AIDA: Identifying Code Switching in Informal Arabic Text

Heba Elfardy
Department of Computer Science
Columbia University
New York, NY
heba@cs.columbia.edu

Mohamed Al-Badrashiny, Mona Diab
Department of Computer Science
The George Washington University
Washington, DC
{badrashiny, mtdiab}@gwu.edu

Abstract
In this paper, we present the latest version of our system for identifying linguistic code switching in Arabic text. The system relies on Language Models and a tool for morphological analysis and disambiguation for Arabic to identify the class of each word in a given sentence. We evaluate the performance of our system on the test datasets of the shared task at the EMNLP workshop on Computational Approaches to Code Switching (Solorio et al., 2014). The system yields an average token-level \( F_{\beta=1} \) score of 93.6\%, 77.7\% and 80.1\%, on the first, second, and surprise-genre test-sets, respectively, and a tweet-level \( F_{\beta=1} \) score of 4.4\%, 36\% and 27.7\%, on the same test-sets.

1 Introduction
Most languages exist in some standard form while also being associated with informal regional varieties. Some languages exist in a state of diglossia (Ferguson, 1959). Arabic is one of those languages comprising a standard form known as Modern Standard Arabic (MSA), that is used in education, formal settings, and official scripts; and dialectal variants (DA) corresponding to the native tongue of Arabic speakers. While these variants have no standard orthography, they are commonly used and have become pervasive across web-forums, blogs, social networks, TV shows, and normal daily conversations. Arabic dialects may be divided into five main groups: Egyptian (including Libyan and Sudanese), Levantine (including Lebanese, Syrian, Palestinian and Jordanian), Gulf, Iraqi and Moroccan. Sub-dialectal variants also exist within each dialect (Habash, 2010). Speakers of a specific Arabic Dialect typically code switch between their dialect and MSA, and less frequently between different dialects, both inter and intra-sententially. The identification and classification of these dialects in diglossic text can enhance semantic predictability.

In this paper we modify an existing system AIDA (Elfardy and Diab, 2012b), (Elfardy et al., 2013) that identifies code switching between MSA and Egyptian DA (EDA). We apply the modified system to the datasets used for evaluating systems participating at the EMNLP Workshop on Computational Approaches to Linguistic Code Switching.1

2 Related Work
Dialect Identification in Arabic is crucial for almost all NLP tasks, and has recently gained interest among Arabic NLP researchers. One of the early works is that of (Biadsy et al., 2009) where the authors present a system that identifies dialectal words in speech through acoustic signals. Zaidan and Callison-Burch (2011) crawled a large dataset of MSA-DA news commentaries and annotated part of the dataset for sentence-level dialectalness employing Amazon Mechanical Turk. Cotterell and Callison-Burch (2014) extended the previous work by handling more dialects. In (Cotterell et al., 2014), the same authors collect and annotate on Amazon Mechanical Turk a large set of tweets and user commentaries pertaining to five Arabic dialects. Bouamor et al. (2014) select a set of 2,000 Egyptian Arabic sentences and have them translated into four other Arabic dialects to present the first multidialectal Arabic parallel corpus.

Eskander et al. (2014) present a system for handling Arabic written in Roman script “Arabizi”. Using decision trees; the system identifies whether each word in the given text is a foreign word or not and further divides non foreign words into four

1Another group in our lab was responsible for the organization of the task, hence we did not officially participate in the task.
classes: Arabic, Named Entity, punctuation, and sound.

In the context of machine-translation, Salloum and Habash (2011) tackle the problem of DA to English Machine Translation (MT) by pivoting through MSA. The authors present a system that uses DA to MSA transfer rules before applying state of the art MSA to English MT system to produce an English translation. In (Elfardy and Diab, 2012a), we present a set of guidelines for token-level identification of DA while in (Elfardy and Diab, 2012b), (Elfardy et al., 2013) we tackle the problem of token-level dialect-identification by casting it as a code-switching problem. Elfardy and Diab (2013) presents our solution for the sentence-level dialect identification problem.

3 Shared Task Description

The shared task for “Language Identification in Code-Switched Data” (Solorio et al., 2014) aims at allowing participants to perform word-level language identification in code-switched Spanish-English, MSA-DA, Chinese-English and Nepalese-English data. In this work, we only focus on MSA-DA data. The dataset has six tags:

1. **lang1**: corresponds to an MSA word, ex. AlrAhn \(^2\) meaning “the current”;
2. **lang2**: corresponds to a DA word, ex. ezyk meaning “how are you”;
3. **mixed**: corresponds to a word with mixed morphology, ex. المألوشون ألم>lw$wn meaning “the ones that were excluded or rejected”;
4. **other**: corresponds to punctuation, numbers and words having punctuation or numbers attached to them;
5. **ambig**: corresponds to a word where the class cannot be determined given the current context, could either be lang1 or lang2; ex. the phrase كله تمام klh tmAm meaning “all is well” is ambiguous if enough context is not present since it can be used in both MSA and EDA.
6. **NE**: corresponds to a named-entity, ex. mSr meaning “Egypt”.

\(^2\)We use Buckwalter transliteration scheme http://www.qamus.org/transliteration.htm

4 Approach

We use a variant of the system that was presented in (Elfardy et al., 2013) to identify the tag of each word in a given Arabic sentence. The original approach relies on language models and a morphological analyzer to assign tags to words in an input sentence. In this new variant, we use MADAMIRA (Pasha et al., 2014); a tool for morphological analysis and disambiguation for Arabic. The advantage of using MADAMIRA over using a morphological analyzer is that MADAMIRA performs contextual disambiguation of the analyses produced by the morphological analyzer, hence reducing the possible options for analyses per word. Figures 1 illustrates the pipeline of the proposed system.

4.1 Preprocessing

We experiment with two preprocessing techniques:

1. **Basic**: In this scheme, we only perform a basic clean-up of the text by separating punctuation and numbers from words, normalizing word-lengthening effects, and replacing all punctuation, URLs, numbers and non-Arabic words with PUNC, URL, NUM, and LAT keywords, respectively

2. **Tokenized**: In this scheme, in addition to basic preprocessing, we use MADAMIRA toolkit to tokenize clitics and affixes by applying the D3-tokenization scheme (Habash and Sadat, 2006). For example, the word بجد which means “with seriousness” becomes “ب+ جد” after tokenization.

4.2 Language Model

The ‘Language Model’ (LM) module uses the preprocessed training data to build a 5-gram LM. All tokens in a given sentence in the training data are tagged with either lang1 or lang2 as described in Section 5. The prior probabilities of each lang1 and lang2 words are calculated based on their frequency in the training corpus. SRILM toolkit (Stolcke, 2002) and the tagged corpora are then used to build the LM.\(^3\) If *tokenized* preprocessing scheme is used, then the built LM is tokenized where all tokens corresponding to a certain word are assigned the same tag corresponding to the tag

\(^3\)A full description of the approach is presented in (Elfardy and Diab, 2012b).
of the original word. For example, if 
\( bjd \) is 
tagged as \( lang2 \), both “+ب”, \( b+ \) and “جد”, \( jd \) get 
tagged as \( lang2 \).

For any new untagged sentence, the ‘Language Model’ module uses the already built LM and 
the prior probabilities via Viterbi search to find 
the best sequence of tags for the given sentence. If 
there is an out-of-vocabulary word in the input sentence, the ‘Language Model’ leaves it un-
tagged.

### 4.3 MADAMIRA

Using MADAMIRA, each word in a given un-
tagged sentence is tokenized, lemmatized, and 
POS-tagged. Moreover, the MSA and English 
glosses for each morpheme of the given word 
are provided. Since MADAMIRA uses two possi-
ble underlying morphological analyzers CALIMA 
(Habash et al., 2012) and SAMA (Maamouri et al., 
2010), as part of the output, MADAMIRA indicates 
which of them is used to retrieve the glosses.

### 4.4 Named Entities List

We use the ANERGazet (Benajiba et al., 2007) to 
identify named-entities. ANERGazet consists of 
the following Gazetteers:

- **Locations**: 1,545 entries corresponding to 
names of continents, countries, cities, etc. 
(ex. \( \text{الأمازيغ} \), \( Almgrb \)) which means “Mo-

- **People**: 2,100 entries corresponding to 
names of people. (ex. 
فهد, \( fhd \));

- **Organizations**: 318 entries corresponding to 
names of Organizations such as companies 
and football teams. (ex. 
تشليبي, \( tshly \) meaning “Chelsea”)

### 4.5 Combiner

Each word in the input sentence can get differ-
ent tags from each module. Thus, the ‘Combiner’ 
module uses all of these decisions and the follow-
ing set of rules to assign the final tag to each word 
in the input sentence.

1. If the word contains any numbers or punctu-
ation, it is assigned \( \text{other} \) tag;
2. Else if the word is present in any of the 
gazetteers or if MADAMIRA assigns it 
\( \text{noun}, \text{prop} \) POS tag, the word is tagged as 
\( \text{NE} \);
3. Else if the word is (or all of its morphemes 
in the tokenized scheme are) identified by the 
LM as either \( lang1 \) or \( lang2 \), the word is as-
signed the corresponding tag;
4. Else if the word’s morphemes are assigned 
different tags, the word is assigned the \( \text{mixed} \) 
tag;
5. Else if the LM does not tag the word (i.e. the 
word is considered an out of vocabulary word 
by the LM) and:
   - If MADAMIRA retrieved the glosses 
from SAMA, the word is assigned a 
\( lang1 \) tag;
   - Else if MADAMIRA outputs that the 
glosses were retrieved from CALIMA, 
then the word is assigned a \( lang2 \) tag
   - Else if the word is still untagged (i.e. 
non-analyzable), the word is assigned 
\( lang2 \) tag.

### 5 Experiments and Results

#### 5.1 Training Phase

The training data that is used to build our LM con-
sists of two main sources:

1. **Shared-task’s training data (STT)**: 119,326 
words collected from Twitter. They are man-
ually annotated on the token-level. We split 
this corpus into:
   - **Training-set; (STT-Tr)**: 107,398 tweets 
representing 90% of STT and used for 
training the system
(b) Development-set; (STT-Dev): 11,928 words representing 10% of STT and used for tuning the system.

2. Web-log training data (WLT): 8 million words. Half of which comes from lang1 corpora while the other half is from lang2 corpora. The data is weakly labeled where all tokens in the sentence/comment are assigned the same tag according to the dialect of the forum (MSA or EDA) it was crawled from.

During the development phase, we use STT-Tr and WLT to train our system. We run several experiments to test the different setups and evaluate the performance of each of these setups on STT-Dev. Once we find the optimal configuration, we then use it to retrain the system using all of STT-Tr, STT-Dev, and WLT.

Since the size of STT is very small compared to WLT (0.1% of WLT size), the existence of six different tags in this corpus can add noise to the already weakly labeled WLT data. Thus, to make STT consistent with WLT, we changed the labels of STT as follows:

- If the number of lang1 tokens in the tweet exceeds the number of lang2 tokens; we assign all tokens in the tweet lang1 tag.
- Otherwise, all tokens in the tweet are assigned lang2 tag.

All tokens in STT tagged as NE have been used to enrich our named entity list.

5.2 Development Phase

Two different setups are tested using WLT and STT-Tr:

- **Surface form setup:** uses the basic preprocessing pipeline described earlier on both the input data and on the training data used to build the LM
- **Tokenized form setup:** uses the tokenized preprocessing pipeline described earlier on both the input data and the training data used to build the LM.

As mentioned earlier, since the size of STT-Tr is much smaller than that of WLT, this causes both datasets to be statistically incomparable. We tried increasing the weights assigned by the LM to STT-Tr by duplicating STT-Tr. We experimented with one, four, and eight copies of STT-Tr for each of the basic and tokenized experimental setups.

The shared task evaluation script has been used to evaluate each setup. The evaluation script produces two main sets of metrics. The first metric yields the accuracy, precision, recall, and $F_{\beta=1}$ score for code switching classification on the tweet-level, while the second set of metrics uses evaluates performance of each tag on the token-level. In this paper, we add an extra metric corresponding to the weighted average of the tag on the token level $F_{\beta=1}$ score in order to rank our overall performance against other participating groups in the task.

Tables 1 and 2 summarize our results for both Surface Form and Tokenized Form setups on STT-Dev. In all experiments, the Tokenized Form setup outperforms the Surface Form setup.

As shown in Table 2, the system that yields the best weighted-average token-level $F_{\beta=1}$ score (77.6%) on the development-set is **Tokenized-2**. Throughout the rest of the paper, we will use the system’s name “AIDA”; to refer to this best configuration (Tokenized-2).

Table 1: Results on STT-Dev using the tweet-level evaluation. (-1, -2, and -8) correspond to the number of copies of STT-Tr that were added to WLT.

|          | Accuracy | Precision | Recall | $F_{\beta=1}$ |
|----------|----------|-----------|--------|---------------|
| Tokenized-1 | 51.5%    | 43.7%     | 97.4%  | 60.3%         |
| Tokenized-2 | 52.5%    | 44.2%     | 97.4%  | 60.8%         |
| Tokenized-8 | 54.2%    | 45.1%     | 96.9%  | 61.6%         |
| Surface-1   | 45.4%    | 40.9%     | 99.5%  | 57.9%         |
| Surface-2   | 45.8%    | 41.1%     | 99.5%  | 58.1%         |
| Surface-8   | 46.5%    | 41.4%     | 99.5%  | 58.5%         |

Table 3 shows the distribution of each test set over the different tags.

5.3 Testing Phase

Three blind test sets have been used for the evaluation:

- **Test1:** 54,732 words of 2,363 tweets collected from some unseen users in the training set;
- **Test2:** Another 32,641 words of 1,777 tweets collected from other unseen users in the training set;
- **Surprise:** 12,017 words of 1,222 sentences from collected from Arabic commentaries.

Table 3 shows the distribution of each test set over the different tags.
Table 2: Results on STT-Dev using the token-level evaluation. (-1, -2, and -8) correspond to the number of copies of STT-Tr that were added to WLT.

|           | ambig | lang1 | lang2 | mixed | NE    | other | Avg-Fβ=1 |
|-----------|-------|-------|-------|-------|-------|-------|----------|
| Tokenized-1 | 0.0%  | 79.5% | 71.5% | 0.0%  | 83.6% | 98.9% | 77.5%    |
| Tokenized-2 | 0.0%  | 79.6% | 71.6% | 0.0%  | 83.6% | 98.9% | 77.6%    |
| Tokenized-8 | 0.0%  | 79.5% | 71.4% | 0.0%  | 83.6% | 98.9% | 77.5%    |
| Surface-1   | 0.0%  | 76.0% | 65.4% | 0.0%  | 83.6% | 98.9% | 73.5%    |
| Surface-2   | 0.0%  | 76.1% | 65.6% | 0.0%  | 83.6% | 98.9% | 73.7%    |
| Surface-8   | 0.0%  | 76.2% | 65.5% | 0.0%  | 83.6% | 98.9% | 73.7%    |

Table 3: Test sets tag distributions

Table 4: Tweet-level evaluation on Test1 set.

|           | Accuracy | Precision | Recall | Fβ=1  |
|-----------|----------|-----------|--------|-------|
| AIDA      | 45.2%    | 2.3%      | 93.8%  | 4.4%  |
| CMU       | 86.1%    | 5.2%      | 53.1%  | 9.5%  |
| A3-107    | 60.5%    | 2.5%      | 71.9%  | 4.8%  |
| IUCL      | 97.4%    | 11.1%     | 12.5%  | 11.8% |
| MSR-IN    | 94.7%    | 9.7%      | 34.4%  | 15.2% |

Table 5: Tweet-level evaluation on Test2 set.

|           | Accuracy | Precision | Recall | Fβ=1  |
|-----------|----------|-----------|--------|-------|
| AIDA      | 44.0%    | 22.2%     | 95.6%  | 36.0% |
| CMU       | 66.2%    | 29.2%     | 73.4%  | 41.7% |
| A3-107    | 46.9%    | 21.3%     | 82.3%  | 33.8% |
| IUCL      | 76.6%    | 27.1%     | 24.9%  | 26.0% |
| MSR-IN    | 71.4%    | 18.3%     | 21.2%  | 19.6% |

Table 6: Token-level evaluation on Test1 set.

|           | Accuracy | Precision | Recall | Fβ=1  |
|-----------|----------|-----------|--------|-------|
| AIDA      | 55.6%    | 16.3%     | 91.2%  | 36.0% |
| CMU       | 79.8%    | 20.7%     | 47.4%  | 9.5%  |
| A3-107    | 66.2%    | 29.2%     | 73.4%  | 9.5%  |
| IUCL      | 87.7%    | 25.0%     | 15.8%  | 19.4% |
| MSR-IN    | 84.8%    | 17.3%     | 16.7%  | 17.0% |

Table 7: Token-level evaluation on Test2 set.

Table 8: Token-level evaluation on Surprise set.

Table 9: Token-level evaluation on Test3 set.

6 Error Analysis

Tables 11, 12, and 13 show the confusion matrices of our best setup for all six tags over the three test sets. The rows represent the gold-labels while the columns represent the classes generated by our system. For example, row 4-column 2 corresponds to the percentage of words that have lang1 (i.e. MSA) gold-label and were incorrectly classified as ambig. The diagonal of each matrix corresponds to the correctly classified instances. All cells of each matrix add-up to 100%. In all three tables, it’s clear that the highest confusability is between lang1 and lang2 classes. In Test-set1, since the majority of words (81.5%) have a lang1 gold-label and a very tiny percentage (0.3%) have ambig, the majority of words (81.5%) have a lang1 gold-label and a very tiny percentage (0.3%) has

---

4The results of the other groups have been obtained from the workshop website. We use ”MSR-IN” to refer to ”MSR-India.”
Table 9: Token-level evaluation on Surprise set.

|               | ambig | lang1 | lang2 | mixed | NE  | other | Avg-F$_{\beta=1}$ |
|---------------|-------|-------|-------|-------|-----|-------|-------------------|
| AIDA          | 0.0%  | 66.6% | 81.9% | 0.0%  | 87.9%| 99.9% | 80.1%            |
| CMU           | 0.0%  | 68.0% | 82.1% | 0.0%  | 61.2%| 97.5% | 77.8%            |
| A3-107        | 0.0%  | 53.8% | 61.3% | 0.0%  | 62.3%| 96.1% | 62.6%            |
| IUCL          | 0.0%  | 48.8% | 60.9% | 0.0%  | 5.5% | 2.0%  | 46.7%            |
| MSR-IN        | 0.0%  | 56.3% | 69.8% | 0.0%  | 33.2%| 96.6% | 65.4%            |

Table 10: Overall tweet-level and token-level F$_{\beta=1}$ scores. (Averaged over the three test-sets)

|               | Tweet Avg-F$_{\beta=1}$ | Token Avg-F$_{\beta=1}$ |
|---------------|-------------------------|------------------------|
| AIDA          | 20.2%                   | 86.8%                  |
| CMU           | 24.3%                   | 86.4%                  |
| A3-107        | 18.4%                   | 76.6%                  |
| IUCL          | 18.2%                   | 61.0%                  |
| MSR-IN        | 17.1%                   | 74.2%                  |

Table 11: The token-level confusion matrix for the best performing setup on Test1 set.

|               | ambig | lang1 | lang2 | mixed | NE  | other |
|---------------|-------|-------|-------|-------|-----|-------|
| AIDA (Predicted) | 0.0%  | 0.0%  | 0.0%  | 0.0%  | 0.0%| 0.0%  |
| lang1         | 0.0%  | 74.4% | 5.7%  | 0.0%  | 1.3%| 0.0%  |
| lang2         | 0.0%  | 0.1%  | 0.2%  | 0.0%  | 0.0%| 0.0%  |
| mixed         | 0.0%  | 0.0%  | 0.0%  | 0.0%  | 0.0%| 0.0%  |
| NE            | 0.0%  | 1.5%  | 0.3%  | 0.0%  | 9.1%| 0.1%  |
| other         | 0.0%  | 0.0%  | 0.0%  | 0.0%  | 0.0%| 7.3%  |

Table 12: The token-level confusion matrix for the best performing setup on Test2 set.

|               | ambig | lang1 | lang2 | mixed | NE  | other |
|---------------|-------|-------|-------|-------|-----|-------|
| AIDA (Predicted) | 0.0%  | 0.3%  | 0.1%  | 0.0%  | 0.0%| 0.0%  |
| lang1         | 0.0%  | 28.8% | 2.8%  | 0.1%  | 0.2%| 0.1%  |
| lang2         | 0.0%  | 16.4% | 28.3% | 0.5%  | 0.2%| 0.1%  |
| mixed         | 0.0%  | 0.0%  | 0.0%  | 0.0%  | 0.0%| 0.0%  |
| NE            | 0.0%  | 1.0%  | 0.6%  | 0.0%  | 11.5%| 0.2%  |
| other         | 0.0%  | 0.0%  | 0.0%  | 0.0%  | 0.0%| 8.9%  |

Table 13: The token-level confusion matrix for the best performing setup on Surprise set.

is “lang1” and the clitic ح. H “will” is “lang2”.

Examples 4 and 5 show instances of the confusability between “lang1” and “lang2” classes. Both words in these two examples can belong to either one of “lang1” and “lang2” classes depending on the context.

One interesting observation is that AIDA, outperforms all other systems tagging named-entities. This suggests the robustness of the NER approach used by AIDA.

The performance on the other tags varies across the three test-sets.
7 Conclusion and Future Work

In this work, we adapt a previously proposed system for automatic detection of code switching in informal Arabic text to handle twitter data. We experiment with several setups and report the results on two twitter datasets and a surprise-genre test-set, all of which were generated for the shared task at EMNLP workshop for Computational Approaches to Code Switching. In the future we plan on handling other Arabic dialects such as Levantine, Iraqi and Moroccan Arabic as well as adapting the system to other genres.

8 Acknowledgment

This work has been funded by the NSF CRI Code Switching Project.

We would like to thank Mahmoud Ghoneim for his thorough review of the paper and the data. We also thank the anonymous reviewer for the useful comments.

References

Yassine Benajiba, Paolo Rosso, and Jos Miguel Benedruiz. 2007. Anersy: An arabic named entity recognition system based on maximum entropy. In Proceedings of CICLing-2007.

Fadi Biadsy, Julia Hirschberg, and Nizar Habash. 2009. Spoken arabic dialect identification using phonotactic modeling. In Proceedings of the Workshop on Computational Approaches to Semitic Languages at the meeting of the European Association for Computational Linguistics (EACL), Athens, Greece.

Houda Bouamor, Nizar Habash, and Kemal Oflazer. 2014. A multidialectal parallel corpus of arabic. In Proceedings of LREC.

Ryan Cotterell and Chris Callison-Burch. 2014. A multi-dialect, multi-genre corpus of informal written
arabic. In Proceedings of the Language Resources and Evaluation Conference (LREC).

Ryan Cotterell, Adithya Renduchintala, Naomi Saphra, and Chris Callison-Burch. 2014. An Algerian arabic-french code-switched corpus. In LREC Workshop on Free/Open-Source Arabic Corpora and Corpora Processing Tools.

Heba Elfardy and Mona Diab. 2012a. Simplified guidelines for the creation of large scale dialectal arabic annotations. In Proceedings of LREC.

Heba Elfardy and Mona Diab. 2012b. Token level identification of linguistic code switching. In Proceedings of COLING, Mumbai, India.

Heba Elfardy and Mona Diab. 2013. Sentence-Level Dialect Identification in Arabic. In Proceedings of ACL2013, Sofia, Bulgaria, August.

Heba Elfardy, Mohamed Al-Badrashiny, and Mona Diab. 2013. Code Switch Point Detection in Arabic. In Proceedings of the 18th International Conference on Application of Natural Language to Information Systems (NLDB2013), MediaCity, UK, June.

Ramy Eskander, Mohamed Al-Badrashiny, Nizar Habash, and Owen Rambow. 2014. Foreign words and the automatic processing of arabic social media text written in roman script. In Proceedings of the First Workshop on Computational Approaches to Code-Switching. EMNLP 2014, Conference on Empirical Methods in Natural Language Processing, October, 2014, Doha, Qatar.

Ferguson. 1959. Diglossia. Word 15. 325340.

Nizar Habash and Fatiha Sadat. 2006. Arabic preprocessing schemes for statistical machine translation.

Nizar Habash, Ramy Eskander, and AbdelAti Hawwari. 2012. A Morphological Analyzer for Egyptian Arabic. In NAACL-HLT 2012 Workshop on Computational Morphology and Phonology (SIGMORPHON2012), pages 1–9.

Nizar Habash. 2010. Introduction to arabic natural language processing. Advances in neural information processing systems.

Mohamed Maamouri, Dave Graff, Basma Bouziri, Sondos Krouna, Ann Bies, and Seth Kulick. 2010. Ldc standard arabic morphological analyzer (sama) version 3.1.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Maneoj Pooleery, Owen Rambow, and Ryan M. Roth. 2014. MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic. In Proceedings of LREC, Reykjavik, Iceland.

Wael Salloum and Nizar Habash. 2011. Dialectal to standard arabic paraphrasing to improve arabic-english statistical machine translation. In Proceedings of the First Workshop on Algorithms and Resources for Modelling of Dialects and Language Varieties. Association for Computational Linguistics.

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 In Proceedings of the First Workshop on Computational Approaches to Code-Switching. EMNLP 2014, Conference on Empirical Methods in Natural Language Processing, October, 2014, Doha, Qatar.

Andreas Stolcke. 2002. Srlm an extensible language modeling toolkit. In Proceedings of ICSLP.

Omar F Zaidan and Chris Callison-Burch. 2011. The arabic online commentary dataset: an annotated dataset of informal arabic with high dialectal content. In ACL.