The CMU Submission for the Shared Task on Language Identification in Code-Switched Data

Chu-Cheng Lin **Waleed Ammar** **Lori Levin** **Chris Dyer**
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA, 15213, USA
{chuchenl,wammar,lsl,cdyer}@cs.cmu.edu

Abstract

We describe the CMU submission for the 2014 shared task on language identification in code-switched data. We participated in all four language pairs: Spanish–English, Mandarin–English, Nepali–English, and Modern Standard Arabic–Arabic dialects. After describing our CRF-based baseline system, we discuss three extensions for learning from unlabeled data: semi-supervised learning, word embeddings, and word lists.

1 Introduction

Code switching (CS) occurs when a multilingual speaker uses more than one language in the same conversation or discourse. Automatic idenefication of the points at which code switching occurs is important for two reasons: (1) to help sociolinguists analyze the frequency, circumstances and motivations related to code switching (Gumperz, 1982), and (2) to automatically determine which language-specific NLP models to use for analyzing segments of text or speech.

CS is pervasive in social media due to its informal nature (Lui and Baldwin, 2014). The first workshop on computational approaches to code switching in EMNLP 2014 organized a shared task (Solorio et al., 2014) on identifying code switching, providing training data of multilingual tweets with token-level language-ID annotations. See §2 for a detailed description of the shared task. This short paper documents our submission in the shared task.

We note that constructing a CS data set that is annotated at the token level requires remarkable manual effort. However, collecting raw tweets is easy and fast. We propose leveraging both labeled and unlabeled data in a unified framework; *conditional random field autoencoders* (Ammar et al., 2014). The CRF autoencoder framework consists of an encoding model and a reconstruction model. The encoding model is a linear-chain conditional random field (CRF) (Lafferty et al., 2001) which generates a sequence of labels, conditional on a token sequence. Importantly, the parameters of the encoding model can be interpreted in the same way a CRF model would. This is in contrary to generative model parameters which explain both the observation sequence and the label sequence. The reconstruction model, on the other hand, independently generates the tokens conditional on the corresponding labels. Both labeled and unlabeled data can be efficiently used to fit parameters of this model, minimizing regularized log loss. See §4.1 for more details.

After modeling unlabeled token sequences, we explore two other ways of leveraging unlabeled data: *word embeddings* and *word lists*. The word embeddings we use capture monolingual distributional similarities and therefore may be indicative of a language (see §4.2). A word list, on the other hand, is a collection of words which have been manually or automatically constructed and share some property (see §4.3). For example, we extract the set of surface forms in monolingual corpora.

In §5, we describe the experiments and discuss results. According to the results, modeling unlabeled data using CRF autoencoders did not improve prediction accuracy. Nevertheless, more experiments need to be run before we can conclude this setting. On the positive side, word embeddings and word lists have been shown to improve CS prediction accuracy, provided they have decent coverage of tokens in the test set.

2 Task Description

The shared task training data consists of code–switched tweets with token-level annotations. The data is organized in four language pairs: English–Spanish (En-Es), English–Nepali (En-

Proceedings of The First Workshop on Computational Approaches to Code Switching, pages 80–86,
October 25, 2014, Doha, Qatar.

Ne), Mandarin–English (Zh-En) and Modern Standard Arabic–Arabic dialects (MSA-ARZ). Table 1 shows the size of the data sets provided for the shared task in each language pair.

For each tweet in the data set, the user ID, tweet ID, and a list of tokens' start offset and end offset are provided. Each token is annotated with one of the following labels: `lang1`, `lang2`, `ne` (i.e., named entities), `mixed` (i.e., mixed parts of lang1 and lang2), `ambiguous` (i.e., cannot be identified given context), and `other`.

Two test sets were used to evaluate each submission for the shared task in each language pair. The first test set consists of Tweets, similar to the training set. The second test set consists of token sequences from a surprise genre. Since participants were not given the test sets, we only report results on a Twitter test set (a subset of the data provided for shared task participants). Statistics of our train/test data splits are given in Table 5.

lang. pair	split	tweets	tokens	users
En–Ne	all	9,993	146,053	18
	train	7,504	109,040	12
	test	2,489	37,013	6
En–Es	all	11,400	140,738	9
	train	7,399	101,451	6
	test	4,001	39,287	3
Zh–En	all	994	17,408	995
	train	662	11,677	663
	test	332	5,731	332
MSA–ARZ	all	5,862	119,775	7
	train	4,800	95,352	6
	test	1,062	24,423	1

Table 1: Total number of tweets, tokens, and Twitter user IDs for each language pair. For each language pair, the first line represents all data provided to shared task participants. The second and third lines represent our train/test data split for the experiments reported in this paper. Since Twitter users are allowed to delete their tweets, the number of tweets and tokens reported in the third and fourth columns may be less than the number of tweets and tokens originally annotated by the shared task organizers.

3 Baseline System

We model token-level language ID as a sequence of labels using a linear-chain conditional random field (CRF) (Lafferty et al., 2001) described in §3.1 with the features in §3.2.

3.1 Model

A linear-chain CRF models the conditional probability of a label sequence $\mathbf{y}$ given a token sequence $\mathbf{x}$ and given extra context $\boldsymbol{\phi}$, as follows:

$$p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\phi}) = \frac{\exp \boldsymbol{\lambda}^\top \sum_{i=1}^{|\mathbf{x}|} \mathbf{f}(\mathbf{x}, y_i, y_{i-1}, \boldsymbol{\phi})}{\sum_{\mathbf{y}'} \exp \boldsymbol{\lambda}^\top \sum_{i=1}^{|\mathbf{x}|} \mathbf{f}(\mathbf{x}, y'_i, y'_{i-1}, \boldsymbol{\phi})}$$

where $\boldsymbol{\lambda}$ is a vector of feature weights, and $\mathbf{f}$ is a vector of local feature functions. We use $\boldsymbol{\phi}$ to explicitly represent context information necessary to compute the feature functions described below.

In a linear-chain structure, y_i only depends on observed variables $\mathbf{x}$, $\boldsymbol{\phi}$ and the neighboring labels y_{i-1} and y_{i+1}. Therefore, we can use dynamic programming to do inference in run time that is quadratic in the number of unique labels and linear in the sequence length. We use L-BFGS to learn the feature weights $\boldsymbol{\lambda}$, maximizing the L_2-regularized log-likelihood of labeled examples $\mathcal{L}$:

$$\ell\ell_{\text{supervised}}(\boldsymbol{\lambda}) = c_{L_2} ||\boldsymbol{\lambda}||_2^2 + \sum_{\langle \mathbf{x}, \mathbf{y} \rangle \in \mathcal{L}} \log p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\phi})$$

After training the model, we use again use dynamic programming to find the most likely label sequence, for each token sequence in the test set.

3.2 Features

We use the following features in the baseline system:

- character n-grams (loweredcased tri- and quadgrams)
- prefixes and suffixes of lengths 1, 2, 3 and 4
- unicode page of the first character[1]
- case (first-character-uppercased vs. all-characters-uppercased vs. all-characters-alphanumeric)
- tweet-level language ID predictions from two off-the-shelf language identifiers: `cld2`[2] and `ldig`[3]

[1] `http://www.unicode.org/charts/`
[2] `https://code.google.com/p/cld2/`
[3] `https://github.com/shuyo/ldig`

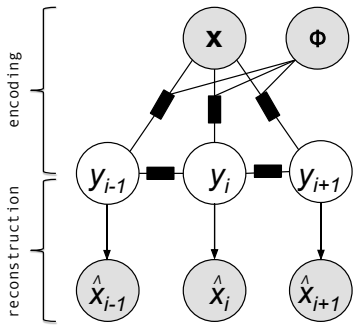

Figure 1: A diagram of the CRF autoencoder

4 Using Unlabeled Data

In §3, we learn the parameters of the CRF model parameters in a standard fully supervised fashion, using labeled examples in the training set. Here, we attempt to use unlabeled examples to improve our system's performance in three ways: modeling unlabeled token sequences in the CRF autoencoder framework, word embeddings, and word lists.

4.1 CRF Autoencoders

A CRF autoencoder (Ammar et al., 2014) consists of an input layer, an output layer, and a hidden layer. Both input and output layer represent the observed token sequence. The hidden layer represents the label sequence. Fig. 1 illustrates the model dependencies for sequence labeling problems with a first-order Markov assumption. Conditional on an observation sequence $\mathbf{x}$ and side information $\boldsymbol{\phi}$, a traditional linear-chain CRF model is used to generate the label sequence $\mathbf{y}$. The model then generates $\hat{\mathbf{x}}$ which represents a reconstruction of the original observation sequence. Elements of this reconstruction (i.e., $\hat{x}_i$) are then independently generated conditional on the corresponding label y_i using simple categorical distributions.

The parametric form of the model is given by:

$$p(\mathbf{y}, \hat{\mathbf{x}} \mid \mathbf{x}, \boldsymbol{\phi}) = \prod_{i=1}^{|\mathbf{x}|} \theta_{\hat{x}_i|y_i} \times \frac{\exp \boldsymbol{\lambda}^\top \sum_{i=1}^{|\mathbf{x}|} \mathbf{f}(\mathbf{x}, y_{i-1}, y_i, i, \boldsymbol{\phi})}{\sum_{\mathbf{y}'} \exp \boldsymbol{\lambda}^\top \sum_{i=1}^{|\mathbf{x}|} \mathbf{f}(\mathbf{x}, y'_{i-1}, y'_i, i, \boldsymbol{\phi})}$$

where $\boldsymbol{\lambda}$ is a vector of CRF feature weights, $\mathbf{f}$ is a vector of local feature functions (we use the same features described in §3.2), and $\theta_{\hat{x}_i|y_i}$ are categorical distribution parameters of the reconstruction model representing $p(\hat{x}_i \mid y_i)$.

We can think of a label sequence as a low-cardinality lossy compression of the corresponding token sequence. CRF autoencoders explicitly model this intuition by creating an information bottleneck where label sequences are required to regenerate the same token sequence despite their limited capacity. Therefore, when only unlabeled examples $\mathcal{U}$ are available, we train CRF autoencoders by maximizing the regularized likelihood of generating reconstructions $\hat{\mathbf{x}}$, conditional on $\mathbf{x}$, marginalizing values of label sequences $\mathbf{y}$:

$$\ell\ell_{\text{unsupervised}}(\boldsymbol{\lambda}, \boldsymbol{\theta}) = c_{L_2}||\boldsymbol{\lambda}||_2^2 + R_{\text{Dirichlet}}(\boldsymbol{\theta}, \alpha) + \sum_{\langle \mathbf{x}, \hat{\mathbf{x}} \rangle \in \mathcal{U}} \log \sum_{\mathbf{y}:|\mathbf{y}|=|\mathbf{x}|} p(\boldsymbol{y}, \hat{\boldsymbol{x}} \mid \boldsymbol{x})$$

where $R_{\text{Dirichlet}}$ is a regularizer based on a variational approximation of a symmetric Dirichlet prior with concentration parameter α for the reconstruction parameters $\boldsymbol{\theta}$.

Having access to labeled examples, it is easy to modify this objective to learn from both labeled and unlabeled examples as follows:

$$\ell\ell_{\text{semi}}(\boldsymbol{\lambda}, \boldsymbol{\theta}) = c_{L_2}||\boldsymbol{\lambda}||_2^2 + R_{\text{Dirichlet}}(\boldsymbol{\theta}, \alpha) + c_{\text{unlabeled}} \times \sum_{\langle \mathbf{x}, \hat{\mathbf{x}} \rangle \in \mathcal{U}} \log \sum_{\mathbf{y}:|\mathbf{y}|=|\mathbf{x}|} p(\boldsymbol{y}, \hat{\boldsymbol{x}} \mid \boldsymbol{x}) + c_{\text{labeled}} \times \sum_{\langle \mathbf{x}, \mathbf{y} \rangle \in \mathcal{L}} \log p(\boldsymbol{y} \mid \boldsymbol{x})$$

We use block coordinate descent to optimize this objective. First, we use c_{em} iterations of the expectation maximization algorithm to optimize the $\boldsymbol{\theta}$-block while the $\boldsymbol{\lambda}$-block is fixed, then we optimize the $\boldsymbol{\lambda}$-block with c_{lbfgs} iterations of L-BFGS (Liu et al., 1989) while the $\boldsymbol{\theta}$-block is fixed.[4]

4.2 Unsupervised Word Embeddings

For many NLP tasks, using unsupervised word representations as features improves accuracy (Turian et al., 2010). We use `word2vec` (Mikolov et al., 2013) to train 100–dimensional word embeddings from a large Twitter corpus of about 20 million tweets extracted from the live stream, in multiple languages. We define an additional feature function

[4] An open source efficient c++ implementation of our method can be found at `https://github.com/ldmt-muri/alignment-with-openfst`

in the CRF autoencoder model §4.1 for each of the 100 dimensions, conjoined with the label y_i. The feature value is the corresponding dimension for x_i. A binary feature indicating the absence of word embeddings is fired for out-of-vocabulary words (i.e., words for which we do not have word embeddings). The token-level coverage of the word embeddings for each of the languages or dialects used in the training data is reported in Table 2.

4.3 Word List Features

While some words are ambiguous, many words frequently occur in only one of the two languages being considered. An easy way to identify the label of such unambiguous words is to check whether they belong to the vocabulary of either language. Moreover, named entity recognizers typically rely on gazetteers of named entities to improve their performance. We generalize the notion of using monolingual vocabularies and gazetteers of named entities to general word lists. Using K word lists $\{l_1, \ldots, l_K\}$, when a token x_i is labeled with y_i, we fire a binary feature that conjoins $\langle y_i, \delta(x_i \in l_1), \ldots, \delta(x_i \in l_K)\rangle$, where δ is an indicator boolean function. We use the following word lists:

- Hindi and Nepali Wikipedia article titles
- multilingual named entities from the JRC dataset[5] and CoNLL 2003 shared task
- word types in monolingual corpora in MSA, ARZ, En and Es.
- set difference between the following pairs of word lists: MSA-ARZ, ARZ-MSA, En-Es, Es-En.

Transliteration from Devanagari The Nepali–English tweets in the dataset are romanized. This renders our Nepali word lists, which are based on the Devanagari script, useless. Therefore, we transliterate the Hindi and Nepali named entities lists using a deterministic phonetic mapping. We romanize the Devanagari words using the IAST scheme.[6] We then drop all accent marks on the characters to make them fit into the 7–bit ASCII range.

[5] http://datahub.io/dataset/jrc-names

[6] http://en.wikipedia.org/wiki/International_Alphabet_of_Sanskrit_Transliteration

language	embeddings coverage	word lists coverage
ARZ	30.7%	68.8%
En	73.5%	55.7%
MSA	26.6%	76.8%
Ne	14.5%	77.0%
Es	62.9%	78.0%
Zh	16.0%	0.7%

Table 2: The type-level coverage of annotated data according to word embeddings (second column) and according to word lists (third column), per language.

5 Experiments

We compare the performance of five models for each language pair, which correspond to the five lines in Table 3. The first model, "CRF" is the baseline model described in §3. The second "CRF + $\mathcal{U}_{\text{test}}$" and the third "CRF + $\mathcal{U}_{\text{all}}$" are CRF autoencoder models (see §4.1) with two sets of unlabeled data: (1) $\mathcal{U}_{\text{test}}$ which only includes the test set,[7] and (2) $\mathcal{U}_{\text{all}}$ which includes the test set as well as *all* tweets by the set of users who contributed any tweets in $\mathcal{L}$. The fourth model "CRF + $\mathcal{U}_{\text{all}}$ + emb." is a CRF autoencoder which uses word embedding features (see §4.2), as well as the features described in §3.2. Finally, the fifth model "CRF + $\mathcal{U}_{\text{all}}$ + emb. + lists" further adds word list features (see §4.3). In all but the "CRF" model, we adopt a transductive learning setup.

Since the CRF baseline is used as the encoding part of the CRF autoencoder model, we use the supervisedly-trained CRF parameters to initialize the CRF autoencoder models. The categorical distributions of the reconstruction model are initialized with discrete uniforms. We set the weight of the labeled data log-likelihood $c_{\text{labeled}} = 0.5$, the weight of the unlabeled data log-likelihood $c_{\text{unlabeled}} = 0.5$, the L_2 regularization strength $c_{L_2} = 0.3$, the concentration parameter of the Dirichlet prior $\alpha = 0.1$, the number of L-BFGS iterations $c_{\text{LBFGS}} = 4$, and the number of EM iterations $c_{EM} = 4$.[8] We stop training after 50 iterations of block coordinate descent.

[7] $\mathcal{U}_{\text{test}}$ is potentially useful when the test set belongs to a different domain than the labeled examples, which is often referred to as "domain adaptation". However we were unable to test this hypothesis since all the CS annotations we had access to are from Twitter.

[8] Hyper-parameters c_{L_2} and α were tuned using cross-validation. The remaining hyper-parameters were not tuned.

config	En–Ne	MSA–ARZ	En–Es	Zh–En
CRF	95.2%	80.5%	94.6%	94.9%
+$\mathcal{T}_{\text{test}}$	95.2%	80.6%	94.6%	94.9%
+$\mathcal{T}_{\text{all}}$	95.2%	80.7%	94.6%	94.9%
+emb.	95.3%	81.3%	95.1%	95.0%
+lists	97.0%	81.2%	96.7%	95.3%

Table 3: Token level accuracy results for each of the four language pairs.

label	predicted MSA	predicted ARZ
true MSA	93.9%	5.3%
true ARZ	32.1%	65.2%

Table 4: Confusion between MSA and ARZ in the Baseline configuration.

Results. The CRF baseline results are reported in the first line in Table 3. For three language pairs, the overall token-level accuracy ranges between 94.6% and 95.2%. In the fourth language pair, MSA-ARZ, the baseline accuracy is 80.5% which indicates the relative difficulty of this task.

The second and third lines in Table 3 show the results when we use CRF autoencoders with the unlabeled test set ($\mathcal{U}_{\text{test}}$), and with all unlabeled tweets ($\mathcal{U}_{\text{all}}$), respectively. While semi-supervised learning did not hurt accuracy on any of the languages, it only resulted in a tiny increase in accuracy for the Arabic dialects task.

The fourth line in Table 3 extends the CRF autoencoder model (third line) by adding unsupervised word embedding features. This results in an improvement of 0.6% for MSA-ARZ, 0.5% for En-Es, 0.1% for En-Ne and Zh-En.

The fifth line builds on the fourth line by adding word list features. This results in an improvement of 1.7% in En-Ne, 1.6% in En-Es, 0.4% in Zh-En, and degradation of 0.1% in MSA-ARZ.

Analysis and Discussion The baseline performance in the MSA-ARZ task is considerably lower than those of the other tasks. Table 4 illustrates how the baseline model confuses lang1 and lang2 in the MSA-ARZ task. While the baseline system correctly labels 93.9% of MSA tokens, it only correctly labels 65.2% of ARZ tokens.

Although the reported semi-supervised results did not significantly improve on the CRF baseline, more work needs to be done in order to conclude these results:

lang. pair	$\lvert\mathcal{U}_{\text{test}}\rvert$	$\lvert\mathcal{U}_{\text{all}}\rvert$	$\lvert\mathcal{L}\rvert$
En–Ne	2489	6230	7504
MSA–ARZ	1062	2520	4800
Zh–En	332	332	663
En–Es	4001	7177	7399

Table 5: Number of tweets in $\mathcal{L}$, $\mathcal{U}_{\text{test}}$ and $\mathcal{U}_{\text{all}}$ used for semi-supervised learning of CRF autoencoders models.

- Use an out-of-domain test set where some adaptation to the test set is more promising.
- Vary the number of labeled examples $|\mathcal{L}|$ and the number of unlabeled examples $|\mathcal{U}|$. Table 5 gives the number of labeled and unlabeled examples used for training the model. It is possible that semi-supervised learning would have been more useful with a smaller $|\mathcal{L}|$ and a larger $|\mathcal{U}|$.
- Tune c_{labeled} and $c_{\text{unlabeled}}$.
- Split the parameters $\boldsymbol{\lambda}$ into two subsets: $\boldsymbol{\lambda}_{\text{labeled}}$ and $\boldsymbol{\lambda}_{\text{unlabeled}}$; where $\boldsymbol{\lambda}_{\text{labeled}}$ are the parameters which have a non-zero value for any input $\mathbf{x}$ in $\mathcal{L}$ and $\boldsymbol{\lambda}_{\text{unlabeled}}$ are the remaining parameters in $\boldsymbol{\lambda}$ which only have non-zero values with unlabeled examples but not with the labeled examples.
- Use a richer reconstruction model.
- Reconstruct a transformation of the token sequences instead of their surface forms.
- Train a token-level language ID model trained on a large number of languages, as opposed to disambiguating only two languages at a time.

Word embeddings improve the results for all language pairs, but the largest improvement is in MSA-ARZ and En-Es. Looking into the word embeddings coverage of those languages (i.e., MSA, ARZ, Es, En in Table 2), we find that they are better covered than the other languages (Ne, Zh). We conclude that further improvements on En-Ne and Zh-En may be expected if they are better represented in the corpus used to learn word embeddings.

As for the word lists, the largest improvement we get is the romanized word lists of Nepali, which have a 77.0% coverage and improve the accuracy by 1.7%. This shows that our transliterated word lists not only cover a lot of tokens, and are also useful for language ID. The Spanish

Config	lang1	lang2	ne
+lists	84.1%	76.5%	73.7%
-lists	84.2%	77.1%	71.5%

Table 6: F–Measures of two Arabic configurations. lang1 is MSA. lang2 is ARZ.

word lists also have a wide coverage, improving the overall accuracy by 1.6%. The overall accuracy of the Arabic dialects slightly degrades with the addition of the word lists. Closer inspection in table 6 reveals that it improves the F–Measure of the named entities at the expense of both MSA (lang1) and ARZ (lang2).

6 Related Work

Previous work on identifying languages in a multilingual document includes (Singh and Gorla, 2007; King and Abney, 2013; Lui et al., 2014). Their goal is generally more about identifying the languages that appear in the document than intra–sentential CS points.

Previous work on computational models of code–switching include formalism (Joshi, 1982) and language models that encode syntactic constraints from theories of code–switching, such as (Li and Fung, 2013; Li and Fung, 2014). These require the existence of a parser for the languages under consideration. Other work on prediction of code–switching points, such as (Elfardy et al., 2013; Nguyen and Dogruoz, 2013) and ours, do not depend upon such NLP infrastructure. Both of the aforementioned use basic character–level features and dictionaries on sequence models.

7 Conclusion

We have shown that a simple CRF baseline with a handful of feature templates obtains strong results for this task. We discussed three methods to improve over the supervised baseline using unlabeled data: (1) modeling unlabeleld data using CRF autoencoders, (2) using pre-trained word embeddings, and (3) using word list features.

We show that adding word embedding features and word lists features is useful when they have good coverage of words in a data set. While modest improvements are observed due to modeling unlabeled data with CRF autoenocders, we identified possible directions to gain further improvements.

While bilingual disambiguation was a good first step for identifying code switching, we suggest a reformulation of the task such that each label can take on one of many languages.

Acknowledgments

We thank Brendan O'Connor who helped assemble the Twitter dataset. We also thank the workshop organizers for their hard work, and the reviewers for their comments. This work was sponsored by the U.S. Army Research Laboratory and the U.S. Army Research Office under contract/grant number W911NF-10-1-0533. The statements made herein are solely the responsibility of the authors.

References

Waleed Ammar, Chris Dyer, and Noah A. Smith. 2014. Conditional random field autoencoders for unsupervised structured prediction. In *Proc. of NIPS*.

Heba Elfardy, Mohamed Al-Badrashiny, and Mona Diab. 2013. Code switch point detection in arabic. In *Natural Language Processing and Information Systems*, pages 412–416. Springer.

John J. Gumperz. 1982. *Discourse Strategies*. Studies in Interactional Sociolinguistics. Cambridge University Press.

Aravind K. Joshi. 1982. Processing of sentences with intra-sentential code-switching. In *Proceedings of the 9th Conference on Computational Linguistics - Volume 1*, COLING '82, pages 145–150, Czechoslovakia. Academia Praha.

Ben King and Steven Abney. 2013. Labeling the languages of words in mixed-language documents using weakly supervised methods. Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1110–1119. Association for Computational Linguistics.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*.

Ying Li and Pascale Fung. 2013. Improved mixed language speech recognition using asymmetric acoustic model and language model with code-switch inversion constraints. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 7368–7372, May.

Ying Li and Pascale Fung. 2014. Code switch language modeling with functional head constraint. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 4913–4917, May.

D. C. Liu, J. Nocedal, and C. Dong. 1989. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*.

Marco Lui and Timothy Baldwin. 2014. Accurate language identification of twitter messages. In *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM)*, pages 17–25, Gothenburg, Sweden, April. Association for Computational Linguistics.

Marco Lui, Han Jey Lau, and Timothy Baldwin. 2014. Automatic detection and language identification of multilingual documents. *Transactions of the Association of Computational Linguistics*, 2:27–40.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proc. of ICLR*.

Dong Nguyen and Seza A. Dogruoz. 2013. Word level language identification in online multilingual communication. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 857–862. Association for Computational Linguistics.

Anil Kumar Singh and Jagadeesh Gorla. 2007. Identification of languages and encodings in a multilingual document. In *Building and Exploring Web Corpora (WAC3-2007): Proceedings of the 3rd Web as Corpus Workshop, Incorporating Cleaneval*, volume 4, page 95.

Thamar Solorio, Elizabeth Blair, Suraj Maharjan, Steve Bethard, Mona Diab, Mahmoud Gonheim, Abdelati Hawwari, Fahad AlGhamdi, Julia Hirshberg, Alison Chang, and Pascale Fung. 2014. Overview for the first shared task on language identification in code-switched data. In *Proceedings of the First Workshop on Computational Approaches to Code-Switching*.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394, Stroudsburg, PA, USA. Association for Computational Linguistics.