</td>
<td>83.15</td>
<td>66.64</td>
<td>73.98</td>
</tr>
<tr>
<td>Token + Ortho. + LXA-C</td>
<td>83.12</td>
<td>66.51</td>
<td>73.89</td>
</tr>
<tr>
<td>Token + Ortho. + n-gram</td>
<td>82.79</td>
<td>74.66</td>
<td>78.52</td>
</tr>
<tr>
<td>Token + Ortho. + n-gram + LXA-A</td>
<td>82.42</td>
<td>74.79</td>
<td>78.42</td>
</tr>
<tr>
<td>Token + Ortho. + n-gram + LXA-C</td>
<td>83.58</td>
<td>75.45</td>
<td>79.31</td>
</tr>
</tbody>
</table>

Table 4.10: Results on CALBC data set, for classification of diseases only.
We see again that Linguistica-based features do show some improvement in F-measure with the CALBC data set. However, individually they do not surpass the benefit on the n-gram set.

Table 4.11 shows results of experiments on multi-class named entity recognition on the CALBC data set. Our goal in this experiment is to compare two different approaches to two-class named entity classification.

- **Multi-class** classification: We train a single CRF model to recognize two different classes of named entities, genes and diseases. We extend our BIO tag set by creating separate tags for each class (i.e. B-DIS and I-DIS for diseases, B-PGN and I-PGN for genes/proteins).

- **Single-class** classification: We train two separate classifiers, one for disease mentions and another for gene mentions, and then apply each model to the same test set. With this system a single token may be tagged as belonging to both a disease and gene term if both models assign a B or I tag. However in practice we find that this only occurs once in our test set of 5,000 sentences.

Results are shown in Table 4.11. Examining results for gene disease mentions, we see that adding LXA features has no benefit, which is expected because words in gene names are often not drawn from standard English. Also, in these experiments Linguistica analysis focused on a disease lexicon. In a multi-class classifier, we see that overall F-measure drops also when LXA feature are added, which we attribute to overfitting of the model to training data. Results in this experiment demonstrate that a richer feature set with task-specific morphological features may not have an overall benefit in a tagging systems trained for multiple types of named entities.

### 4.5 Conclusion

In this chapter we present methods to apply CRFs to biomedical named entity recognition, specifically for disease mentions. Our experiments differ from previous works in that we use morphological features acquired from the Linguistica toolkit, in addition to more simple features such as fixed-length prefix and suffix strings and internal character n-grams. We find that structured morphological features acquired from Linguistica improves performance when other word-internal features are not present; however, that benefit is generally lost as the feature set becomes saturated with other word internal information. We also show how the segmentation of stems into sub-stems can expand our Linguistica-based feature sets beyond the initial signature analysis.
<table>
<thead>
<tr>
<th>Model</th>
<th>Feature set</th>
<th>Prec.</th>
<th>Recall</th>
<th>F-meas.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disease Mentions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single</td>
<td>Token + Ortho. + n-gram</td>
<td>82.79</td>
<td>74.66</td>
<td>78.52</td>
</tr>
<tr>
<td>Single</td>
<td>Token + Ortho. + n-gram + LXA-C</td>
<td>83.58</td>
<td>75.45</td>
<td>79.31</td>
</tr>
<tr>
<td>Multi</td>
<td>Token + Ortho. + n-gram</td>
<td>83.34</td>
<td>75.45</td>
<td>79.20</td>
</tr>
<tr>
<td>Mulit</td>
<td>Token + Ortho. + n-gram + LXA-C</td>
<td>83.08</td>
<td>75.36</td>
<td>79.03</td>
</tr>
<tr>
<td></td>
<td>Gene Mentions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single</td>
<td>Token + Ortho. + n-gram</td>
<td>80.29</td>
<td>58.64</td>
<td>67.78</td>
</tr>
<tr>
<td>Single</td>
<td>Token + Ortho. + n-gram + LXA-C</td>
<td>80.82</td>
<td>57.66</td>
<td>67.31</td>
</tr>
<tr>
<td>Multi</td>
<td>Token + Ortho. + n-gram</td>
<td>82.46</td>
<td>58.46</td>
<td>68.42</td>
</tr>
<tr>
<td>Mulit</td>
<td>Token + Ortho. + n-gram + LXA-C</td>
<td>81.14</td>
<td>58.11</td>
<td>67.72</td>
</tr>
<tr>
<td></td>
<td>Combined Results</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single</td>
<td>Token + Ortho. + n-gram</td>
<td>81.68</td>
<td>66.70</td>
<td>73.44</td>
</tr>
<tr>
<td>Single</td>
<td>Token + Ortho. + n-gram + LXA-C</td>
<td>82.37</td>
<td>66.62</td>
<td>73.66</td>
</tr>
<tr>
<td>Multi</td>
<td>Token + Ortho. + n-gram</td>
<td>82.96</td>
<td>67.01</td>
<td>74.14</td>
</tr>
<tr>
<td>Mulit</td>
<td>Token + Ortho. + n-gram + LXA-C</td>
<td>82.23</td>
<td>66.79</td>
<td>73.71</td>
</tr>
</tbody>
</table>

Table 4.11: Results on CALBC data set, for classification of diseases and genes/proteins. The first column indicates whether one multi-class NER model was trained (Multi), or if two separate single-class models were trained (Single).
CHAPTER 5
CONCLUSION

In this thesis we present works that study language processing of named entities, and specifically present methods to improve the processing of out-of-vocabulary words. In Chapter 2, we consider the problem of recognizing the language of origin of a name, which has applications in text-to-speech synthesis, machine transliteration, and data collection for demographics and life sciences. We specifically consider the case where the set of possible origin languages is not known in advance and word origin training data is limited or unavailable. We apply Cluster Adaptive Sampling (CAS), an active learning method, to guide collection of a labeled data set for subsequent supervised classification. The question we ask is whether a data set that is labeled with CAS will result in a more accurate supervised classifier than one trained on data selected randomly. We find that CAS improves supervised classification when language classes are easily distinguished from each other, which is the case in our first data set of geographic place names. For a second data set of personal US surnames, however, we find that higher word origin ambiguity limits the benefit of CAS. We attribute this ambiguity to the presence of similar language pairs (e.g. Italian and Spanish) and patterns of Americanization that can obfuscate word origin. One contribution of this work is that we consider the word origin problem in a data set where variation occurs naturally and is not introduced artificially by combining names from language-specific lists. Our methods differ from previous works in that we use active learning to improve subsequent supervised classification. Additionally, we present a sequential Maximum Entropy model that improves on a non-sequential Maximum Entropy model trained on the same features.

In Chapter 3, we apply word origin recognition to grapheme-to-phoneme conversion, which has applications in text-to-speech synthesis and automatic speech recognition. As in Chapter 2 our goal is to consider scenarios where we do not have prior knowledge of the factor languages in a data set, which may be the case for many NLP tasks. Our baseline is a G2P system trained on a single data set without knowledge of word origin. We compare this baseline to G2P systems where output of multiple language-specific models are combined; training data sets for language-specific models are obtained by labeling the initial training set with a word origin model. We consider different
methods of word origin modeling as well as different ways to combine results with smoothing. We find that a G2P system using only language-specific G2P models performs worse than a single language-independent baseline due to sparsity introduced by dividing the training set. However we see an improvement in accuracy when we include output of a language-independent model trained on all data. We also present a system that uses an unsupervised word origin model where language labels are assigned via clustering of words by character n-grams. This system requires no word origin annotation but is able to improve on the language-independent G2P baseline. Experiments use the same set of US surnames as in Chapter 2, and we note that word origin information is shown to improve G2P conversion despite the fact that word origin modeling is imperfect for this data set. The methods we present can be used with any G2P approach that outputs posterior probabilities of the phoneme sequence given the grapheme sequence.

In Chapter 4 we study another task where OOV words are problematic, that is, named entity recognition of disease terms in biomedical text. We show how output of the Linguistica toolkit for unsupervised learning of morphology can be applied to this task. We find that disease terms, which are frequently composed of compound stems, can be decomposed into component stems with minimal additional processing. We find that the benefit of structured morphology information is lessened as the feature set becomes saturated with other word-internal features; however, experiments show that morphology information may be applicable for other types of named entities in the biomedical domain. Our experiments differ from previous works in that we use morphological features acquired from the Linguistica toolkit, in addition to more simple features such as fixed-length prefix and suffix strings and internal character n-grams.
REFERENCES


89


calbc silver standard corpus - harmonizing multiple semantic annotations in a large biomedical corpus. CALBC.


[Srinivasan and Wedemeyer 2003] Padmini Srinivasan and Micah Wedemeyer. 2003. Mining concept profiles with the vector model or where on earth are diseases being studied? Text Mining Workshop (SIAM International Conference on Data Mining 2003).


