Foreign Words and the Automatic Processing of Arabic Social Media Text Written in Roman Script

Ramy Eskander, Mohamed Al-Badrashiny[†], Nizar Habash[‡] and Owen Rambow
Center for Computational Learning Systems, Columbia University
{reskander,rambow}@ccls.columbia.edu
[†]Department of Computer Science, The George Washington University
[†]badrashiny@gwu.edu
[‡]Computer Science Department, New York University Abu Dhabi
[‡]nizar.habash@nyu.edu

Abstract

Arabic on social media has all the properties of any language on social media that make it tough for natural language processing, plus some specific problems. These include diglossia, the use of an alternative alphabet (Roman), and code switching with foreign languages. In this paper, we present a system which can process Arabic written in Roman alphabet ("Arabizi"). It identifies whether each word is a foreign word or one of another four categories (Arabic, name, punctuation, sound), and transliterates Arabic words and names into the Arabic alphabet. We obtain an overall system performance of 83.8% on an unseen test set.

1 Introduction

Written language used in social media shows differences from that in other written genres: the vocabulary is informal (and sometimes the syntax is as well); there are intentional deviations from standard orthography (such as repeated letters for emphasis); there are typos; writers use non-standard abbreviations; non-linguistic sounds are written (*haha*); punctuation is used creatively; non-linguistic signs such as emoticons often compensate for the absence of a broader communication channel in written communication (which excludes, for example, prosody or visual feedback); and, most importantly for this paper, there frequently is code switching. These facts pose a well-known problem for natural language processing of social media texts, which has become an area of interest as applications such as sentiment analysis, information extraction, and machine translation turn to this genre.

This situation is exacerbated in the case of Arabic social media. There are three principal reasons. First, the Arabic language is a collection of varieties: Modern Standard Arabic (MSA), which is used in formal settings, and different forms of Dialectal Arabic (DA), which are commonly used informally. This situation is referred to as "diglossia". MSA has a standard orthography, while the dialects do not. What is used in Arabic social media is typically DA. This means that there is no standard orthography to begin with, resulting in an even broader variation in orthographic forms found. Diglossia is seen in other linguistic communities as well, including German-speaking Switzerland, in the Czech Republic, or to a somewhat lesser extent among French speakers. Second, while both MSA and DA are commonly written in the Arabic script, DA is sometimes written in the Roman script. Arabic written in Roman is often called "Arabizi". It is common in other linguistic communities as well to write informal communication in the Roman alphabet rather than in the native writing system, for example, among South Asians. And third, educated speakers of Arabic are often bilingual or near-bilingual speakers of another language as well (such as English or French), and will code switch between DA and the foreign language in the same utterance (and sometimes MSA as well). As is well known, code switching is common in many linguistic communities, for example among South Asians.

In this paper, we investigate the issue of processing Arabizi input with code switching. There are two tasks: identification of tokens that are not DA or MSA (and should not be transliterated into Arabic script for downstream processing), and then the transliteration into Arabic script of the parts identified as DA or MSA. In this paper, we

Proceedings of The First Workshop on Computational Approaches to Code Switching, pages 1–12,
October 25, 2014, Doha, Qatar.

use as a black box an existing component that we developed to transliterate from Arabizi to Arabic script (Al-Badrashiny et al., 2014). This paper concentrates on the task of identifying which tokens should be transliterated. A recent release of annotated data by the Linguistic Data Consortium (LDC, 2014c; Bies et al., 2014) has enabled novel research on this topic. The corpus provides each token with a tag, as well as a transliteration if appropriate. The tags identify foreign words, as well as Arabic words, names, punctuation, and sounds. Only Arabic words and names are transliterated. (Note that code switching is not distinguished from borrowing.) Emoticons, which may be isolated or part of an input token, are also identified, and converted into a conventional symbol (#). This paper presents taggers for the tags, and an end-to-end system which takes Arabizi input and produces a complex output which consists of a tag for each input token and a transliteration of Arabic words and names into the Arabic script. To our knowledge, this is the first system that handles the complete task as defined by the LDC data. This paper focuses on the task of identifying foreign words (as well as the other tags), on creating a single system, and on evaluating the system as a whole.

This paper makes three main contributions. First, we clearly define the computational problem of dealing with social media Arabizi, and propose a new formulation of the evaluation metric for the LDC corpus. Second, we present novel modules for the detection of foreign words as well as of emoticons, sounds, punctuation marks, and names in Arabizi. Third, we compose a single system from the various components, and evaluate the complete system.

This paper is structured as follows. We start by presenting related work (Section 2), and then we present relevant linguistic facts and explain how the data is annotated (Section 3). After summarizing our system architecture (Section 4) and experimental setup (Section 5), we present our systems for tagging in Sections 6, 7 and 8. The evaluation results are presented in Section 9.

2 Related Work

While natural language processing for English in social media has attracted considerable attention recently (Clark and Araki, 2011; Gimpel et al., 2011; Gouws et al., 2011; Ritter et al., 2011; Derczynski et al., 2013), there has not been much work on Arabic yet. We give a brief summary of relevant work on Arabic.

Darwish et al. (2012) discuss NLP problems in retrieving Arabic microblogs (tweets). They discuss many of the same issues we do, notably the problems arising from the use of DA such as the lack of a standard orthography. However, they do not deal with DA written in the Roman alphabet (though they do discuss non-Arabic characters).

There is some work on code switching between Modern Standard Arabic (MSA) and dialectal Arabic (DA). Zaidan and Callison-Burch (2011) are interested in this problem at the inter-sentence level. They crawl a large dataset of MSA-DA news commentaries. They use Amazon Mechanical Turk to annotate the dataset at the sentence level. Then they use a language modeling approach to predict the class (MSA or DA) for an unseen sentence. There is other work on dialect identification, such as AIDA (Elfardy et al., 2013; Elfardy et al., 2014). In AIDA, some statistical and morphological analyses are applied to capture code switching between MSA and DA within the same sentence. Each word in the sentence is tagged to be either DA or MSA based on the context. The tagging process mainly depends on the language modeling (LM) approach, but if a word is unknown in the LM, then its tag is assigned through MADAMIRA, a morphological disambiguator Pasha et al. (2014).

Lui et al. (2014) proposed a system that does language identification in multilingual documents, using a generative mixture model that is based on supervised topic modeling algorithms. This is similar to our work in terms of identifying code switching. However, our system deals with Arabizi, a non-standard orthography with high variability, making the identification task much harder.

Concerning specifically NLP for Arabizi, Darwish (2013) (published in an updated version as (Darwish, 2014)) is similar to our work in that he identifies English in Arabizi text and he also transliterates Arabic text from Arabizi to Arabic script. We compare our transliteration method to his in Al-Badrashiny et al. (2014). For identification of non-Arabic words in Arabizi, Darwish (2013) uses word and sequence-level features with CRF modeling; while we use SVMs and decision trees. Darwish (2013) identifies three tags: Arabic, foreign and others (such as email addresses and URLs). In contrast, we identify a bigger set: Arabic, foreign, names, sounds, punctuation

and emoticons. Furthermore, Darwish (2013) uses around 5K words for training his taggers and 3.5K words for testing; this is considerably smaller than our training and test sets of 113K and 32K words, respectively.

Chalabi and Gerges (2012) presented a hybrid approach for Arabizi transliteration. Their work does not address the detection of English words, punctuation, emoticons, and so on. They also do not handle English when mixed with Arabizi.

Voss et al. (2014) deal with exactly the problem of classifying tokens in Arabizi as Arabic or not. More specifically, they deal with Moroccan Arabic, and with both French and English, meaning they do a three-way classification. There are many differences between our work and theirs: they have noisy training data, and they have a much more balanced test set. They also only deal with foreignness, and do not address the other tags we deal with, nor do they actually discuss transliteration itself.

3 Linguistic Facts and Data Annotation

3.1 Arabizi

Arabizi refers to Arabic written using the Roman script (Darwish, 2013; Voss et al., 2014). Arabizi orthography is spontaneous and has no standard references, although there are numerous commonly used conventions making specific usage of the so-called Arabic numerals and punctuation in addition to Roman script letters. Arabizi is commonly used by Arabic speakers to write mostly in dialectal Arabic in social media, SMS and chat applications.

Arabizi orthography decisions mainly depend on a phoneme-to-grapheme mapping between the Arabic pronunciation and the Roman script. This is largely based on the phoneme-to-grapheme mapping used in English (in Middle Eastern Arab countries) or French (in Western North African Arab countries). Since there is no standard orthography for Arabizi, it is *not* a simple transliteration of Arabic. For example, in Arabizi, words omit vowels far less frequently than is done when writers follow standard Arabic orthography. Furthermore, there are several cases of many-to-many mappings between Arabic phonemes and Roman script letters: for example, the letter "t" is used to represent the sound of the Arabic letters ت *t*[1] and ط *T* (which itself can be also be represented using the digit "6").

Text written in Arabizi also tends to have a large number of foreign words, that are either borrowings such as *telephone*, or code switching, such as *love you!*. Note that Arabizi often uses the source language orthography for borrowings (especially recent borrowings), even if the Arabic pronunciation is somewhat modified. As a result, distinguishing borrowings from code switching is, as is usually the case, hard. And, as in any language used in social media and chat, Arabizi may also include abbreviations, such as *isa* which means إن شاء الله *Ăn šA' Allh* 'God willing' and *lol* 'laugh out loud'.

The rows marked with **Arabizi** in Figure 1 demonstrate some of the salient features of Arabizi. The constructed example in the figure is of an SMS conversation in Egyptian Arabic.

3.2 Data Annotation

The data set we use in this paper was created by the Linguistic Data Consortium (Bies et al., 2014; LDC, 2014a; LDC, 2014b; LDC, 2014c). We summarize below the annotation decisions. The system we present in this paper aims at predicting exactly this annotation automatically. The input text is initially segregated into Arabic script and Arabizi. Arabic script text is not modified in any way. Arabizi text undergoes two sets of annotation decisions: Arabizi word tagging and Arabizi-to-Arabic transliteration. All of the Arabizi annotations are initially done using an automatic process (Al-Badrashiny et al., 2014) and then followed by manual correction and validation.

Arabizi Word Tagging Each Arabizi word receives one of the following five tags:

- *Foreign* All words from languages other than Arabic are tagged as *Foreign* if they would be kept in the same orthographic form when translated into their source language (which in our corpus is almost always English). Thus, non-Arabic words that include Arabic affixes are not tagged as *Foreign*. The definition of "foreign" thus means that uninflected borrowings spelled as in the source language orthography are tagged as "foreign", while borrowings that are spelled differently, as well as borrowing that have been inflected

[1] Arabic transliteration is presented in the Habash-Soudi-Buckwalter scheme (Habash et al., 2007): (in alphabetical order) *AbtθjHxdðrzsšSDTĎςγfqklmnhwy* and the additional symbols: ' ء, Â أ, Ă إ, Ā آ, ŵ ؤ , ŷ ئ , ى ħ ة, ý ى.

(1)

Arabizi	Youmna	i	need	to	know	anti	gaya	wala	la2	?
Tag	*Name*	*Foreign*	*Foreign*	*Foreign*	*Foreign*	*Arabic*	*Arabic*	*Arabic*	*Arabic*	*Punct*
Arabic	يمنى	اي	نيد	تو	نو	انتي	جاية	ولا	لأ	؟
	ymný	*Ay*	*nyd*	*tw*	*nw*	*Anty*	*jAyħ*	*wlA*	*lÂ*	*?*
English	Youmna	I	need	to	know	you	coming	or	not	?

(2)

Arabizi	Mmmm	ok	ana	7aseb	el	sho3’l	now	w	ageelk	isa	:-)
Tag	*Sound*	*Foreign*	*Arabic*	*Arabic*	*Arabic*	*Arabic*	*Foreign*	*Arabic*	*Arabic*	*Arabic*	*Arabic*
Arabic	مم	اوكيه	انا	حاسيب	ال[+]	شغل	ناو	و[+]	اجي[-]لك	ان[-]شاء[-]الله	#
	mmm	*Awkyh*	*AnA*	*HAsyb*	*Al[+]*	*šgl*	*nAw*	*w[+]*	*Ajy[-]lk*	*An[-]šA’[-]Allh*	*#*
English	mmm	OK	I	will-leave	the	work	now	and	I-come-to-you	God-willing	:-)

(3)

Arabizi	qishta!:D
Tag	*Arabic*
Arabic	قشطة![-]#
	qšTħ![-]#
English	cream!:D (slang for *cool!*)

Figure 1: A short constructed SMS conversation written in Arabizi together with annotation of word tags and transliteration into Arabic script. A Romanized transliteration of the Arabic script and English glosses are provided for clarity. The cells with gray background are the parts of the output that we evaluate.

following Arabic morphology, are not tagged as “foreign” (even if the stem is spelled as in the source language, such as *Almobile*). The Arabic transliterations of these words are not manually corrected.

- *Punct* Punctuation marks are a set of conventional signs that are used to aid interpretation by indicating division of text into sentences and clauses, etc. Examples of punctuation marks are the semicolon **;**, the exclamation mark **!** and the right brace }. Emoticons are not considered punctuation and are handled as part of the transliteration task discussed below.
- *Sound* Sounds are a list of interjections that have no grammatical meaning, but mimic non-linguistic sounds that humans make, and that often signify emotions. Examples of sounds are *hahaha* (laughing), *hmm* (wondering) and *eww* (being disgusted). It is common to stretch sounds out to make them stronger, i.e., to express more intense emotions. For example, *hmm* could be stretched out into *hmmmmm* to express a stronger feeling of wondering. The Arabic transliterations of these words are not manually corrected.
- *Name* Proper names are tagged as such and later manually corrected.
- *Arabic* All other words are tagged as *Arabic* and are later manually corrected.

See the rows marked with **Tag** in Figure 1 for examples of these different tags. It is important to point out that the annotation of this data was intended to serve a project focusing on machine translation from dialectal Arabic into English. This goal influenced some of the annotation decisions and was part of the reason for this selection of word tags.

Arabizi-to-Arabic Transliteration The second annotation task is about converting Arabizi to an Arabic-script-based orthography. Since, dialectal Arabic including Egyptian Arabic has no standard orthography in Arabic script, the annotation uses a conventionalized orthography for Dialectal Arabic called CODA (Habash et al., 2012a; Eskander et al., 2013; Zribi et al., 2014). Every word has a single orthographic representation in CODA.

In the corpus we use, only words tagged as *Arabic* or *Name* are manually checked and corrected. The transliteration respects the white-space boundaries of the original Arabizi words. In cases where an Arabizi word represents a prefix or suffix that should be joined in CODA to the next or previous word, a [+] symbol is added to mark this decision. Similarly, for Arabizi words that should be split into multiple CODA words, the CODA words are written with added [-] symbol delimiting the word boundaries.

The Arabic transliteration task also includes handling emoticons. Emoticons are digital icons or sequences of keyboard symbols serving to represent facial expressions or to convey the writer’s emotions. Examples of emoticons are :d, :-(, O.O and ♡ used to represent laughing, sadness, being surprised and positive emotion, respectively. All emoticons, whether free-standing or attached to a

word, are replaced by a single hash symbol (#). Free-standing emoticons are tagged as *Arabic*. Attached emoticons are not tagged separately; the word they are attached to is tagged according to the usual rules. See Figure 1 for examples of these different decisions.

Since words tagged as *Foreign*, *Punct*, or *Sound* are not manually transliterated in the corpus, in our performance evaluation we combine the decisions of tags and transliteration. For foreign words, punctuation and sounds, we only consider the tags for accuracy computations; in contrast, for names and Arabic words, we consider both the tag and transliteration.

4 System Architecture

Figure 2 represent the overall architecture of our system. We distinguish below between existing components that we use and novel extensions that we contribute in this paper.

4.1 Existing Arabization System

For the core component of Arabizi-to-Arabic transliteration, we use a previously published system (Al-Badrashiny et al., 2014), which converts Arabizi into Arabic text following CODA conventions (see Section 3). The existing system uses a finite state transducer trained on 8,500 Arabizi-to-Arabic transliteration pairs at the character level to obtain a large number of possible transliterations for the input Arabizi words. The generated list is then filtered using a dialectal Arabic morphological analyzer. Finally, the best choice for each input word is selected using a language model. We use this component as a black box except that we retrain it using additional training data. In Figure 2, this component is represented using a central black box.

4.2 Novel Extension

In this paper, we add **Word Type Tagging** as a new set of modules. We tag the Arabizi words into five categories as discussed above: Arabic, Foreign, Names, Sounds, and Punctuation. Figure 2 illustrates the full proposed system. First, we process the Arabizi input to do punctuation and sound tagging, along with emoticon detection. Then we run the transliteration system to produce the corresponding Arabic transliteration. The Arabizi input and Arabic output are then used together to do name tagging and foreign word tagging. The *Arabic* tag is assigned to all untagged words, i.e., words not tagged as Foreign, Names, Sounds, or Punctuation. The outputs from all steps are then combined to produce the final Arabic transliteration along with the tag.

5 Experimental Setup

5.1 Data Sets

We define the following sets of data:

- *Train-S*: A small size dataset that is used to train all taggers in all experiments to determine the best performing setup (feature engineering).
- *Train-L*: A larger size dataset that is used to train the best performing setup.
- *Dev*: The development set that is used to measure the system performance in all experiments
- *Test*: A blind set that is used to test the best system (LDC, 2014a).

The training and development sets are extracted from (LDC, 2014b). Table 1 represents the tags distribution in each dataset. Almost one of every five words is not Arabic text and around one of every 10 words is foreign.

5.2 Arabizi-to-Arabic Transliteration Accuracy

For the Arabizi-to-Arabic transliteration system, we report on using the two training data sets with two modifications. First, we include the 8,500 word pairs from Al-Badrashiny et al. (2014), namely 2,200 Arabizi-to-Arabic script pairs from the training data used by Darwish (2013) (manually revised to be CODA-compliant) and about 6,300 pairs of proper names in Arabic and English from the Buckwalter Arabic Morphological Analyzer (Buckwalter, 2004). (Since these pairs are not tagged, we do not use them to train the taggers.) Second, we exclude all the foreign tagged words from training the transliteration system since they were not manually corrected.

Table 2 shows the overall transliteration accuracy of Arabic words and names only, using different training data sets and evaluating on *Dev* (as determined by the gold standard). The accuracy when using the original Arabizi-to-Arabic transliteration system from Al-Badrashiny et al. (2014) gives an accuracy of 68.6%. Retraining it on *Train-S* improves the accuracy to 76.9%. The accuracy goes up further to 79.5% when using the

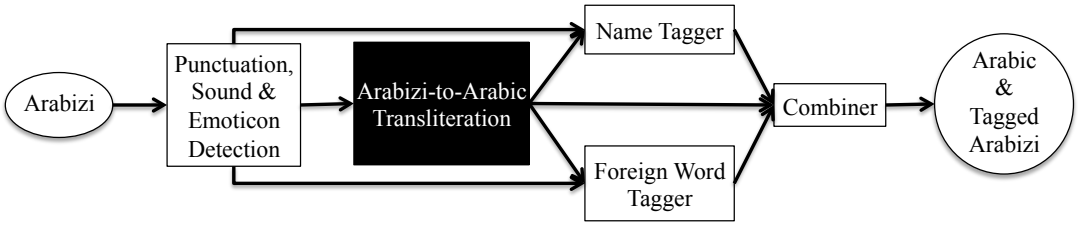

Figure 2: The architecture of our complete Arabizi processing system. The "Punctuation, Sound and Emoticon Detection" component does labeling that is read by the "Name" and "Foreign Word" taggers, While the actual Aribizi-to-Arabic transliteration system is used as a black box.

Data	# Words	Arabic	Foreign	Name	Sound	Punct	Emoticon
Train-S	21,950	80.5%	12.1%	2.8%	1.7%	1.3%	1.6%
Train-L	113,490	82.3%	9.8%	2.4%	1.8%	1.1%	2.6%
Dev	5,061	76.3%	16.2%	2.9%	1.8%	1.2%	1.5%
Test	31,717	86.1%	6.0%	2.7%	1.6%	0.9%	2.8%

Table 1: Dataset Statistics

Data	Translit. Acc.
Al-Badrashiny et al. (2014)	68.6%
Train-S	76.9%
Train-L	79.5%

Table 2: Transliteration accuracy of Arabic words and names when using different training sets and evaluating on *Dev*

bigger training set *Train-L*. The overall transliteration accuracy of Arabic words and names on *Test* using the bigger training set *Train-L* is 83.6%.

6 Tagging Punctuation, Emoticons and Sounds

6.1 Approach

We start the tagging process by detecting three types of closed classes: punctuation, sounds and emoticons. Simple regular expressions perform very well at detecting their occurrence in text. The regular expressions are applied to the Arabizi input, word by word, after lower-casing, since both emoticons and sounds could contain either small or capital letters.

Since emoticons can be composed of just concatenated punctuation marks, their detection is required before punctuation is tagged. Once detected, emoticons are replaced by #. Then punctuation marks are detected. If a *non-emoticon* word is only composed of punctuation marks, then it gets tagged as *Punct*. Sounds are targeted next. A word gets tagged as *Sound* if it matches the sound detection expression, after stripping out any attached punctuation marks and/or emoticons.

6.2 Results

Table 6 in Section 9 shows the accuracy, recall, precision and F-score for the classification of the *Punct* and *Sound* tags and detection of emoticons. Since emoticons can be part of another word, and in that case do not receive a specific tag (as specified in the annotation guidelines by the LDC), emoticon evaluation is concerned with the number of detected emoticons within an Arabizi word, as opposed to a binary tagging decision. In other words, emoticon identification is counted as correct ("positive") if the number of detected emoticons in a word is correct in the test token. The *Punct* and *Sound* tags represent standard binary classification tasks and are evaluated in the usual way.

7 Tagging Names

7.1 Approach

We consider the following set of binary features for learning a model of name tagging. The features are used either separately or combined using a modeling classifier implemented with decision trees.

- **Capitalization** A word is considered a name if the first letter in Arabizi is capitalized.

- **MADAMIRA** MADAMIRA is a system for morphological analysis and disambiguation of Arabic (Pasha et al., 2014). We run MADAMIRA on the Arabic output after running the Arabizi-to-Arabic transliteration. If the selected part-of-speech (POS) of a word is proper noun *(NOUN_PROP)*, then the word is tagged as *Name*.
- **CALIMA** CALIMA is a morphological analyzer for Egyptian Arabic (Habash et al., 2012b). If the Arabic transliteration of a given Arabizi word has a possible proper noun analysis in CALIMA, then the word is tagged as *Name*.
- **Maximum Likelihood Estimate (MLE)** An Arabizi word gets assigned the *Name* tag if *Name* is the most associated tag for that word in the training set.
- **Tharwa** Tharwa is a large scale Egyptian Arabic-MSA-English lexicon that includes POS tag information (Diab et al., 2014). If an Arabizi word appears in Tharwa as an English gloss with a proper noun POS, then it is tagged as *Name*.
- **Name Language Model** We use a list of 280K unique lower-cased English words associated with their probability of appearing capitalized (Habash, 2009). When using this feature, any probability that is not equal to one is rounded to zero.

All the features above are modeled after case-lowering the Arabizi input, and removing speech effects. Any attached punctuation marks and/or emoticons are stripped out. One exception is the capitalization feature, where the case of the first letter of the Arabizi word is preserved. The techniques above are then combined together using decision trees. In this approach, the words tagged as *Name* are given a weight that balances their infrequent occurrence in the data.

7.2 Results

Table 3 shows the performance of the *Name* tagging on *Dev* using *Train-S*. The best results are obtained when looking up the MLE value in the training data, with an accuracy and F-score of 97.8% and 56.0%, respectively. When using *Train-L*, the accuracy and F-score given by MLE go up to 98.1% and 63.9%, respectively. See Table 6. The performance of the combined approach does not outperform the most effective single classifier, MLE. This is because adding other features decreases the precision by an amount that exceeds the increase in the recall.

Feature	Accuracy	Recall	Precision	F-Score
Capitalization	85.6	28.3	6.4	10.4
MADAMIRA	95.9	24.8	28.3	26.5
CALIMA	86.3	50.3	10.9	17.9
MLE	**97.8**	46.9	69.4	**56.0**
THARWA	96.3	22.8	33.0	26.9
NAME-LM	84.5	30.3	6.3	10.4
All Combined (Decision Trees)	97.7	49.7	63.2	55.6

Table 3: *Name* tagging results on *Dev* with *Train-S*

8 Tagging Foreign Words

As shown earlier, around 10% of all words in Arabizi text are foreign, mostly English in our data set. Tagging foreign words is challenging since there are many words that can be either Arabic (in Arabizi) or a word in a foreign languages. For example the Arabizi word *mesh* can refer to the English reading or the Arabic word مش *mš* 'not'. Therefore, simple dictionary lookup is not sufficient to determine whether a word is Arabic or Foreign. Our target in this section is to identify the foreign words in the input Arabizi text .

8.1 Baseline Experiments

We define a foreignness index formula that gives each word a score given its unigram probabilities against Arabic and English language models (LMs).

$$\varepsilon(w) = \alpha P_E(w) + (1-\alpha)(1 - P_A(w_t)) \quad (1)$$

$\varepsilon(w)$ is the foreignness *score* of the Arabizi word w. $P_E(w)$ is the unigram probability of w in the English LM, and $P_A(w_t)$ is the unigram probability in the Arabic LM of the transliteration into Arabic (w_t) proposed by our system for the Arabizi word w. α is a tuning parameter varying from zero to one. From equation 1 we define the minimum and maximum ε values as follows:

$$\begin{aligned} \varepsilon_{min} &= \alpha P_{E_{min}} + (1-\alpha)(1 - P_{A_{max}}) \\ \varepsilon_{max} &= \alpha P_{E_{max}} + (1-\alpha)(1 - P_{A_{min}}) \end{aligned} \quad (2)$$

Where $P_{E_{min}}$ and $P_{E_{max}}$ are the minimum and maximum uni-gram probabilities in the English LM. And $P_{A_{min}}$ and $P_{A_{max}}$ are the minimum

and maximum uni-gram probabilities in the Arabic LM. The foreignness index $Foreignness(w)$ is the normalized foreignness *score* derived using equations 1 and 2 as follow:

$$Foreignness\,(w) = \frac{\varepsilon\,(w) - \varepsilon_{min}}{\varepsilon_{max} - \varepsilon_{min}} \quad (3)$$

If the foreignness index of a word is higher than a certain threshold β, we consider the word *Foreign*. We define three baseline experiments as follows:

- **FW-index-manual**: Use brute force search to find the best α and β that maximize the foreign words tagging on *Dev*.
- **FW-index-SVM**: Use the best α from above and train an SVM model using the foreignness index as sole feature. Then use this model to classify each word in *Dev*.
- **LM-lookup**: The word is said to be *Foreign* if it exists in the English LM and does not exist in the Arabic LM.

8.2 Machine Learning Experiments

We conducted a suite of experiments by training different machine learning techniques using WEKA (Hall et al., 2009) on the following groups of features. We performed a two-stage feature exploration, where we did an exhaustive search over all features in each group in the first phase, and then exhaustively searched over all retained feature groups. In addition, we also performed an exhaustive search over all features in the first three groups.

- **Word n-gram features**: Run the input Arabizi word through an English LM and the corresponding Arabic transliteration through an Arabic LM to get the set of features that are defined in "Group1" in Table 4. Then find the best combination of features that maximizes the F-score on *Dev*.
- **FW-char-n-gram features**: Run the input Arabizi word through a character-level n-gram LM of the Arabizi words that are tagged as foreign in the training data. We get the set of features that are defined in "Group2" in Table 4. Then find the best feature combination from this group that maximizes the F-score on *Dev*.
- **AR-char-n-gram features**: Run the input Arabizi word through a character-level n-gram LM of the Arabizi words that are tagged as non-foreign in the training data. We get the set of features that are defined in "Group3" in Table 4. Then find the best feature that maximizes the F-score on *Dev*.
- **Word identity**: Use the input Arabizi word to get all features that are defined in "Group4" in Table 4. Then find the best combination of features that maximizes the F-score on *Dev*.
- **Word properties**: Use the input Arabizi word to get all features that are defined in "Group5" in Table 4. Then find the best combination of features that maximizes the F-score on *Dev*.
- **Best-of-all-groups**: Use the best selected set of features from each of the above experiments. Then find the best combination of these features that maximizes the F-score on *Dev*.
- **All-features**: Use all features from all groups.
- **Probabilistic-features-only**: Find the best combination of features from "Group1", "Group2", and "Group3" in Table 4 that maximizes the F-score on *Dev*.

Group	Description
Group1	Uni and bi-grams probabilities from English and Arabic LMs
Group2	1,2,3,4, and 5 characters level n-grams of foreign words
Group3	1,2,3,4, and 5 characters level n-grams of Arabic words
Group4	Use the Arabizi word itself as a feature
	Was the input Arabizi word tagged as foreign in the gold training data?
	Was the input Arabizi word tagged as Arabic in the gold training data?
Group5	Does the input word has speech effects?
	Word length
	Is the Arabizi word capitalized?

Table 4: List of the different features that are used in the foreign word tagging

8.3 Results

Table 5 shows the results on *Dev* using *Train-S*. It can be seen that the decision tree classifier is doing better than the SVM except in the "Word properties" and "All-features" experiments. The best performing setup is "Probabilistic-features-only" with decision trees which has 87.3% F-score. The best selected features are EN-Unigram, AR-char-2-grams, FW-char-1-grams, FW-char-2-grams, FW-char-5-grams.

Experiment	Recall	Precision	F-Score	Classifier	Selected Features
LM-lookup	7.6	**95.4**	14.1		
FW-index-manual	75.0	51.0	60.7		α =0.8 , β = 0.23
FW-index-SVM	4.0	89.0	7.7	SVM	
Word n-gram features	76.7	73.2	74.9	SVM	AR-unigram, EN-unigram
AR-char-n-gram features	55.4	34.8	42.8		AR-char-4-grams
FW-char-n-gram features	42.4	52.2	46.8		FW-char-3-grams
Word properties	2.4	28.6	4.5		Has-speech-effect, Word-length, Is-capitalized
Word identity	70.3	63.0	66.4		FW-tagged-list
Best-of-all-groups	82.1	76.1	79.0		AR-unigram, EN-unigram, Word-length
All-features	69.4	87.7	77.5		All features from all groups
Probabilistic-features-only	84.5	80.6	82.5		AR-unigram, EN-unigram, AR-char-3-grams, FW-char-3-grams
Word n-gram features	82.8	80.5	81.6	Decision-Tree	AR-unigram, EN-unigram
AR-char-n-gram features	80.6	63.2	70.8		AR-char-5-grams
FW-char-n-gram features	73.8	76.3	75.0		FW-char-3-grams
Word properties	1.9	25.4	3.6		Has-speech-effect, Word-length
Word identity	73.2	60.9	66.5		FW-tagged-list
Best-of-all-groups	87.0	81.5	84.1		AR-unigram, EN-unigram, AR-char-5-grams, FW-char-3-grams
All-features	**92.0**	53.4	67.6		All features from all groups
Probabilistic-features-only	89.9	84.9	**87.3**		**EN-Unigram, AR-char-2-grams, FW-char-1-grams, FW-char-2-grams, FW-char-5-grams**

Table 5: Foreign words tagging results on *Dev* in terms of F-score (%).

9 System Evaluation

9.1 Development and Blind Test Results

We report the results on *Dev* using *Train-L* and with the best settings determined in the previous three sections. Table 6 summarizes the recall, precision and F-score results for the classification of the *Punct*, *Sound*, *Foreign*, *Name* and *Arabic* tags, in addition to emoticon detection.

We report our results on *Test*, our blind test set, using *Train-L* and with the best settings determined in the previous three sections in Table 7.

The punctuation, sounds and emoticons have high F-scores but lower than expected. This is likely due to the limitations of the regular expressions used. The performance on these tags drops further on the test set. A similar drop is seen for the *Foreign* tag. *Name* is the hardest tag overall. But it performs slightly better in test compared to the development set, and so does the *Arabic* tag.

Tag	Accuracy	Recall	Precision	F-Score
Punct	99.8	100.0	88.7	94.0
Sound	99.4	93.5	78.9	85.6
Foreign	95.8	91.6	84.0	87.6
Name	98.1	57.5	71.8	63.9
Arabic	94.5	95.6	97.3	96.4
Emoticon Detection	100.0	97.5	98.7	98.1

Table 6: Tagging results on *Dev* using *Train-L*

Tag	Accuracy	Recall	Precision	F-Score
Punct	99.8	98.2	80.1	88.3
Sound	99.3	87.4	74.2	80.3
Foreign	96.5	92.3	64.3	75.8
Name	98.6	53.7	90.2	67.3
Arabic	95.4	96.3	98.5	97.4
Emoticon Detection	99.2	85.3	93.6	89.3

Table 7: Tagging results on *Test* using *Train-L*

9.2 Overall System Evaluation

In this subsection we report on evaluating the overall system accuracy. This includes the correct tagging and Arabizi to Arabic transliteration. However, since there is no manually annotated gold transliteration for foreign words, punctuation, or sounds into Arabic, we cannot compare the system transliteration of foreign words to the gold transliteration. Thus, we define the following metric to judge the overall system accuracy.

Overall System Accuracy Metric A word is said to be correctly transliterated according to the following rules:

1. If the gold tag is anything other than *Arabic* and *Name*, the produced tag must match the gold tag.
2. If the gold tag is either *Arabic* or *Name*, the produced tag and the produced transliteration must both match the gold.

Data	Baseline Accuracy	System Accuracy
Dev	65.7%	82.5%
Test	76.8%	83.8%

Table 8: Baseline vs. System Accuracy

Tag	Gold Errors		System Errors		Typos
	Not Tagged	Over generated	Not Tagged	Over generated	
Punct	100.0	0.0	0.0	0.0	0.0
Sound	79.3	10.3	10.3	0.0	0.0
Foreign	47.2	1.9	12.3	20.3	18.4
Name	26.3	13.7	45.3	8.4	6.3

Table 9: Error Analysis of tag classification errors

As a baseline, we use the most frequent tag, which is *Arabic* in our case, along with the transliteration of the word using our black box system. Then we apply the above evaluation metric on both *Dev* and *Test*. The results are shown in table 8. The baseline accuracies on *Dev* and *Test* are 65.7% and 76.8% respectively. By considering the actual output of our system, the accuracy on the *Dev* and *Test* data increases to 82.5% and 83.8% respectively.

9.3 Error Analysis

We conducted an error analysis for tag classification on the development set. The analysis is done for the tags that we built models for, which are *Punct*, *Sound*, *Foreign* and *Name*.[2] Table 9 shows the different error types for classifying the tags. Tagging errors could be either gold errors or system errors. These errors could be either due to tag over-generation or because the correct tag is not detected. Additionally, there are typos in the input Arabizi that sometimes prevent the system from assigning the correct tags. Gold errors contribute to a large portion of the tagging errors, representing 100.0%, 89.6%, 49.1% and 40.0% for the *Punct*, *Sound*, *Foreign* and *Name* tags, respectively.

10 Conclusion and Future Work

We presented a system for automatic processing of Arabic social media text written in Roman script, or Arabizi. Our system not only transliterates the Arabizi text in the Egyptian Arabic dialect but also classifies input Arabizi tokens as sounds, punctuation marks, names, foreign words, or Arabic words, and detects emoticons. We define a new task-specific metric for evaluating the complete system. Our best setting achieves an overall performance accuracy of 83.8% on a blind test set.

In the future, we plan to extend our work to other Arabic dialects and other language contexts such as Judeo-Arabic (Arabic written in Hebrew script with code switching between Arabic and Hebrew). We plan to explore the use of this component in the context of specific applications such as machine translation from Arabizi Arabic to English, and sentiment analysis in social media. We also plan to make the system public so it can be used by other people working on Arabic NLP tasks related to Arabizi.

[2] As mentioned in Section 4, the *Arabic* tag is assigned to any remaining untagged words after running the classification models.

Acknowledgement

This paper is based upon work supported by DARPA Contract No. HR0011-12-C-0014. Any opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of DARPA. Nizar Habash performed most of his contribution to this paper while he was at the Center for Computational Learning Systems at Columbia University.

References

Mohamed Al-Badrashiny, Ramy Eskander, Nizar Habash, and Owen Rambow. 2014. Automatic Transliteration of Romanized Dialectal Arabic. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 30–38, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Ann Bies, Zhiyi Song, Mohamed Maamouri, Stephen Grimes, Haejoong Lee, Jonathan Wright, Stephanie Strassel, Nizar Habash, Ramy Eskander, and Owen Rabmow. 2014. Transliteration of Arabizi into Arabic Orthography: Developing a Parallel Annotated Arabizi-Arabic Script SMS/Chat Corpus. In *Arabic Natural Language Processing Workshop, EMNLP*, Doha, Qatar.

Tim Buckwalter. 2004. Buckwalter Arabic Morphological Analyzer Version 2.0. LDC catalog number LDC2004L02, ISBN 1-58563-324-0.

Achraf Chalabi and Hany Gerges. 2012. Romanized Arabic Transliteration. In *Proceedings of the Second Workshop on Advances in Text Input Methods (WTIM 2012)*.

Eleanor Clark and Kenji Araki. 2011. Text normalization in social media: Progress, problems and applications for a pre-processing system of casual english. *Procedia - Social and Behavioral Sciences*, 27(0):2 – 11. Computational Linguistics and Related Fields.

Kareem Darwish, Walid Magdy, and Ahmed Mourad. 2012. Language Processing for Arabic Microblog Retrieval. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 2427–2430, New York, NY, USA. ACM.

Kareem Darwish. 2013. Arabizi Detection and Conversion to Arabic. *CoRR*.

Kareem Darwish. 2014. Arabizi Detection and Conversion to Arabic . In *Arabic Natural Language Processing Workshop, EMNLP*, Doha, Qatar.

Leon Derczynski, Alan Ritter, Sam Clark, and Kalina Bontcheva. 2013. Twitter part-of-speech tagging for all: Overcoming sparse and noisy data. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, pages 198–206, Hissar, Bulgaria, September. INCOMA Ltd. Shoumen, BULGARIA.

Mona Diab, Mohamed Al-Badrashiny, Maryam Aminian, Mohammed Attia, Pradeep Dasigi, Heba Elfardy, Ramy Eskander, Nizar Habash, Abdelati Hawwari, and Wael Salloum. 2014. Tharwa: A Large Scale Dialectal Arabic - Standard Arabic - English Lexicon. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Reykjavik, Iceland.

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.

Heba Elfardy, Mohamed Al-Badrashiny, and Mona Diab. 2014. AIDA: Identifying Code Switching in Informal Arabic Text. In *Workshop on Computational Approaches to Linguistic Code Switching, EMNLP*, Doha, Qatar, October.

Ramy Eskander, Nizar Habash, Owen Rambow, and Nadi Tomeh. 2013. Processing Spontaneous Orthography. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Atlanta, GA.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 42–47, Stroudsburg, PA, USA. Association for Computational Linguistics.

Stephan Gouws, Donald Metzler, Congxing Cai, and Eduard Hovy. 2011. Contextual bearing on linguistic variation in social media. In *Proceedings of the Workshop on Languages in Social Media*, LSM '11, pages 20–29, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nizar Habash, Abdelhadi Soudi, and Tim Buckwalter. 2007. On Arabic Transliteration. In A. van den Bosch and A. Soudi, editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Springer.

Nizar Habash, Mona Diab, and Owen Rabmow. 2012a. Conventional Orthography for Dialectal Arabic. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Istanbul.

Nizar Habash, Ramy Eskander, and Abdelati Hawwari. 2012b. A Morphological Analyzer for Egyptian Arabic. In *Proceedings of the Twelfth Meeting of the Special Interest Group on Computational Morphology and Phonology*, pages 1–9, Montréal, Canada.

Nizar Habash. 2009. REMOOV: A tool for online handling of out-of-vocabulary words in machine translation. In Khalid Choukri and Bente Maegaard, editors, *Proceedings of the Second International Conference on Arabic Language Resources and Tools*. The MEDAR Consortium, April.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18.

LDC. 2014a. BOLT Phase 2 SMS and Chat Arabic DevTest Data – Source Annotation, Transliteration and Translation. LDC catalog number LDC2014E28.

LDC. 2014b. BOLT Phase 2 SMS and Chat Arabic Training Data – Source Annotation, Transliteration and Translation R1. LDC catalog number LDC2014E48.

LDC. 2014c. BOLT Program: Romanized Arabic (Arabizi) to Arabic Transliteration and Normalization Guidelines. Version 3. Linguistic Data Consortium.

Marco Lui, Jey Han Lau, and Timothy Baldwin. 2014. Automatic detection and language identification of multilingual documents. In *Proceedings of LREC*.

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

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1524–1534, Stroudsburg, PA, USA. Association for Computational Linguistics.

Clare Voss, Stephen Tratz, Jamal Laoudi, and Douglas Briesch. 2014. Finding Romanized Arabic Dialect in Code-Mixed Tweets. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios

Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, may. European Language Resources Association (ELRA).

Omar F Zaidan and Chris Callison-Burch. 2011. The Arabic online commentary dataset: an annotated dataset of informal Arabic with high dialectal content. In *Proceedings of ACL*, pages 37–41.

Ines Zribi, Rahma Boujelbane, Abir Masmoudi, Mariem Ellouze, Lamia Belguith, and Nizar Habash. 2014. A Conventional Orthography for Tunisian Arabic. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Reykjavik, Iceland.