A comparison of character neural language model and bootstrapping for language identification in multilingual noisy texts

Wafia Adouane, Simon Dobnik, Jean-Philippe Bernardy, Nasredine Semmar

A Comparison of Character Neural Language Model and Bootstrapping for Language Identification in Multilingual Noisy Texts

Wafia Adouane[1], Simon Dobnik[1], Jean-Philippe Bernardy[1], and Nasredine Semmar[2]
[1]Department of Philosophy, Linguistics and Theory of Science (FLoV),
Centre for Linguistic Theory and Studies in Probability (CLASP), University of Gothenburg
[2]CEA, LIST, Vision and Content Engineering Laboratory Gif-sur-Yvette, France
{wafia.adouane,simon.dobnik,jean-philippe.bernardy}@gu.se
nasredine.semmar@cea.fr

Abstract

This paper seeks to examine the effect of including background knowledge in the form of character pre-trained neural language model (LM), and data bootstrapping to overcome the problem of unbalanced limited resources. As a test, we explore the task of language identification in mixed-language short non-edited texts with an under-resourced language, namely the case of Algerian Arabic for which both labelled and unlabelled data are limited. We compare the performance of two traditional machine learning methods and a deep neural networks (DNNs) model. The results show that overall DNNs perform better on labelled data for the majority categories and struggle with the minority ones. While the effect of the untokenised and unlabelled data encoded as LM differs for each category, bootstrapping, however, improves the performance of all systems and all categories. These methods are language independent and could be generalised to other under-resourced languages for which a small labelled data and a larger unlabelled data are available.

1 Introduction

Most Natural Language Processing (NLP) tools are generally designed to deal with monolingual texts with more or less standardised spelling. However, users in social media, especially in multilingual societies, generate multilingual non-edited material where at least two languages or language varieties are used. This phenomenon is linguistically referred to as language (code) mixing where code-switching and borrowing, among others, are the most studied phenomena. Poplack and Meechan (1998) defined borrowing as a morphological or a phonological adaptation of a word from one language to another and code-switching as the use of a foreign word, as it is in its original language, to express something in another language. However, the literature does not make it clear whether the use of different script is counted as borrowing, or code-switching or something else. For instance, there is no linguistic well-motivated theory about how to classify languages written in other scripts, like French written in Arabic script which is frequently the case in North Africa. This theoretical gap could be explained by the fact that this fairly recent phenomenon has emerged with the widespread of the new technologies. In this paper, we consider both code-switching and borrowing and refer to them collectively as language mixing. Our motivation in doing so is to offer to sociolinguists a linguistically informative tool to analyse and study the language contact behaviour in the included languages.

The task of identifying languages in mixed-language texts is a useful pre-processing tool where sequences belonging to different languages/varieties are identified. They are then processed by further language/variety-specific tools and models. This task itself has neither been well studied for situations when many languages are mixed nor has it been explored as a main or an auxiliary task in multi-task learning (see Section 2).

1.1 Related Work

There has been some interesting work in detecting code mixing for a couple of languages/language varieties, mostly using traditional sequence labelling algorithms like Conditional Random Field (CRF), Hidden Markov Model (HMM), linear kernel Support Vector Machines (SVMs) and a combination of different methods and linguistic resources (Elfardy and Diab, 2012; Elfardy et al., 2013; Barman et al., 2014b,a; Diab et al., 2016; Samih and Maier, 2016; Adouane and Dobnik, 2017). Prior work that is most closely related to our work using neural networks and related languages, Samih et al. (2016) used supervised

Proceedings of the Second Workshop on Subword/Character LEvel Models, pages 22–31
New Orleans, Louisiana, June 6, 2018.

deep neural networks (LSTM) and a CRF classifier on the top of it to detect code-switching, using small datasets of tweets, between Egyptian Arabic and MSA and between Spanish and English using pre-trained word embeddings trained on larger datasets. However, in their annotation they combined ambiguous words, which are words that could be of either languages depending on the context, in one category called 'ambiguous' and ignored words from minority languages. Moreover, the system was evaluated on a dataset with no instances of neither 'ambiguous' nor 'mixed-language' words, basically distinguishing between MSA and Egyptian Arabic words in addition to Named Entities and other non-linguistic tokens like punctuation, etc.

Similar to our work, Kocmi and Bojar (2017) proposed a supervised bidirectional LSTM model. However, the data used to train the model was created by mixing edited texts, at a line level, in 131 languages written in different scripts to create a multilingual data, making it a very different task from the one investigated here. We use non-edited texts, a realistic data as generated by users reflecting the real use of the included languages which are all written in the same Arabic script. Our texts are shorter and the size of the dataset is smaller, therefore, our task is more challenging.

By comparison to our work, most of the literature focuses on detecting code-switching points in a text, either at a token level or at a phrase level or even beyond a sentence boundaries, we distinguish between borrowing and code-switching at a word level by assigning all borrowed words to a separate variety (BOR). Most importantly, our main focus is to investigate ways to inject extra knowledge to take advantage of the unlabelled data.

1.2 Linguistic Situation in Algeria

The linguistic landscape in Algeria consists of several languages which are used in different social and geographic contexts to different degrees (Adouane et al., 2016a): local Arabic varieties (ALG), Modern Standard Arabic (MSA) which is the only standardised Arabic variety, Berber which is an Afro-Asiatic language different from Arabic and widely spoken in North Africa, and other non-Arabic languages such as French, English, Spanish, Turkish, etc. A typical text consists of a mixture of these languages, and this mixture is often referred to, somewhat mistakenly as Algerian Arabic. In this paper, we use the term Algerian language to refer to a mixture of languages and language varieties spoken in Algeria, and the term Algerian variety (ALG) to refer to the local variety of Arabic, which is used alongside other languages such as, for example Berber (BER).

This work seeks to identify the language or language variety of each word within an Algerian language text. Algerian language is characterised by non-standardised spelling and spelling variations based on the phonetic transcription of many local variants. For instance, the Algerian sentence in (1), which is user generated, is a mixture of 3 languages (Arabic, French and Berber) and 2 Arabic varieties (MSA and ALG). Each word is coloured by its language in d., b. is an IPA transcription and c. is the human English translation. To illustrate the difficulty of the problem, we additionally show the (incorrect) translation proposed by Google translate e., where words in black are additional words not appearing in the original sentence.

(1) a. سيلتوبلي حل الطاقة و سكر لباب موراك

b. [muræk ælbæb sekkær wu ætʕaqæ ħæl si:ltupli:]

c. Please open the window and close the door behind you

d. French Algerian Berber MSA Berber MSA Algerian

e. SELTOPLEY POWER SOLUTION AND SUGAR FOR MORAK PAPER

All the words in different languages are normally written in the Arabic script, which causes high degree of lexical ambiguity and therefore even if we had dictionaries (only available for MSA) it would be hard to disambiguate word senses this way. In (1), the ALG word حل *open* means *solution* in MSA, the Berber word الطاقة *window* which is adapted to the MSA morphology by adding the MSA definite article ال (case of borrowing) means *energy/capacity* in MSA. The Berber word سكر

close means *sugar / sweeten / liquor / get drunk* in MSA.

Moreover, the rich morphology of Arabic is challenging because it is a fusional language where suffixes and other morphemes are added to the base word, and a single morpheme denotes multiple aspects and features. Algerian Arabic shares many linguistic features with MSA, but it differs from it mainly phonologically, morphologically and lexically. For instance, a verb in the first person singular in ALG is the same as the first person plural in MSA. The absence of a morphological/syntactic analyser for ALG makes it challenging to correctly analyse an ALG text mixed with other languages and varieties.

Except for MSA, Arabic varieties are neither well-documented nor well-studied, and they are classified as under-resourced languages. Furthermore, social media are the only source of written texts for Algerian Arabic. The work in NLP on Algerian Arabic and other Arabic varieties also suffers severely from the lack of labelled (and even unlabelled) data that would allow any kind of supervised training. Another challenge is that we have to deal with all the complications present in social media domain, namely the use of short texts, spelling and word segmentation errors, etc. in addition to the non-standard orthography used in informal Arabic varieties. We see the task of identification of the variety of each word in a text a necessary first step towards developing more sophisticated NLP tools for this Arabic variety which is itself a mixture of other languages and varieties.

In this paper we explore two avenues for improving the state of the art in variety identification for Algerian Arabic. First, we measure the ability of recurrent neural networks to identify language mixing using only a limited training corpus. Second, we explore to what extent adding background knowledge in the form of pre-trained character-based language model and bootstrapping can be effective in dealing with under-resourced languages in the domain of language identification in mixed-language texts for which neither large labelled nor unlabelled datasets exist.

The paper is organized as follows: in Section 2, we give a brief overview of methods for levering learning from limited datasets. In Section 3, we describe the data. In Section 4, we present the architecture of our learning configurations which include both traditional approaches and deep neural networks and explain the training methods used on the labelled data, experiments and results. In Section 5, we experiment with these models when adding background knowledge and report the results.

2 Leveraging Limited Datasets

Deep learning has become the leading approach to solving linguistic tasks. However deep neural networks (DNNs) used in a supervised and unsupervised learning scenario usually require large datasets in order for the trained models to perform well. For example, Zhang et al. (2015) estimates that the size of the training dataset for character-level DNNs for text classification task should range from hundreds of thousands to several million of examples.

The limits imposed by the lack of labelled datasets have been countered by combining *structural learning* and *semi-supervised learning* (Ando and Zhang, 2005). Contrary to the supervised approach where a labelled dataset is used to train a model, in *structural learning*, the learner first learns underlying structures from either labelled or unlabelled data. If the model is trained on labelled data, it should be possible to reuse the knowledge encoded in the relations of the predictive features in this auxiliary task, if properly trained, to solve other related tasks. If the model is trained on unlabelled data, the model captures the underlying structures of words or characters in a language as a language model (LM), i.e., model the probabilistic distribution of words and characters of a text.

Such pre-trained LM should be useful for various supervised tasks assuming that linguistic structures are predictive of the labels used in these tasks. Approaches like this are known as *transfer learning* or *multi-task learning* (MTL) and are classified as a semi-supervised approaches (with no bootstrapping) (Zhou et al., 2004). There is an increasing interest in evaluating different frameworks (Ando and Zhang, 2005; Pan and Yang, 2010) and comparing neural network models (Cho et al., 2014; Yosinski et al., 2014). Some studies have shown that MTL is useful for certain tasks (Sutton et al., 2007) while others reported that it is not always effective (Alonso and Plank, 2017).

Bootstrapping (Nigam et al., 2000) is a gen-

eral and commonly used method of countering the limits of labelled datasets for learning. It is a semi-supervised method where a well-performing model is used to automatically label new data which is subsequently used as a training data for another model. This helps to enhance supervised learning. However, this is also not always effective. For example, Pierce and Cardie (2001) and Ando and Zhang (2005) show that bootstrapping degraded the performance of some classifiers.

3 Data

In this section, we describe the datasets that we use for training and testing our models. We use two datasets: small dataset, annotated with language labels, and a larger dataset lacking such annotation.

3.1 Labelled data

We use the human labelled corpus described by Adouane and Dobnik (2017) where each word is tagged with one of the following labels: ALG (Algerian), BER (Berber), BOR (Borrowing), ENG (English), FRC (French), MSA (Modern Standard Arabic), NER (Named Entity), SND (interjections/sounds) and DIG (digits). The annotators have access to the full context for each word. To the best of our knowledge, this corpus is the only available labelled dataset for code-switching and borrowing in Algerian Arabic, written in Arabic script, and in fact also one of the very few available datasets for this particular language variety overall. Because of the limited annotation resources the corpus is small, containing only 10,590 samples (each sample is a short text, for example one post in a social media platform). In total, the data contains 215,875 tokens distributed unbalancely as follows: 55.10% ALG (representing the majority category with 118,960 words), 38.04% MSA (82,121 words), 2.80% FRC (6,049 words), 1.87% BOR (4,044 words), 1.05% NER (2,283 words), 0.64% DIG (1,392 numbers), 0.32% SND (691 tokens), 0.10% ENG (236 words), and 0.04% BER (99 words).

3.2 Unlabelled data

Unfortunately, there is no existing user-generated unlabelled textual corpus for ALG. Therefore, we also collected, automatically and manually, new content from social media in Algerian Arabic which include social networking sites, blogs, microblogs, forums, community media sites and user reviews.[1]

The new raw corpus contains mainly short non-edited texts which require further processing before useful information can be extracted from them. We cleaned and pre-processed the corpus following the pre-processing and normalisation methods described by Adouane and Dobnik (2017). The data pre-processing and normalisation is based on applying certain linguistic rules, including: 1. Removal of non-linguistic words like punctuation and emoticons (indeed emoticons and inconsistent punctuation are abundant in social media texts.) 2. Reducing all adjacent repeated letters to maximum two occurrences of letters, based on the principle that MSA allows no more than two adjacent occurrences of the same letter. 3. Removal of diacritics representing short vowels, because these are rarely used; 4. Removal all duplicated instances of texts; 5. Removal of texts not mainly written in Arabic script 6. Normalisation all remaining characters to the Arabic script. Indeed, some users use related scripts like Persian, Pashto or Urdu characters, either because of their keyboard layout or to express some sounds which do not exist in the Arabic alphabet, e.g. /p/, /v/ and /g/.

Additionally, we feed each document, as a whole, to a language identification system that distinguishes between the most popular Arabic varieties (Adouane et al., 2016b) including MSA; Moroccan (MOR); Tunisian (TUN); Egyptian (EGY); Levantine (LEV); Iraqi (IRQ) and Gulf (GUF) Arabic. We retain only those predicted to be Algerian language, so that we can focus on language identification within Algerian Arabic, at the word level.

Table 1 gives some statistics about the labelled and unlabelled datasets. Texts refer to short texts from social media, words to linguistic words excluding punctuation and other tokens, and types to sets of words or unique words. We notice that 82.52% of the words occur less than 10 times in both datasets. This is due to the high variation of spelling and misspellings which are common in these kinds of texts.

[1] We have a documented permission from the owners/users of the used social media platforms to use their textual contributions for research.

Dataset	#Texts	#Words	#Types
Labelled	10,590	213,792	57,054
Unlabelled	189,479	3,270,996	290,629

Table 1: Information about datasets.

4 Using Labelled Data

4.1 Systems and Models

We frame the task as a sequence labelling problem, namely to assign each word in a sequence the label of the language that the word has in that context. We use three different approaches: two existing sequence labelling systems – (i) an HHM-based sequence labeller (Adouane and Dobnik, 2017); (ii) a classification-based system with various back-off strategies from (Adouane and Dobnik, 2017) which previously performed best on this task, henceforth called the state-of-the-art system; and (iii) a new system using deep neural networks (DNNs).

4.1.1 HMM system

The HMM system is a classical probabilistic sequence labelling system based on Hidden Markov Model where the probability of a label is estimated based on the history of the observations, previous words and previous labels. In order to optimise the probabilities and find the best sequence of labels based on a sequence of words, the Viterbi algorithm is used. For words that have not been seen in the training data, an constant low probability computed from the training data is assigned.

4.1.2 State-of-the-art system

The best-so-far performing system for identifying language mixing in Algerian texts is described by Adouane and Dobnik (2017). The system is a classifier-based model that predicts the language or variety of each word in the input text with various back-off-strategies: trigram and bigram classification, lexicon lookup from fairly large manually compiled and curated lexicons, manually-defined rules capturing linguistic knowledge based on word affixes, word length and character combinations, and finally the most frequent class (unigram).

4.1.3 DNN model

Recurrent Neural Networks (RNNs) (Elman, 1990) have been used extensively in sequence prediction. The most popular RNN variants are the Long Short-Term Memory (LSTMs) (Hochreiter and Schmidhuber, 1997) and the Gated Recurrent Unit (GRUs) (Cho et al., 2014).

Our neural networks consists of four layers: one embedding layer, two recurrent layers, and a dense layer with softmax activation. All our models are optimized using the *Adam* optimizer, built using the *Keras* library (Chollet, 2015), and run using a TensorFlow backend. A summary of the model architecture is shown in Figure 1. (This variant is composed of only the uncoloured (white) parts of the figure; the coloured parts are added in the model described in section 5). The DNN is provided the input character by character. We opt for character-based input rather than word-based input for two reasons. First, we expect that the internal structure of words (phonemes and morphemes) is predictive of a particular variety. This way we hope to capture contexts within words and across words. Second, we do not have to worry about the size of the vocabulary, which we would if we were to use word embeddings.

This language-identification model is trained end-to-end. Because of the nature of RNNs, the network will assign one language variant per input symbol, and thus per character — even though the tags are logically associated word-by-word. To deal with this mismatch, when training we tag each character of a word and the space which follows it with the variant of the word. When evaluating the model, we use the tag associated with the space, so that all the word has been fed to the model before a prediction is made.

We have trained models with various values for the hyper-parameters: number of layers, number of epochs, memory size, drop-out rate and the batch size, but report detailed results for the model with the best behaviour. We experimented with both GRU and LSTM RNNs and found that the GRU performs better than LSTM on our task which is in line with the results of the previous comparisons but on different tasks (Chung et al., 2014). We also found out that our best systems are optimised with the architecture shown in Figure 1 with a memory size of 200, batch size of 512 and number of epochs of 25. Increasing or decreasing these values caused the overall accuracy to drop. Using drop-out improved the performance of the systems (overall accuracy $> 90\%$) over not using it ($< 70\%$). The best results are obtained using drop-out rate of 0.2 for the recurrent layers. We

refer to this model as *DNN* in the following.

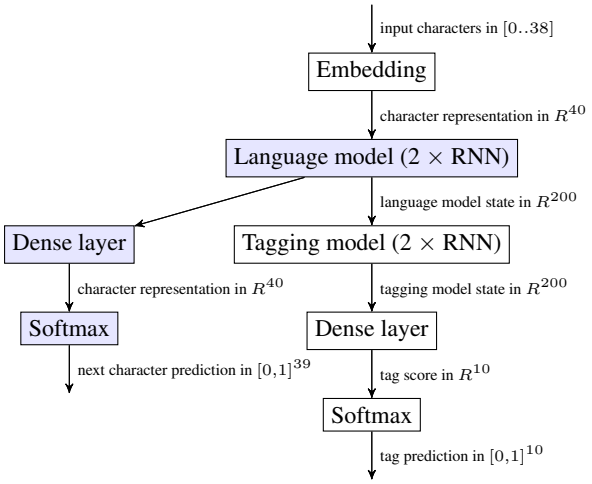

Figure 1: DNN architecture.

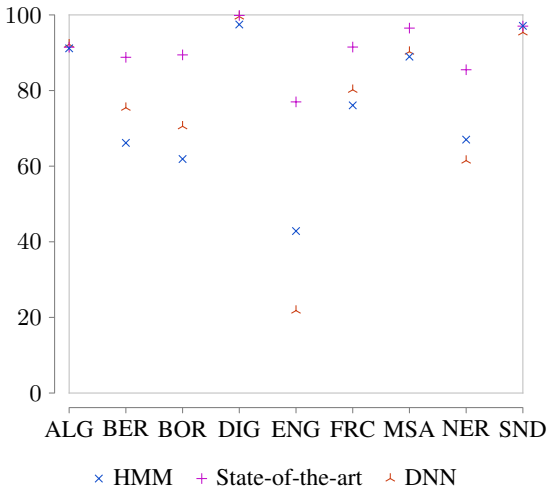

Figure 2: Models' average F-score per category.

4.2 Results

To ensure a fair comparison, all the models have been evaluated under the same conditions. We use 10-fold cross validation on all of them and report their performance measured as the average accuracy. Table 2 shows the results. Note that for the DNN we only report the results of the (best-performing) GRU models. As a baseline we take the most frequent category in the labelled data. State-of-the-art (2) outperforms slightly HMM (1). DNN (3) outperforms slightly the State-of-the-art (2). All the systems perform better than the baseline.

	Model	Accuracy (%)
1	HMM	89.29
2	State-of-the-art	89.83
3	DNN	**90.53**
4	Baseline	55.10

Table 2: Performance of the models on labelled data.

Figure 2 shows the performance of each model per category reported as average F-score. Overall the models perform better on the majority categories such as ALG (Algerian) and MSA (Modern Standard Arabic), and non linguistic categories like DIG (digits) and SND (sounds) because their patterns are more or less regular and language independent. The State-of-the-art system achieves the best performances for all categories except for ALG where it is slightly outperformed by DNN, average F-score of 91.45 and 92.22 respectively. A possible explanation for this is that the State-of-the-art system is more robust because it involves several strategies of classification. DNN performed better than HMM in all cases except for ENG (English) and SND. Both DNN and HMM struggle with minority categories like ENG, BOR (borrowing), BER (Berber), NER (Named Entities), and FRC (French). Note that in this experiment we only used the smaller labelled dataset. In the following section, we explore ways to take advantage of the additional relatively large unlabelled dataset in order to improve the performance of the systems.

5 Using Data Augmentation With Background Knowledge

5.1 Training Methods

In this section, we examine which data augmentation method performed on the unlabelled corpus can best enhance the performance of our three models. We experiment with data bootstrapping, pre-training a language model, and the combination of both methods. In each case, we are providing some form of background knowledge compared to the task described in Section 4.

5.1.1 Bootstrapping

For bootstrapping, we use the State-of-the-art system (Section 4.1.2) to label the unlabelled data without additional checking of the quality of annotation and then use this bootstrapped data in further training. We re-run the experiments described in Section 4 using the bootstrapped data as the

training data. We refer to the systems as *HMM bootstrapped*, *State-of-the-art bootstrapped*, and *DNN bootstrapped* respectively.

5.1.2 Language Model

Another way to take advantage of the unlabelled data is to train a language model (LM) on the whole data and use the internal state of the LM as input to the tagger, rather than using the raw textual input. To this end, we modify the structure of our DNN as indicated by the blue-coloured parts in Figure 1. Namely, we add two language-modelling RNN layers between the embedding and the tagging layers. They are followed by a dense layer with softmax activation, which predicts the next character in the input.

With this setup, we train the language-modelling layers on the unlabelled corpus, as a generative language model on the unlabelled data set. Thus, the output of these layers contains the information necessary to predict the next character given the previous sequence of characters. The language model is trained on 80% of the unlabelled data and evaluated on the remaining 20%. The rest of the network is then trained as in the previous case (Section 4.1.3). We stress that, in this instance, only the last two layers are trained on the language-identification task. We refer to this model as *DNN with LM*.

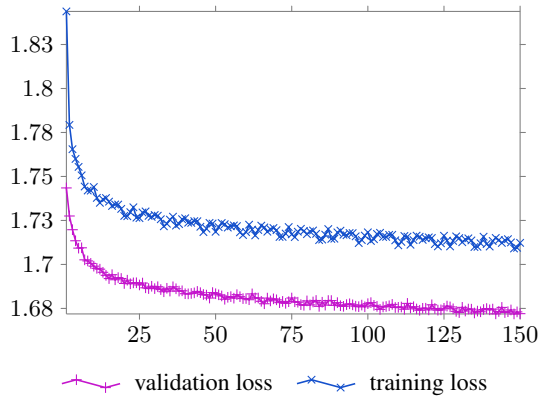

Figure 3: Language model loss through training epochs

You may notice in Figure 3 that the model is still improving (at 150 epochs), albeit slowly, even after exhausting our computational budget. Nevertheless, the model appears to be working well as a text generator. For instance, we took sentence (1) as a seed and obtained sentences that are grammatically and structurally acceptable, even if they are semantically meaningless and reproduce the many spelling variants found in the original corpus. Here are two examples:

1. المهم انا ما نقدرش نموت عليها و الله يا ختي راني معاك و الله ما نقولك انا ما نقدرش نتعلم من الحال
2. و الله ما نقول الله يبارك ما يعرفش يتبلاو و يعرف وين راه المرا الي ما يحبش يحب يقول الله يهدينا

5.1.3 Language Model and Bootstrapping

We retrain the DNN model using the pre-trained LM and the bootstrapped data in order to optimise the use of the unlabelled data. We refer to this model as *DNN bootstrapped and LM*.

5.2 Results

We evaluate all the models under the same conditions as in Section 4, using 10-fold cross validation we report the average accuracy over the folds. The evaluation set in the bootstrapping models in each fold is only taken from the labelled data while the training part consists of a combined 9-folds from the labelled data and the entire bootstrapped data. In other words, the entire bootstrapped data is added to the training data at each time. In the case of DNNs, we found again that GRUs perform significantly better than LSTM, and that bootstrapped models are optimised with drop-out rate of 0.2 whereas models with language model perform better with drop-out rate of 0.1. The obtained results are reported as the average accuracy in Table 3. For the DNN, we only report the results of the (best-performing) GRU models.

	Model	Accuracy (%)
1	HMM bootstrapped	93.97
2	State-of-the-art bootstrapped	**95.42**
3	DNN bootstrapped	93.31
4	DNN with LM	90.31
5	DNN bootstrapped and LM	90.19

Table 3: Performance of the models with background knowledge.

The best performance overall is achieved by the bootstrapped state-of-of-the-art model (2). HMM bootstrapped (1) performs slightly better than the DNN bootstrapped (3). Bootstrapping helps the State-of-the-art system and HMM more than DNN. This is due to the training nature of the DNN which is based on capturing frequent regular patterns, hence adding the bootstrapped data means introducing even more irregular patterns.

Compared to the results in Section 4.2, the DNN bootstrapped (3) outperforms all the models with the labelled data: (1), (2) and (3) in Table 2. The bootstrapping method thus improves the performance of all configurations, whether they are using DNNs or not. The reported benefits of bootstrapping are contrary to the previous observations where bootstrapping did not help (Section 2).

However, the use of the language model (4) decreases slightly (-0.22%) the performance of the DNN compared to its performance with the labelled data (3) in Table 2. The use of the bootstrapping and the language model (5) leads to no significant difference in performance in respect to (4). Overall, it appears that the usage of the language model has no strong effect. This could be caused by the noise in the data, and adding more unlabelled data makes it hard for the language model to learn all the data irregularities. Maybe the system requires more training data.

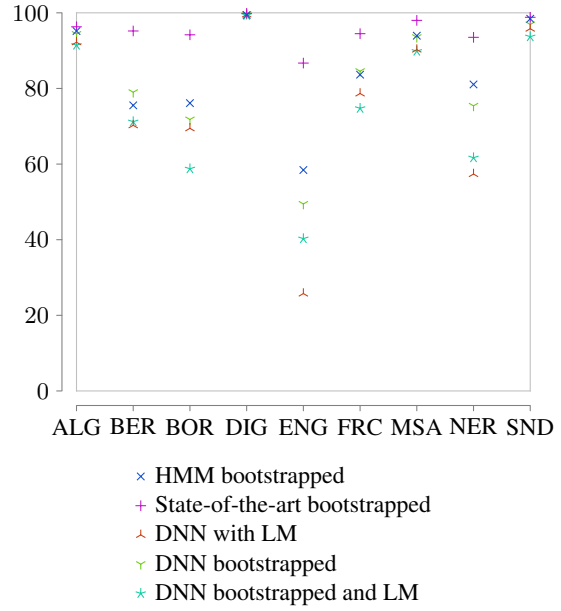

Figure 4: Models' average F-score per category.

Figure 4 sums up the performance of each model per category reported as the average F-score. The first thing to notice is that bootstrapping improves the performance of all systems, and the best performance is achieved with the State-of-the-art. This could be explained by 'the more data, the better performance'. HMM bootstrapped outperforms the DNN bootstrapped except for FRC and BER. Adding language model to the DNN causes the overall accuracy to drop compared to the DNN bootstrapped. Nevertheless, compared to the results in Figure 2, language model behaves differently with each category. For instance, it boosts the performance of the DNN on ENG, and the performance on BOR, BER, FRC over HMM. Whereas combining language model and bootstrapped data performs the worst except for BER, ENG and NER. The effect of combining bootstrapping and language model is better for minority categories: BER, ENG and NER.

Error analysis of the confusion matrices shows that all the systems are confused, chiefly between ALG and MSA, BOR and ALG, FRC and ALG. The confusions are caused mainly by the lexical ambiguity between these categories, given that we identify the language of each word in its context.

6 Conclusions

We have examined the automatic classification of language identification in mixed-language texts on limited datasets of Algerian Arabic, in particular a small unbalanced labelled dataset and a slightly larger unlabelled dataset. We tested whether the inclusion of a pre-trained LM on the unlabelled dataset and bootstrapping the unlabelled dataset can leverage the performance of the systems. Overall when using only the small labelled data, DNNs outperformed the HMM and the State-of-the-art system. However, DNNs performed better on the majority categories and struggled with the minority ones in comparison to the State-of-the-art system. Bootstrapping improved the performance of all models, both DNNs and not DNNs for all categories.

Adding a background knowledge in the form of a pre-trained LM to DNNs had a different effect per category. While it boosted the performance of the minority categories, its effect on the majority ones was not clear. Despite the generative behaviour of the LM, tested in Section 5.1.2, which showed that LM did learn the underlying structures of the unlabelled data, the effect of the encoded knowledge maybe was not suitable for our main task. This could be also caused by the high noise level in the data, even though deep learning is generally thought to handle noise well.

In our future work, we will focus on exploring (i) different DNN configurations to investigate the best ways of injecting background knowledge as well as (ii) different data pre-processing methods to normalise spelling and remove misspellings for MSA, and deal with word segmentation errors.

Acknowledgement

The research reported in this paper was supported by a grant from the Swedish Research Council (VR project 2014-39) for the establishment of the Centre for Linguistic Theory and Studies in Probability (CLASP) at the University of Gothenburg.

References

Wafia Adouane and Simon Dobnik. 2017. *Proceedings of the Third Arabic Natural Language Processing Workshop*, chapter Identification of Languages in Algerian Arabic Multilingual Documents. Association for Computational Linguistics.

Wafia Adouane, Nasredine Semmar, Richard Johansson, and Victoria Bobicev. 2016a. Automatic Detection of Arabicized Berber and Arabic Varieties. pages 63–72. The COLING 2016 Organizing Committee.

Wafia Adouane, Nasredine Semmar, Richard Johansson, and Victoria Bobicev. 2016b. 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.

Héctor Martìnez Alonso and Barbara Plank. 2017. When is Multitask Learning Effective? Semantic Sequence Prediction under Varying Data Conditions. In *EACL (long)*.

Rie Kubota Ando and Tong Zhang. 2005. A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data. *The Journal of Machine Learning Research*, 6:1817–1853.

Utsab Barman, Amitava Das, Joachim Wagner, and Jennifer Foster. 2014a. Code-mixing: A challenge for Language Identification in the Language of Social Media. In *In Proceedings of the First Workshop on Computational Approaches to Code-Switching*.

Utsab Barman, Joachim Wagner, Grzegorz Chrupala, and Jennifer Foster. 2014b. Dcu-uvt: Word-Level Language Classification with Code-Mixed Data.

Kyunghyun Cho, Bart Van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. *Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation*. arXiv preprint arXiv:1406.1078.

François Chollet. 2015. Keras. https://github.com/fchollet/keras.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. arXiv:1412.3555.

Mona Diab, Pascale Fung, Mahmoud Ghoneim, Julia Hirschberg, and Thamar Solorio. 2016. *Proceedings of the Second Workshop on Computational Approaches to Code Switching*. Association for Computational Linguistics, Austin, Texas.

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)*.

Heba Elfardy and Mona Diab. 2012. Token Level Identification of Linguistic Code Switching. In *COLING*, pages 287–296.

Jeffrey L. Elman. 1990. Finding Structure in Time. *Cognitive Science*, 14(2):179–211.

Sepp Hochreiter and Juergen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Tom Kocmi and Ondřej Bojar. 2017. Lanidenn: Multilingual Language Identification on Text Stream. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 927–936. Association for Computational Linguistics.

Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text Classification from Labeled and Unlabeled Documents Using EM. *Machine Learning, Special issue on Information Retrieval*, pages 103–134.

Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10).

David Pierce and Claire Cardie. 2001. Limitations of Co-training for Natural Language Learning from Large Datasets. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Shana Poplack and Marjory Meechan. 1998. How Languages Fit Together in Codemixing. *The International Journal of Bilingualism*, 2(2):127–138.

Younes Samih, Suraj Maharjan, Mohammed Attia, Laura Kallmeyer, and Thamar Solorio. 2016. Multilingual code-switching identification via lstm recurrent neural networks. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 50–59.

Younes Samih and Wolfgang Maier. 2016. Detecting Code-Switching in Moroccan Arabic. In *Proceedings of SocialNLP @ IJCAI-2016*.

Charles Sutton, Andrew McCallum, and Khashayar Rohanimanesh. 2007. Dynamic Conditional Random Fields: Factorized Probabilistic Models for Labeling and Segmenting Sequence Data. *Journal of Machine Learning Research*, 8(Mar):693–723.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How Transferable are Features in Deep Neural Networks? *In Advances in Neural Information Processing Systems 27 (NIPS '14), NIPS Foundation*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. *Character-level Convolutional Networks for Text Classification*. Advances in Neural Information Processing Systems 28 (NIPS 2015).

Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. 2004. *Learning with Local and Global Consistency*. In NIPS 2003.