Identification of Languages in Algerian Arabic Multilingual Documents

Wafia Adouane and Simon Dobnik
CLASP, Department of FLoV
University of Gothenburg, Sweden
{wafia.adouane,simon.dobnik}@gu.se

Abstract

This paper presents a language identification system designed to detect the language of each word, in its context, in a multilingual documents as generated in social media by bilingual/multilingual communities, in our case speakers of Algerian Arabic. We frame the task as a sequence tagging problem and use supervised machine learning with standard methods like HMM and Ngram classification tagging. We also experiment with a lexicon-based method. Combining all the methods in a fall-back mechanism and introducing some linguistic rules, to deal with unseen tokens and ambiguous words, gives an overall accuracy of 93.14%. Finally, we introduced rules for language identification from sequences of recognised words.

1 Introduction

Most of the current Natural Language Processing (NLP) tools deal with one language, assuming that all documents are monolingual. Nevertheless, there are many cases where more than one language is used in the same document. The present study seeks to fill in some of the needs to accommodate multilingual (including bilingual) documents in NLP tools. The phenomenon of using more than one language is common in multilingual societies where the contact between different languages has resulted in various language (code) mixing like code-switching and borrowings. Code-switching is commonly defined as the use of two or more languages/language varieties with fluency in one conversation, or in a sentence, or even in a single word. Whereas borrowing is used to refer to the altering of words from one language into another.

There is no clear-cut distinction between borrowings and code-switching, and scholars have different views and arguments. We based our work on (Poplack and Meechan, 1998) where the authors consider borrowing as the adaptation of lexical items, with a phonological and morphological integration, from one language to another. Otherwise, it is a code-switching, at single lexical item, phrasal or clausal levels, either the lexical item/phrase/clause exists or not in the first language.[1] We will use "language mixing" as a general term to refer to both code-switching and borrowing.

We frame the task of identifying language mixing as a segmentation of a document/text into sequences of words belonging to one language, i.e. segment identification or chunking based on the language of each word. Since language shifts can occur frequently at each point of a document we base our work on the isolated word assumption as referred to by (Singh and Gorla, 2007) wherein the authors consider that it is more realistic to assume that every word in a document can be in a different language rather than a long sequence of words being in the same language. However, we are also interested in identifying the boundaries of each language use, sequences of words belonging to the same language, which we address by adding rules for language chunking.

This paper's main focus is the detection of language mixing in Algerian Arabic texts, written in Arabic script, used in social media while its contribution is to provide a system that is able to detect the language of each word in its context. The paper is organized as follows: in Section 2, we give a brief overview of Algerian Arabic which is

[1] Refers to the first language the speakers/users use as their mother tongue.

Proceedings of The Third Arabic Natural Language Processing Workshop (WANLP), pages 1–8,
Valencia, Spain, April 3, 2017.

a well suited, and less studied, language for detecting language mixing. In Section 3, we present our newly built linguistic resources, from scratch, and we motivate our choices in annotating the data. In Section 4, we describe the different methods used to build our system. In Section 5, we survey some related work, and we conclude with the main findings and some of our future directions.

2 Algerian Arabic

Algerian Arabic is a group of North African Arabic dialects mixed with different languages spoken in Algeria. The language contact between many languages, throughout the history of the region, has resulted in a rich complex language comprising words, expressions, and linguistic structures from various Arabic dialects, different Berber varieties, French, Italian, Spanish, Turkish as well as other Mediterranean Romance languages. Modern Algerian Arabic is typically a mixture of Algerian Arabic dialects, Berber varieties, French, Classical Arabic, Modern Standard Arabic, and a few other languages like English. As it is the case with all North African languages, Algerian Arabic is heavily influenced by French where code-switching and borrowing at different levels could be found.

Algerian Arabic is different from Modern Standard Arabic (MSA) mainly phonologically and morphologically. For instance, some sounds in MSA are not used in Algerian Arabic, namely the interdental fricatives 'ث' /θ/, 'ذ' /ð/ and the glottal fricative 'ه' /h/ at a word final position. Instead they are pronounced as aspirated stop 'ت' /t/, dental stop 'د' /d/ and bilabial glide 'و' /w/ respectively. Hence, the MSA word ذهب /*hb/ "gold" is pronounced/written as 'دهب' /dhb/ in Algerian Arabic. Souag (2000) gives a detailed description of the characteristics of Algerian Arabic and describes at length how it differs from MSA. Compared to the rest of Arabic varieties, Algerian Arabic is different in many aspects (vocabulary, pronunciation, syntax, etc.). Maybe the main common characteristics between them is the use on non-standard orthography where people write according to their pronunciation.

3 Corpus and Lexicons

In this section, we describe how we collected and annotated our corpus and explain the motivation behind some annotation decisions. We then describe how we build lexicons for each language and provide some statistics about each lexicon.

3.1 Corpus

We automatically collected content from various social media platforms that we knew they use Algerian Arabic. We included texts of various topics, structures and lengths. In total, we collected 10,597 documents. On this corpus we ran an automatic language identifier which is trained to distinguish between the most popular Arabic varieties (Adouane et al., 2016). Afterwards, we only consider the documents that were identified as Algerian Arabic which gives us 10,586 documents (215,843 tokens).[2] For robustness, we further preprocessed the data where we removed punctuation, emoticons and diacritics, and then we normalized it. In social media users do not use punctuation and diacritics/short vowels in a consistent way, even within the same text. We opt for such normalization because we assume that such idiosyncratic variation will not affect language identification.

Based on our knowledge of Algerian Arabic and our goal to distinguish between borrowing and code-switching at a single lexical item, we decided to classify words into six languages: Algerian Arabic (ALG), modern standard Arabic (MSA), French (FRC), Berber (BER)[3], English (ENG) and Borrowings (BOR) which includes foreign words adapted to the Algerian Arabic morphology. Moreover, we grouped all Named Entities in one class (NER), sounds and interjections in another (SND). Our choice is motivated by the fact that these words are language independent. We also keep digits to keep the context of words and grouped them in a class called DIG.

In total, we have nine separate classes. First, three native speakers of Algerian Arabic annotated the first 1,000 documents (22,067 words) from the pre-processed corpus, following a set of annotation guidelines which takes into account the above-mentioned linguistic differences between

[2]We use *token* to refer to lexical words, sounds and digits (excluding punctuation and emoticons) and *word* to refer only to lexical words.

[3]Berber is an Afro-Asiatic language used in North Africa and which is not related to Arabic.

Algerian Arabic and Modern Standard Arabic. To assess the quality of the data annotation, we computed the inter-annotator agreement using the Cohen's kappa coefficient (κ), a standard metric used to evaluate the quality of a set of annotations in classification tasks by assessing the annotators' agreement (Carletta, 1996). The κ on the human annotated 1,000 documents is 89.27%, which can be qualitatively interpreted as "really good".

Next, we implemented a tagger based on Hidden Markov Models (HMM) and the Viterbi algorithm, to find the best sequence of language tags over a sequence of words. The assumption is that the context of the surrounding words and their language tags will predict the language for the current word. We apply smoothing – we assign an equal low probability (estimated from the training data) for unseen words – during training to estimate the emission probability and compute the transmission probabilities. We trained the HMM tagger on the human annotated 1,000 documents. We divided the remaining corpus (non-annotated data) into 9 parts (each part from 1-8 includes 1,000 documents and the last part includes 1,586 documents). We first used the trained tagger to automatically annotate the first part, then manually checked/corrected the annotation. After that, we added the checked annotated part to the already existing training dataset and used that to annotate the following part. We performed the same bootstrapping process until we annotated all the parts.

The gradual bootstrapping annotation of new parts of the corpus helped us in two ways. First, it speeded up the annotation process which took five weeks for three human annotators to check and correct the annotations in the entire corpus compiled so far. It would take them far longer if they started annotation without the help of the HMM tagger. Second, checking and correcting the annotation of the automatic tagger served us to analyse the errors the tagger was making. The final result is a large annotated corpus with a human annotation quality which is an essential element for learning useful language models. Table 1 shows some statistics about the current annotated corpus.

Category	ALG	MSA	FRC	BOR	NER	ENG	BER	DIG	SND
# Words	118,942	82,114	6,045	4,025	2,283	254	99	1,394	687

Table 1: Statistics about the annotated corpus.

3.2 Lexicons

We asked two other Algerian Arabic native speakers to collect words for each included language from the web excluding the platforms used to build the above-described corpus. We cleaned the newly compiled word lists and kept only one occurrence for each word, and we removed all ambiguous words: words that occur in more than one language. Table 2 gives some statistics about the final lexicons that are lists of words that unambiguously occur in a given language, one word per line in a `.txt` file. Effectively, we see the role of dictionaries as stores for exceptions, while for ambiguous words we work towards a disambiguation mechanism.

Category	ALG	MSA	FRC	BOR	NER	ENG	BER
# Types	42,788	94,167	3,206	2,751	1,945	157	21,789

Table 2: Statistics about the lexicons.

4 Experiments and Results

In this section, we describe the methods and the different experimental setups we used to build our language identification tool. We analyze and discuss the obtained results. We start identifying language at a word level and then we combine words to identify the language of sequences. We approach the language identification at the word level by taking into account the context of these words. We supplement the method with a lexicon lookup approach and manually constructed rules.

To evaluate the performance of the system, we divided the final human annotated dataset into two parts: the training dataset which contains 10,008 documents (215,832 tokens) and the evaluation dataset which contains 578 documents (10,107 tokens). None of the documents included in the evaluation dataset were used to compile the lexicons previously described.

4.1 Identifying words

4.1.1 HMM Tagger

In Section 3.1 we describe an implementation of a tagger based on Hidden Markov Models (HMM) used as a helping tool to bootstrap data annotation. Now, having an annotated corpus we are interested in the performance of the tagger on our final fully annotated corpus which we discuss here. We train the HMM tagger on the training data and evaluate

it on the evaluation data. Table 3 shows the performance of the tagger.

Category	Precision (%)	Recall (%)	F-score (%)
ALG	87.10	89.96	88.50
BER	100	18.18	30.77
BOR	97.71	40.38	57.14
DIG	100	94.74	97.30
ENG	100	24.14	38.89
FRC	82.28	63.87	71.92
MSA	84.03	88.04	85.99
NER	84.07	61.69	71.16
SND	100	85.71	92.31

Table 3: Performance of the HMM tagger.

The overall accuracy of the tagger is 85.88%. This quite high performance gives an idea about how useful and helpful was the use of the HMM tagger to annotate the data before the human checking. The tagger also outperforms the majority baseline (#majority class / #total tokens) which is 55.10%. From Table 3 we see that the HMM tagger is good at identifying ALG and MSA words, given an F-score of 88.50% and 85.99% respectively.[4] However, this performance dropped with other categories, it is even lower than the majority baseline for BER and ENG.

The confusion matrix of the tagger (omitted here due to space constraints) shows that all categories are confused either with ALG or MSA. This can be explained by the fact that ALG and MSA are the majority classes which means that both emission and transmission probabilities are biased to these two categories. The analysis of the most frequent errors shows that errors can be grouped into two types. The first type includes ambiguous words. For example, in the sentence الماتش مشري حارس خلا البيت يدخل
/AlmAt$ m$ry HArs xlA Albyt ydxl/
"the football match is bought, the goal keeper allowed the (goal) ball to enter", the word 'البيت' is "the goal" in French, the same word means "the house" in MSA and "the room" in ALG. Also the following word 'يدخل' which means " to enter" is used with all the possible meanings of 'البيت' (enter a house/ a room and ball enters). The second type of errors relates to unseen words in the training data. Because of the smoothing we used, the HMM tagger does not return 'unseen word'. Instead, another tag is assigned, mostly ALG and MSA. We could identify such words by setting manually some thresholds, but it is not clear what these should be.

The Precision is high for all unambiguous tokens, however the Recall is very low. To overcome the limitation of the HMM tagger in dealing with unseen words, we decided to explore other methods. Moreover, we want to reduce the uncertainty of our tagger deciding what is an unseen word. We found it difficult to set any threshold that is not data-dependent. Therefore, we introduced a new category called unknown UNK which is inspired from *active learning* (Settles, 2009). We believe that this should be used in all automatic systems instead of returning a simple guess based on its training model.

4.1.2 Lexicon-based Tagger

We devised a simple algorithm that performs a lexicon look-up and returns for each word the language of the lexicon it appears in (note that lexicons contain only unambiguous words). For SND, we created a list of most common sounds like 'بفف' "pff", 'هه' "hh". For digits, we used the `isdigit` method built-in Python. In the case where a word does not appear in any lexicon, the unknown UNK category is returned. This method does not require training, but it requires good quality lexicons with a wide coverage. We evaluated the lexicon-based tagger on the same evaluation dataset and the results are shown in Table 4.

Category	Precision (%)	Recall (%)	F-score (%)
ALG	97.39	81.55	88.77
BER	100	63.64	77.78
BOR	98.52	83.91	90.63
DIG	100	100	100
ENG	100	55.17	71.11
FRC	96.30	84.85	90.21
MSA	97.69	82.43	89.42
NER	97.46	74.68	84.56
SND	100	100	100

Table 4: Performance of the lexicon tagger.

The overall accuracy of the tagger is 81.98%. From comparing the results shown in Table 4 and

[4]We ignore the DIG and SND categories because we are interested in lexical words. As explained above, we kept them to keep the context of each word.

Table 3, it is clear that the Recall has increased for all categories except for ALG and MSA. The reason is that now we have the UNK category where among the 10,107 tokens used for evaluation, 1,610 words are tagged as UNK instead of ALG or MSA. We examined the UNK words and found that these words do not exist in the lexicons. Either they are completely new words or they are different spellings of already covered words (which count as different words).

The confusion matrix of the lexicon-based tagger (omitted here) shows that the most frequent errors are between all categories and the UNK category. The tagger often confuses between ALG/MSA and MSA/ALG. It also occasionally confuses between ALG/FRC and ALG/NER. These errors could be explained by the fact that the context of a word is ignored.
For example, in the sentence
حطولنا بلا بقلاوا بلا ميقطعوه حرنا كيفاش نكلوه
/HTwlnA blA bqlAwA blA myqTEwh HrnA kyfA$ nklwh/ "they served us a dish of Baklava without cutting it, we did not know how to eat it", the first "بلا" means "dish" in French and the second "بلا" means "without" in MSA. In the sentence
وجدنا كلشي بلقيس لي قالولنا عليه
/wjdnA kl$y blqys ly qAlwlnA Elyh/ "we prepared everything according to the measures they (gave) told us", the word "بلقيس" means "with the measure" in ALG and it is a female name (NER). Analysing the tagging errors indicates that using lexicon-based tagger is not effective in dealing with ambiguous words because it ignores the context of words, and as known, the context is the main means of ambiguity resolution.

4.1.3 n-gram Tagger

Our goal is to build a language tagger, at a word level, which takes into account the context of each word in order to be able to properly deal with ambiguous words. At the same time, we want it to be able to deal with unseen words. Ideally we want it to return UNK for each word it did not see before. This is because we want to analyse the words the tagger is not able to identify and appropriately update our dictionaries.

The Natural Language Toolkit (NLTK) n-gram POS tagger (Steven et al., 2009) is well suited for further experimentation. First, the tagging principle is the same and the only difference is the set of tags. Secondly, the NLTK Ngram tagger offers the possibility of changing the context of a word up to trigrams as well as the possibility of combining taggers (unigram, bigram, trigram) with the back-off option. It is also possible to select a single category, for example the most frequent tag or UNK, as a default tag in case all other options fail. This combination of different taggers and the back-off option leads to the optimization of the tagger performance. We start with the method involving most knowledge/context, if it fails we back off progressively to a simpler method. Table 5 summarizes the results of different configurations. We train and evaluate on the same training and evaluation sets as before.

Tagger	**Accuracy (%)**
Unigram	74.89
Bigram	12.27
Trigram	07.97
BackOff(Trigram, Bigram, Unigram, ALG)	87.12
BackOff(Trigram, Bigram, Unigram, UNK)	74.95
Default (ALG)	52.12

Table 5: Performance of different n-gram tagger configurations.

The use of bigram and trigram taggers alone has a very little effect because of the data sparsety. It is unlikely to find the same word sequences (bigram, trigram) several times. However, chaining the taggers has a positive effect on the overall performance. Notice also that tagging words with the majority class ALG performs less than the majority baseline, 52.12% compared to 55.10%. In Table 6, we show the performance of the BackOff(Trigram, Bigram, Unigram, UNK) tagger in detail.

Category	**Precision (%)**	**Recall (%)**	**F-score (%)**
ALG	96.17	75.27	84.44
BER	100	27.27	42.86
BOR	99.24	41.01	58.04
DIG	100	94.74	97.30
ENG	100	20.69	34.29
FRC	97.38	60.61	74.71
MSA	97.45	79.48	87.55
NER	94.69	69.48	80.15
SND	100	85.71	92.31

Table 6: Performance of the BackOff(Trigram, Bigram, Unigram, UNK) tagger.

Compared to the previous tagger, this tagger suffers mainly from the unseen words where 2,279 tokens were tagged as UNK. This could account for the low Recall obtained for all categories. There is also some confusion between MSA/ALG, ALG/MSA and FRC/ALG.

4.1.4 Combining n-gram taggers and lexicons

The unknown words predicted by the BackOff(Trigram, Bigram, Unigram, UNK) tagger can be replaced with words from our dictionaries. First, we run the BackOff(Trigram, Bigram, Unigram, UNK), and then we run the lexicon-based tagger to catch some of the UNK tokens. Table 7 summarizes the results.

Category	Precision (%)	Recall (%)	F-score (%)
ALG	96.47	92.88	94.64
BER	100	81.82	90.00
BOR	99.28	86.44	92.41
DIG	100	100	100
ENG	100	90.91	95.24
FRC	98.95	88.08	93.20
MSA	98.42	93.64	95.97
NER	96.05	94.81	95.42
SND	100	100	100

Table 7: Performance of the tagger combining n-gram and lexicons.

Combining information from the training data and the lexicons increases the performance of the language tagging for all categories, giving an overall accuracy of 92.86%. Still there are errors that are mainly caused by unseen and ambiguous words. Based on the confusion matrix of this tagger (omitted here) the errors affect the same language pairs as before.

All language tags are missing words that are tagged as UNK words (in total 476 words). We found that these words are neither seen in the training data nor covered by any existing lexicons new words or different (even as spelling variants of the existing words). Keeping track of the unseen words, by assigning them the UNK tag, allows us to extend the lexicons to ensure a wider coverage.

To test how data-dependent is our system, we cross-validated it, and all the accuracies were close to the reported overall accuracy of the system, combining n-grams and lexicons, evaluated on the evaluation data.

4.1.5 Adding rules

We analysed the lexicons and manually extracted some features that would help us identify the language, for instance the starting and the final sequence of characters of a word. The application of these rules improved the performance of the system, given an overall accuracy of 93.14%, by catching some unseen vocabulary (the number of UNK dropped to 446). As shown in Table 8, this hybrid tagger is still unable to deal with unseen words in addition to confusing some language pairs due to lexical ambiguity.

		Misclassified languages									
		ALG	BER	BOR	DIG	ENG	FRC	MSA	NER	SND	UNK
Correct languages	ALG	4912	0	0	0	0	4	56	1	0	295
	BER	1	9	0	0	0	0	0	0	0	1
	BOR	1	0	280	0	0	5	1	0	0	30
	DIG	0	0	0	38	0	0	0	0	0	0
	ENG	1	0	0	0	10	0	0	0	0	0
	FRC	28	0	0	0	0	384	0	0	0	16
	MSA	134	0	1	0	0	1	3612	5	0	101
	NER	6	0	2	0	0	0	8	135	0	3
	SND	0	0	0	0	0	0	0	0	7	0

Table 8: Confusion matrix of the Hybrid Tagger.

4.2 Identifying sequences of words

Now that we have a model that predicts the category of each token in a text, we added rules to label also non-linguistic words (punctuation (PUN) and emoticons (EMO)). This helps us to keep the original texts as produced by users as well as PUN and EMO be might be useful for other NLP tasks like sentiment and opinion analysis. Based on this extended annotation, we designed rules to identify the language of a specific segment of a text. The output of the system is a chunked text (regardless of its length) identifying language boundaries. It is up to the user how to chunk language independent categories, i.e. NER, DIG and SND, either separately or include them in larger segments based on a set of rules. For instance, the sentence

واش نذير يا نااااس راني توجور روطاار الرفاي
نتاعي مينوضنيش مام نريقليه ماشي لا فوط نتاعي 😂😁

/wA$ ndyr yA nAs rAny twjwr rwtAr AlrAfAy ntAEy mynwDny$/ mAm nryqlyh mA$y lA fwT ntAEy/ “ what should I do people, I am always late my alarm clock does not wake me up even I set it , it is not my fault” is chunked as follows:

[واش نديرALG] [يا ناسMSA] [رانيALG] [توجور روطارFRC]
[الرفايBOR] [نتاعي مينوضنيشALG] [مامFRC] [نريقليهBOR]
[ماشيALG] [لا فوطFRC] [نتاعيALG] [😂😂EMO]

Chunking text segments based on the language is entirely based on the identification of the language of each word in the segment. One of the open questions is what to do when words tagged as UNK are encountered. We still do not have a good way to deal with this situation, so we leave them as separate chunks UNK. Extending the training dataset and the coverage of the current lexicons would help to solve the problem.

5 Related Work

There is an increasing need to accommodate multilingual documents in different NLP tasks. Most work focuses on detecting different language pairs in multilingual texts, among others, Dutch-Turkish (Nguyen and Doğruöz, 2013), English-Bengali and English-Hindi (Das and Gambäck, 2013), English-French (Carpuat, 2014), Swahili-English (Piergallini et al., 2016). Since 2014, a Shared Task on Language Identification in Code-Switched Data is also organized (Solorio et al., 2014).

Detecting language mixing in Arabic social media texts has also attracted the attention of the research community. (Elfardy et al., 2013) propose an automatic system to identify linguistic code switch points between MSA and dialectal Arabic (Egyptian). The authors use a morphological analyser to decide whether a word is in MSA or DA, and they compare the performance of the system to the previous one (Elfardy and Diab, 2012) where they used unsupervised approach based on lexicons, sound-change rules, and language models. There is also work on detecting language mixing in Moroccan Arabic (Samih and Maier, 2016). In contrast to the previous work on Arabic, our annotation scheme and the system make a distinction between code-switching and borrowing which they do not consider. We also detect words in their contexts and do not group them in a Mixed category. To the best of our knowledge, we are not aware of any similar system which identifies language mixing in Algerian Arabic documents.

6 Conclusions and Future Work

We have presented a system for identifying the language at word and long sequence levels in multilingual documents in Algerian Arabic. We described the data and the different methods used to train the system that is able to identify language of words in their context between Algerian Arabic, Berber, English, French, Modern Standard Arabic and mixed languages (borrowings). The system achieves a very good performance, with an overall accuracy of 93.14% against a baseline of the majority class of 55.10%.

We discussed the limitations of the current system and gave insights on how to overcome them. The system is also able to identify language boundaries, i.e. sequence of tokens, including digits, sounds, punctuation and emoticons, belonging to the same language/category. Moreover, it performs also well in identifying Named Entities. Our system trained on a multilingual data from multiple domains handles several tasks, namely context sensitive language identification at a word level (borrowing or code-switching), language identification at long sequence level (chunking) and Named Entity recognition.

In the future, we plan to evaluate the automatic lexicon extension, as well as use the system in tasks such as error correction, Named Entity categorization(Person, Location, Product, Company), topic identification, sentiment analysis and textual entailment. We are currently extending our corpus and annotating it with other linguistic information.

References

Wafia Adouane, Nasredine Semmar, Richard Johansson, and Victoria Bobicev. 2016. Automatic detection of Arabicized Berber and Arabic varieties. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 63–72.

Jean Carletta. 1996. Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 2(22):249–254.

Marine Carpuat. 2014. Mixed-language and code-switching in the canadian hansard. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pages 107–115.

Amitava Das and Björn Gambäck. 2013. Code-mixing in social media text: The last language identification frontier? *Traitement Automatique des Langues*, 54(3):41–64.

Heba Elfardy and Mona Diab. 2012. Token level identification of linguistic code switching. In *COLING*, pages 287–296.

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)*.

Dong Nguyen and A. Seza Doğruöz. 2013. Word level language identification in online multilingual communication. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 18–21.

Mario Piergallini, Rouzbeh Shirvani, Gauri S. Gautam, and Mohamed Chouikha. 2016. Word-level language identification and predicting code-switching points in swahili-english language data. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 21–29.

Shana Poplack and Marjory Meechan. 1998. How languages fit together in codemixing. *The International Journal of Bilingualism*, 2(2):127–138.

Younes Samih and Wolfgang Maier. 2016. Detecting code-switching in moroccan arabic. In *Proceedings of SocialNLP @ IJCAI-2016*.

Burr Settles. 2009. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

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*.

Thamar Solorio, Elizabeth Blair, Suraj Maharjan, Steven Bethard, Mona Diab, Mahmoud Ghoneim, Abdelati Hawwari, Fahad AlGhamdi, Julia Hirschberg, 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*, pages 62–72.

Lameen Souag. 2000. *Grammar of Algerian Darja*. Retrieved from `https://goo.gl/p2EmVH`.

Bird Steven, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media.