Anupam Jamatia*, Amitava Das and Björn Gambäck

Deep Learning-Based Language Identification in English-Hindi-Bengali Code-Mixed Social Media Corpora

https://doi.org/10.1515/jisys-2017-0440
Received August 28, 2017.

Abstract: This article addresses language identification at the word level in Indian social media corpora taken from Facebook, Twitter and WhatsApp posts that exhibit code-mixing between English-Hindi, English-Bengali, as well as a blend of both language pairs. Code-mixing is a fusion of multiple languages previously mainly associated with spoken language, but which social media users also deploy when communicating in ways that tend to be rather casual. The coarse nature of code-mixed social media text makes language identification challenging. Here, the performance of deep learning on this task is compared to feature-based learning, with two Recursive Neural Network techniques, Long Short Term Memory (LSTM) and bidirectional LSTM, being contrasted to a Conditional Random Fields (CRF) classifier. The results show the deep learners outscoring the CRF, with the bidirectional LSTM demonstrating the best language identification performance.

Keywords: Language identification, code-mixing, deep learning.

2010 Mathematics Subject Classification: 68T50.

1 Introduction

Social media are clearly having an impact on language usage and have drastically changed the way people all over the world interact and communicate. The informal ways the “netizens” in social media express their feelings in written form can be considered as a sort of “online-lingo”. Code-mixing is a common phenomenon in this online-lingo where bilingual and multilingual users alternate (or switch) languages (codes) between or inside words, clauses and sentences. Code-mixing occurs most frequently in areas of the world where people are naturally multilingual, such as parts of Europe, Africa and South East Asia. Here, we will focus on India, which is home to several hundred languages, with language diversity and dialectal changes instigating frequent code-mixing. In particular, the article will look at the task of collecting and annotating code-mixed English-Hindi and English-Bengali social media texts from Facebook, Twitter and WhatsApp and at automatic language identification in these code-mixed corpora.

Research on code-mixing in conversational spoken language has been common in psycho- and socio-linguists for half a century, but the first work on computational processing of code-mixed text was only carried out in the early 1980s [24], while code-mixing in social media text started to be studied in the late 1990s [34]. Similarly, automatic language identification for monolingual data has been a well-known research problem for a long time, and there are methods showing high accuracy on both long [46] and short [28, 44] documents. However, it is only recently that language identification in code-mixed text has become topical, with shared tasks dedicated to the theme in 2014 both at EMNLP [42] and FIRE [8], followed by shared tasks at FIRE 2015 [39], EMNLP 2016 [30] and FIRE 2016 [4].

***Corresponding author: Anupam Jamatia,** National Institute of Technology, Agartala, Tripura, India, e-mail: anupamjamatia@gmail.com
Amitava Das: Indian Institute of Information Technology, Sricity, Andhra Pradesh, India
Björn Gambäck: Norwegian University of Science and Technology, Trondheim, Norway

The paper is organized as follows: Section 2 discusses the background and related work on word-level language identification in code-mixed social media corpora. The collection and annotation of code-mixed corpora and the language tagset used are described in Section 3. The actual language identification experiments are discussed in Section 4, starting by presenting a conditional random fields (CRFs) baseline approach and the features used in it, and then detailing experiments using two different deep learning-based approaches: long short-term memory (LSTM) and bidirectional LSTM. The results are further elaborated on in Section 5, whereas Section 6 sums up and points to directions for future research.

2 Related Work

Code-mixing is a natural phenomenon in conversations between multilingual speakers, where at least two linguistic units from different languages are combined in some way [17, 19, 31, 32]. A *code* is a language, or a variety or style of a language, and mixing or switching occurs to some extent in the speech of most multilinguals. In text, this phenomenon [20] is mostly seen in different social media, such as Facebook, Twitter and WhatsApp, and when the users are from multilingual societies. Although social media often contain more ungrammatical text than more formal writings, the relative occurrence of non-standard English syntax tends to be fairly constant across various types of social media [2], with the level of formality of the language depending more on the style of the writer than on the media as such. However, in general, tweets (Twitter messages) tend to be more formal than chat messages (e.g. on WhatsApp) in that they more often follow grammatical norms and use standard lexical items [22], while chats are more conversational [35], and hence less formal. Due to the ease of availability of Twitter, most previous research on social media text has focused on tweets; however, the conversational nature of chats tends to increase the level of code-mixing [6]; thus, we have collected data from Twitter, Facebook and WhatsApp for comparison.

Different systematic approaches or mixture of approaches have been taken to the problem of identifying code-mixing, including the use of dictionaries, language models, morphological and phonological analysis, while utilizing different features such as parts-of-speech (POS), lemmata, and word and character *n*-grams. A range of machine learning algorithms have been tried, such as sequence labeling with Hidden Markov Models and CRFs, as well as word-level classification using Maximum Entropy, Naïve Bayes, Logistic Regression, and Support Vector Machines, with researchers having worked on code-mixing of various language pairs such as English-Spanish [41], Turkish-Dutch [33], English-Maltese [37], Romanized Moroccan Arabic (Darija)-English-French [45], modern standard Arabic-Egyptian dialect [12], Mandarin-English [27, 29] and Malay-English [48]. Balažević et al. [1] showed that incorporating user-specific information leads to an improvement of detection results using several machine learners on a Twitter data set in 16 different languages. Zazo et al. [50] analyzed an end-to-end LSTM-based system for language identification and then proposed several conditions based on the NIST Language Recognition Evaluations 2009 and 2015, comparing performance with respect to a classical i-vector system [49].

Here, we are, in particular, concerned with language identification of code-mixed social media text involving Indian languages. Thus, though Diab and Kamboj [11] briefly explained the process of corpus collection and suggested crowd sourcing as a good method for annotating formal (non-social media) Hindi-English code-mixed data, the first Indian code-mixing social media text corpus (Bengali-Hindi-English) was reported by Das and Gambäck [10] in the context of language identification at the word level. Bali et al. [3] then argued that structural and discourse linguistic analysis is required in order to fully analyze code-mixing for Indian languages. Gupta et al. [18] discussed the phenomenon in the context of information retrieval, applying deep learning techniques to the problem of identifying term equivalents in code-mixed text, while Jhamtani et al. [23] proposed an idea of utilizing the patterns in Hindi-English sentences by considering the language and the POS of consecutive words. Chanda et al. [7] discussed several problematic phenomena for language identification in code-mixed text, taking two small (about 500 tokens each) English-Bengali corpora as basis, from FIRE 2013 and from Facebook chats.

3 Corpus Collection

For this work, code-mixed corpora were collected from the popular social media platforms Facebook, Twitter and WhatsApp. 763 and 2017 Facebook posts were collected for English-Bengali (EN-BN) and English-Hindi (EN-HI), respectively; from Twitter, 4880 and 3740 tweets for EN-BN and EN-HI, respectively; and from WhatsApp, 1042 and 979 messages, respectively, for EN-BN and EN-HI.

The tweets were collected with a system developed on top of the Java-based Twitter API[1] and mainly gathered from Bengali- and Hindi-speaking politicians, celebrities and athletes with Twitter handles such as `@MamataOfficial`, `@sujoy_g`, `@rituparnas11`, `@SirPareshRawal` and `@virendersehwag`, using mixed queries on various current topics in both languages, for example:

```
          “Kashmir” AND “main”;
          “college” AND “kobe”;
     “summer vacation” AND “shuru”;
      “Supreme Court” AND “ballo”.
```

The EN-HI and EN-BN Facebook posts were collected from campus-related university billboard postings on the “Confession” pages of the Indian Institute of Technology, Bombay[2] (a predominantly Hindi-speaking university) and Jadavpur University, Kolkata,[3] which is mainly Bengali-speaking. The posts on these pages are mainly of the form of one longer story (a “confession” about something a student did on campus) followed by several shorter chat-style comments from other users. The “confession” posts tend to be written in more formal language and mainly in English with some Hindi or Bengali mixed in, while the comments are more informal in style and freely mix English-Hindi or English-Bengali.

WhatsApp is a private social network; therefore, WhatsApp data collection was relatively more challenging than for the other two social media. Data were collected at the personal level, mostly from undergraduate students and from friends and relatives of the authors. WhatsApp has an option to email the chat data; thus, we opted for that and collected data for both language pairs.

After the collection, the corpora were tokenized and annotated manually at the word level using the tagset of Barman et al. [5]. The CMU tokenizer [13] was used for tokenization; although it originally was developed for English tweets, empirical testing showed that it also works reasonably well for Indian languages. The annotation process was mainly carried out in-house by students and faculty members. None of the annotators were linguists, but all annotators were confirmed to be fluent in both languages. However, while all students in India know English, there are differences in how many other languages they know. Thus, do Bengali speakers commonly know Hindi as a second language, but the reverse is mostly not true. Hence, the annotation tasks were assigned in accordance with language knowledge. Altogether six annotators took part in the entire annotation process, with two annotators for each social media platform. Initially, in the three social media category, average inter-annotator agreement (using Cohen’s κ) was 96%, and for almost 3%, more annotators agreed after discussion on the second possibility. Hence, finally, the agreement between the annotators was 99%. The maximum confusion between the annotators was caused by “named entities” (NE) and “mixed” tokens. However, such ambiguities can normally be resolved by inspecting the context.

The token-level language distributions of the corpora are reported in Table 1. The third group of columns (EN, HI, and BN) in the table shows the share of language-specific tokens in each corpus. Among the three EN-HI corpora (Table 1A), WhatsApp posts are mainly written in Hindi, with over 80% of the language-specific tokens being in Hindi, whereas on Facebook and Twitter, Hindi is only narrowly the main language, represented by slightly less than 60% of those tokens. In the EN-BN corpora (Table 1B), Bengali is only marginally the most common language in Twitter and WhatsApp, with just over 50% of the language-specific tokens,

1 http://twitter4j.org/.
2 http://www.facebook.com/Confessions.IITB.
3 https://www.facebook.com/JU-Confessions-1609256459297929/.

Table 1: Corpora Token-Level Language Distribution (%).

Source	Tokens	EN	HI	BN	Univ	NE	Acro	Mixed	Undef
(A) English-Hindi									
Facebook	180,235	34.44	50.39	–	13.30	1.50	0.08	0.15	0.11
Twitter	72,227	32.85	43.86	–	19.65	2.76	0.58	0.03	0.23
WhatsApp	4057	14.59	71.99	–	12.02	1.23	0.14	–	–
Total	256,519	33.68	48.90	–	15.07	1.85	0.22	0.11	0.14
(B) English-Bengali									
Facebook	24,314	20.51	0.47	59.04	14.46	4.18	1.30	0.01	0.004
Twitter	50,761	36.54	2.03	38.84	18.73	2.71	0.77	–	0.35
WhatsApp	11,664	32.29	0.14	40.47	23.35	1.94	1.78	–	–
Total	86,739	31.47	1.34	44.72	18.15	3.02	1.06	0.004	0.20
(C) English-Hindi-Bengali									
Facebook+ Twitter+ WhatsApp	343,258	33.12	36.88	11.30	15.85	2.14	0.43	0.08	0.16

Univ indicates language-independent symbols such as punctuation marks, NE, Named entities; Acro, abbreviations and acronyms; Mixed, tokens showing code-mixing down at the character level (i.e. word-internal); Undef, unclear/impossible to tag.

while it clearly is the main language of the EN-BN Facebook corpus, making up over 70% of the language tokens of the corpus. Note that some Hindi appear in the English-Bengali corpora, whereas the English-Hindi corpora contain no Bengali. The total mixing of both pairs of corpora (Table 1C) gave nearly equivalent shares of tokens in Hindi (45.4%) and English (40.7%), whereas Bengali (13.9%) hence occurs in a smaller portion of the in total 81.3% of all tokens that are language-specific.

4 Experiments

This section discusses the language identification experiments, starting by introducing a baseline approach utilizing a classifier based on CRFs, and then describing the features used for the classification. The bulk of the section is devoted to approaches using deep learning networks, together with an elaboration on the word embeddings that are used to train those networks. Finally, Section 4.3 reports the results of using both types of methods for language identification in code-mixed data.

4.1 Baseline Approach

To establish a baseline, a supervised model was built using a CRF classifier created in the C++-based open-source CRF++ package [25] for learning sequential data. CRFs are conditional, undirected discriminative graphical models that can easily incorporate a large number of non-independent, arbitrary features while still having efficient procedures for non-greedy finite-state inference and training [26]. CRFs can be used to classify each token in a corpus into one of several categories based on different features, most of which are extracted from the training data.

CRFs are used to calculate the conditional probabilities of values (labels) on designated output nodes given values assigned to other designated input nodes. The conditional probability of a sequence of labels (states) $l=<l_1, l_2, \ldots, l_T>$, given an observation (input) sequence $o=<o_1, o_2, \ldots, o_t>$ is calculated as

$$P_\lambda(l|o)=\frac{1}{Z_o}exp\left(\sum_{t=1}^{T}\sum_{k=1}^{K}\lambda_k \times f_k(l_{t-1}, l_t, o, t)\right), \tag{1}$$

where $f_k(l_{t-1}, l_t, o, t)$ is an arbitrary feature function over its arguments and λ_k is a learned weight for each feature function. The features may take on any positive real values, but typically they are binary. Then each feature function has a value of 0 for most cases and is only set to be 1 when it is triggered, that is when the observation at time t matches certain sought-for properties given the previous state l_{t-1} and the resulting state l_t. The actual trigger conditions are specific to each feature function and not restricted by the CRF itself. Hence, feature functions can be designed to indicate a high probability for a specific state change or just a state change in general. To make the conditional probabilities of all label sequences sum up to 1, the normalization factor Z_o is calculated as the sum of all possible label sequences, so $Z_o = \sum_l \exp(\sum_t \sum_k \lambda_k \times f_k(l_{t-1}, l_t, o, t))$. Hence, the number of terms in the sum can be exponentially large, but it can still be calculated efficiently using the Viterbi dynamic programming algorithm [9].

To train a CRF, the objective function to be optimized is the penalized log-likelihood of the state sequences given the observation sequences:

$$L_\lambda = \sum_{j=1}^{N} \log(P_\lambda(l^{(j)} \mid o^{(j)})) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}, \tag{2}$$

where $<o^{(j)}, l^{(j)}>$ is the labeled training data and the second sum is a regularization factor, which gives a penalty to weights with too large norms to avoid overfitting. Here, the penalty is thus based on the Euclidean norm and a regularization parameter $1/\sigma^2$, which corresponds to assigning a Gaussian prior with mean 0 and variance σ^2 to the parameters λ_k [43]. This facilitates optimization by making the likelihood surface strictly convex. The optimum can be calculated using quasi-Newton methods such as the limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm [40], which is highly efficient and results only in minor changes in accuracy due to changes in the parameters λ_k.

Feature selection plays a key role in supervised language identification using CRF. The important features of the task have been identified based on the training data set. The features include the pivot (target) word itself, a fixed length size n characters stripped from the beginning and ending of the pivot word (where n is up to four characters), whether the pivot word starts with capital letters or digits, whether the pivot word is alphanumeric or contains punctuation or special characters, and the previous word's language tag.

4.2 Deep Learning-Based Approach

A recurrent neural network (RNN) is a deep learning approach where the recurrent connections are loops in the network that allow it to maintain a memory based on history information, enabling the model to predict the current output conditioned on long-distance features. RNNs have been applied to various language processing applications, such as language modeling, speech recognition, machine translation, conversation modeling and question answering. An LSTM network [15, 21] is a version of RNN where the hidden layer updates are replaced by purpose-built gated memory cells. As a result, LSTMs may be better at finding and exploiting long range dependencies in the data. The output of an LSTM hidden layer h_t given input x_t is computed as follows [16]:

$$\begin{cases} i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \\ f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \\ o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \\ c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\ h_t = o_t \tanh(c_t), \end{cases} \tag{3}$$

where σ is the logistic sigmoid function and i, f, o and c are a cell's input gate, forget gate, output gate and activation vectors, respectively. These multiple gates allow the cell in LSTM memory block to store information over long periods of time, thereby avoiding the vanishing gradient problem [14].

In end-to-end sequence labeling, we can access both past and future input features at a given time; thus, we can utilize a bidirectional network, a Bi-LSTM [16]. Bi-LSTM networks have two parallel layers propagating, memorizing the information of sentences as read in the left-to-right and right-to-left direction, respectively [38]. The forward and backward passes of each pair of layers are carried out in a way similar to regular neural networks, but since there are two LSTM layers in the network, the hidden layer output function must be split into two cases, for the forward and the backward passes over the input, separately:

$$\begin{cases} h_{ft} = H(W_{xh_f} x_t + W_{h_f h_f} h_{f_{t-1}} + b_{h_f}) \\ h_{bt} = H(W_{xh_b} x_t + W_{h_b h_b} h_{b_{t-1}} + b_{h_b}) \end{cases}, \tag{4}$$

where $h_f \in R^d$ and $h_b \in R^d$ denote the output vector of the forward and backward layers, respectively. Following Yao and Huang [47], the combination of forward and backward layers is defined as a single Bi-LSTM layer, giving the final output:

$$y_t = W_{h_f} h_{f_t} + W_{h_b} h_{b_t} + b_y. \tag{5}$$

The proposed model for deep learning was trained using word embeddings. Word embeddings are dense vector space representations of words, built by factoring a co-occurrence matrix through various means. It has been shown that words that have related meanings (or at least occur in the similar contexts) are close in this space according to the cosine similarity and that some analogy relationships are encoded in transitions. Here the GloVe [36] word embedding software[4] was used for training purposes, with a 100 dimensional word embedding trained on a 27,147,297 words corpus consisting of both code-mixed data (631,195 sentences with a total 10,142,090 words) collected from Facebook (84.97%) and Twitter (15.13%), and additional monolingual data (17,005,207 words) extracted from Wikipedia.[5] The word embeddings were trained using a context window of size 5. In general, larger window sizes create more topical representations, and smaller window sizes focus on syntactic aspects.

The models read a sentence from the corpus word by word, mapping the word IDs to their embedding representations, and then feed the representations to an LSTM or Bi-LSTM recurrent unit. The hidden state of the recurrent unit after reading all the words is used as input to a dense layer with a softmax activation function to predict the probability of the language identification tags. The embedding layer can be initialized with the trained word embedding. Since it should be possible to use them with words that are not in the training data, the word embeddings were not fine-tuned, that is, they do not account for cross-lingual homographs such as "main", which occurs both in English and Hindi (meaning "myself"), or "to" which means "so" in Bengali. To be able to properly analyze such words, the embeddings would need to be fine-tuned, but doing so would make handling of out-of-vocabulary words harder.

4.3 Results

The CRF baseline experiments were run using the Java-based miralium package[6] with a 80:20 split between training and test data for each corpus. The test data for the joint model (English-Hindi-Bengali) then contained 69,246 tokens with the following class distributions: EN: 23,289 tokens (33.9%), HI: 25,610 (37.3%), BN: 4729 (6.85%), Univ: 13,416 (19.5%), NE: 1562 (2.28%), Acro: 546 (0.795%), Mixed: 23 (0.0335%), and Undef: 171 (0.249%). The CRFs obtained an accuracy of 76.70% with a 75.06% weighted average F-score on the mixture of all corpora, i.e. English-Hindi-Bengali, compared to 83.93% accuracy with F-score 81.57% and 91.54% accuracy with F-score 91.02% on the language pairs English-Bengali and English-Hindi, respectively, as reported in Table 2.

4 https://nlp.stanford.edu/projects/glove/.
5 http://mattmahoney.net/dc/text8.zip.
6 https://github.com/benob/miralium.

Table 2: System Performance (Accuracy and F_1-score) on the Test Data (%).

Languages	CRF		LSTM		Bi-LSTM	
	Acc.	F_1	Acc.	F_1	Acc.	F_1
EN-BN	83.93	81.57	88.27	88.19	87.57	88.23
EN-HI	91.54	91.02	86.97	87.01	85.49	86.20
EN-BN-HI	76.70	75.06	86.61	86.00	87.16	87.07

The deep learning setups were trained using the “Nadam” optimizer with a dropout rate of 0.5 to reduce overfitting in the models. Generally, a small dropout rate of 20%–50% of neurons provides a good starting point. A too low dropout probability has minimal effect, whereas a value too high results in under-learning by the network. The Python Deep Learning Keras library[7] was used on top of Theano[8] to build and train the models. The model training was carried out on a 2.3-GHz Intel Xeon processor with 64-GB RAM and NVIDIA Tesla K40 GPU, with a batch size of 64 and with a 50:20:30 split among training, validation and test sets for each of the corpora.

The results obtained after training for 20 epochs on the code-mixed social media data sets are summarized in Table 2. Increasing the number of epochs to 30 and 50 gave no improvements. As the table shows, reasonable accuracy levels were achieved without usage of any additional information and hand-crafted features.

5 Discussion

An important observations is that the word vector plays a crucial role in neural network tasks with the minimal availability of code-mixed data in language identification. The default parameters of the optimizer were used as increasing the number of parameters can lead to overfitting. The results in Table 2 show that there only is a slight performance difference between the LSTM (86.61% accuracy with 86.00% F-score) and Bi-LSTM approaches (87.16% accuracy with 87.07% F-score) when applied to the trilingual English-Bengali-Hindi data set. However, a formidable improvement was achieved in relation to the baseline CRF approach, which achieved a 76.70% accuracy with 75.06% F-score on that data set.

In addition, on the bilingual English-Bengali data set, the deep learners clearly outperformed the CRF classifier, but in contrast, the CRF baseline approach reached an accuracy of 91.54% with F-score 91.02% on the English-Hindi pair, which is much better than the LSTM (accuracy 86.97% with F-score 87.01%) and Bi-LSTM (accuracy 85.49% with F-score 86.20%) approaches. CRFs outperform LSTM only on the EN-HI data. This inconsistency might partially be due to the nature of the social media data and the difference in the type of sources between the language pairs, with the English-Hindi data having a clear bias toward Facebook and only a small fraction of WhatsApp posts, whereas the English-Bengali data are more evenly distributed among the three different social media, but with the lion share coming from Twitter.

Furthermore, it is noticeable in Table 2 that all the learners perform better on the bilingual data than on the trilingual, with, in particular, the CRF classifier showing clearly worse performance on the EN-BN-HI corpus. This is most likely a result of the trilingual task being more difficult in itself and getting extra complicated by the fact that some (short) common words can belong to all three languages. An example of such a word is “to”, which means “so” in both Bengali and Hindi. However, the reduced capacity on the trilingual data is not completely consistent across the classifiers; thus, even though the CRF’s performance drops drastically, the deep learners appear more robust. In fact, the performance on the Bengali- and Hindi-specific tokens in the joint model was hardly hurt at all in the LSTM, which showed an accuracy 86.51% on the BN

7 https://keras.io/.
8 https://github.com/Theano/Theano.

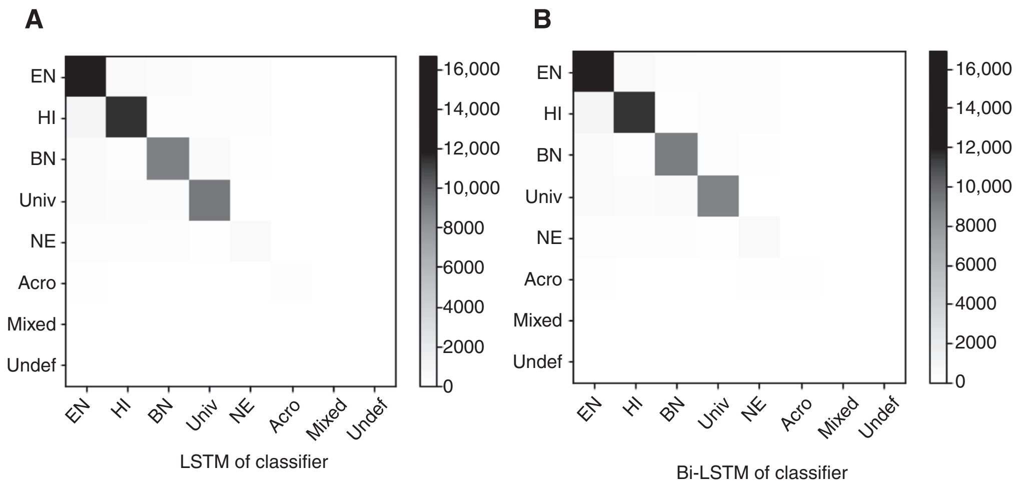

Figure 1: Confusion Matrices for the Deep Learners on the EN-HI-BN Corpus Test Set. (A) LSTM and (B) Bi-LSTM classifier.

tokens and 86.80% on the HI tokens, compared to 86.61% in total overall tokens. The Bi-LSTM even achieved a slightly better accuracy on those tokens (89.19% on BN and 87.53% on HI) than its average performance on all tokens (87.16%). Figure 1 shows the confusion matrices for the two deep learners on the EN-BN-HI corpus, with the true labels on the x-axes and the predicted labels on the y-axes. As can be seen, the five largest classes dominate the correct labeling, but even more so, the classifier output (as could be expected), and so the Bi-LSTM, for example, does not predict a single “mixed” tag, even though there were 47 such tokens in the data. Interestingly, the classifiers rarely confuse Bengali and Hindi tokens but very frequently misclassify English as one of the Indian languages and the other way around.

6 Conclusion

This article has shown that straightforward and relatively unoptimized deep learning-based approaches to the task of language identification can achieve reasonable accuracy levels compared to a supervised approach such as CRF. In addition, given the scarcity of data, the proposed LSTM approach improves on the CRF-based approach by incorporating pre-trained word embeddings instead of learning those from data, allowing the classifiers to learn in an unsupervised setting.

This work represents a preliminary study, and further work is needed to evaluate also other deep learning models such as convolutional neural networks on more extensive data sets of varied sequence length as well as other word-embedding possibilities and handcrafted features.

Bibliography

[1] I. Balažević, M. Braun and K.-R. Müller, Language detection for short text messages in social media, *arXiv preprint arXiv:1608.08515* (2016).

[2] T. Baldwin, P. Cook, M. Lui, A. MacKinlay and L. Wang, How noisy social media text, how different social media sources?, in: *Proceedings of the 6th International Joint Conference on Natural Language Processing*, pp. 356–364, AFNLP, Nagoya, Japan, October 2013.

[3] K. Bali, J. Sharma, M. Choudhury and Y. Vyas, “I am borrowing *ya* mixing?”: an analysis of English-Hindi code mixing in Facebook, in: *1st Workshop on Computational Approaches to Code Switching*, pp. 116–126, ACL, Doha, Qatar, October 2014.

[4] S. Banerjee, K. Chakma, S. K. Naskar, A. Das, P. Rosso, S. Bandyopadhyay and M. Choudhury, Overview of the mixed script information retrieval (MSIR) at FIRE, in: *Proceedings of the Workshops at the 8th Forum for Information Retrieval Evaluation*, pp. 7–10, Springer, Kolkata, December 2016.

[5] U. Barman, J. Wagner, G. Chrupała and J. Foster, DCU-UVT: word-level language classification with code-mixed data, in: *1st Workshop on Computational Approaches to Code Switching*, pp. 127–132, ACL, Doha, Qatar, October 2014.
[6] M. S. Cárdenas-Claros and N. Isharyanti, Code-switching and code-mixing in Internet chatting: Between 'yes', 'ya', and 'si' – a case study, *J. Comput. Mediat. Commun.* **5** (2009), 67–78.
[7] A. Chanda, D. Das and C. Mazumdar, Unraveling the English-Bengali code-mixing phenomenon, in: *2nd Workshop on Computational Approaches to Code-Switching*, pp. 80–89, ACL, Austin, TX, November 2016.
[8] M. Choudhury, G. Chittaranjan, P. Gupta and A. Das, Overview of FIRE 2014 track on transliterated search, in: *Proceedings of the Workshops at the 6th Forum for Information Retrieval Evaluation*, Bangalore, December 2014.
[9] A. Culotta and A. McCallum, Confidence estimation for information extraction, in: *Proceedings of the 2004 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 109–112, ACL, Boston, Massachusetts, May 2004.
[10] A. Das and B. Gambäck, Code-mixing in social media text: the last language identification frontier?, *Traitement Automatique des Langues* **54** (2013), 41–64.
[11] M. Diab and A. Kamboj, *Feasibility of Leveraging Crowd Sourcing for the Creation of a Large Scale Annotated Resource for Hindi English Code Switched Data: A Pilot Annotation*, DTIC Document, Report, 2011.
[12] H. Elfardy, M. Al-Badrashiny and M. Diab, Code switch point detection in Arabic, in: *International Conference on Application of Natural Language to Information Systems*, pp. 412–416, Springer, Berlin, Heidelberg, 2013.
[13] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan and N. A. Smith, 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*, 2: short papers, pp. 42–47, ACL, Portland, OR, June 2011.
[14] A. Graves, *Supervised sequence labelling with recurrent neural networks*, 385, Springer Science & Business Media, Berlin, Heidelberg, 2012.
[15] A. Graves and J. Schmidhuber, Framewise phoneme classification with bidirectional LSTM and other neural network architectures, *Neural Netw.* **18** (2005), 602–610.
[16] A. Graves, A.-R. Mohamed and G. Hinton, Speech recognition with deep recurrent neural networks, in: *Proceedings of the 2013 International Conference on Acoustics, Speech and Signal Processing*, pp. 6645–6649, IEEE, NJ, USA, 2013.
[17] J. J. Gumperz, *Discourse strategies*, 1, Cambridge University Press, Cambridge, UK, 1982.
[18] P. Gupta, K. Bali, R. E. Banchs, M. Choudhury and P. Rosso, Query expansion for mixed-script information retrieval, in: *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 677–686, ACM, Gold Coast, Queensland, Australia, 2014.
[19] M. Gysels, French in urban Lubumbashi Swahili: codeswitching, borrowing, or both?, *J. Multiling. Multicult. Dev.* **13** (1992), 41–55.
[20] S. C. Herring, Media and language change: introduction, *J. Hist. Pragmat.* **4** (2003), 1–17.
[21] S. Hochreiter and J. Schmidhuber, Long short-term memory, *Neural Comput.* **9** (1997), 1735–1780.
[22] Y. Hu, K. Talamadupula and S. Kambhampati, Dude, srsly?: the surprisingly formal nature of Twitter's language, in: *Proceedings of the 7th International Conference on Weblogs and Social Media*, AAAI, Boston, MA, July 2013.
[23] H. Jhamtani, S. K. Bhogi and V. Raychoudhury, Word-level language identification in bi-lingual code-switched texts, in: *Proceedings of the 26th Pacific Asia Conference on Language, Information and Computation*, pp. 348–357, Chulalongkorn University, Phuket, Thailand, 2014.
[24] A. K. Joshi, Processing of sentences with intra-sentential code-switching, in: *Proceedings of the 9th International Conference on Computational Linguistics*, pp. 145–150, ACL, Prague, Czechoslovakia, July 1982.
[25] T. Kudo, *CRF++: Yet Another CRF Toolkit*, 2005.
[26] J. D. Lafferty, A. McCallum and F. Pereira, Conditional random fields: probabilistic models for segmenting and labeling sequence data, in: *Proceedings of the 18th International Conference on Machine Learning*, pp. 282–289, Morgan Kaufmann Publishers Inc., Williamstown, MD, June 2001.
[27] Y. Li, Y. Yu and P. Fung, A Mandarin-English code-switching corpus, in: *Proceedings of the 8th International Conference on Language Resources and Evaluation*, pp. 2515–2519, ELRA, Istanbul, Turkey, May 2012.
[28] M. Lui and T. Baldwin, langid.py: an off-the-shelf language identification tool, in: *Proceedings of the ACL 2012 System Demonstrations*, pp. 25–30, Association for Computational Linguistics, Jeju Island, Korea, 2012.
[29] D.-C. Lyu, T.-P. Tan, E.-S. Chng and H. Li, Mandarin-English code-switching speech corpus in South-East Asia: SEAME, *Lang. Resour. Eval.* **49** (2015), 581–600.
[30] G. Molina, F. AlGhamdi, M. Ghoneim, A. Hawwari, N. Rey-Villamizar, M. Diab and T. Solorio, Overview for the second shared task on language identification in code-switched data, in: *Second Workshop on Computational Approaches to Code-Switching*, pp. 40–49, ACL, Austin, TX, November 2016.
[31] P. Muysken, *Bilingual speech: a typology of code-mixing*, Cambridge University Press, Cambridge, UK, 2000.
[32] C. Myers-Scotton, *Duelling languages: grammatical structure in code-switching*, Clarendon Press, Oxford, New York, 1993.
[33] D. Nguyen and A. Seza Dogruöz, Word Level language identification in online multilingual communication, in: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 857–862, ACL, Seattle, WA, October 2013.

[34] J. Paolillo, Language choice on SOC.CULTURE.PUNJAB, *EJC/REC* **6** (1996), n3.
[35] J. Paolillo, The virtual speech community: social network and language variation on IRC, *J. Comput. Mediat. Commun.* **4** (1999), JCMC446.
[36] J. Pennington, R. Socher and C. D. Manning, Glove: global vectors for word representation, in: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543, ACL, Doha, Qatar, October 2014.
[37] M. Rosner and P.-J. Farrugia, A tagging algorithm for mixed language identification in a noisy domain, in: *Proceedings of the 8th Annual INTERSPEECH Conference*, 3, pp. 1941–1944, ISCA, Antwerp, Belgium, August 2007.
[38] M. Schuster and K. K. Paliwal, Bidirectional recurrent neural networks, *IEEE Trans. Sig. Process.* **45** (1997), 2673–2681.
[39] R. Sequiera, M. Choudhury, P. Gupta, P. Rosso, S. Kumar, S. Banerjee, S. K. Naskar, S. Bandyopadhyay, G. Chittaranjan, A. Das and K. Chakma, Overview of FIRE-2015 shared task on mixed script information retrieval, in: *Proceedings of the Workshops at the 7th Forum for Information Retrieval Evaluation*, pp. 19–25, ACM, Gandhinagar, India, December 2015.
[40] F. Sha and F. Pereira, Shallow parsing with conditional random fields, in: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology – Volume 1*, pp. 134–141, Association for Computational Linguistics, Edmonton, Canada, 2003.
[41] T. Solorio and Y. Liu, Part-of-speech tagging for English-Spanish code-switched text, in: *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pp. 1051–1060, ACL, Honolulu, HI, October 2008.
[42] T. Solorio, E. Blair, S. Maharjan, S. Bethard, M. Diab, M. Gohneim, A. Hawwari, F. AlGhamdi, J. Hirschberg, A. Chang and P. Fung, Overview for the first shared task on language identification in code-switched data, in: *1st Workshop on Computational Approaches to Code Switching*, pp. 62–72, ACL, Doha, Qatar, October 2014.
[43] C. Sutton and A. McCallum, An introduction to conditional random fields, *FTML* **4** (2011), 276–373.
[44] E. Tromp and M. Pechenizkiy, Graph-based n-gram language identification on short texts, in: *Proceedings of the 20th Machine Learning Conference of Belgium and the Netherlands*, pp. 27–34, The Hague, 2011.
[45] C. Voss, S. Tratz, J. Laoudi and D. Briesch, Finding Romanized Arabic dialect in code-mixed tweets, in: *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pp. 188–199, ELRA, Reykjavik, Iceland, May 2014.
[46] F. Xia, W. D. Lewis and H. Poon, Language ID in the context of harvesting language data off the web, in: *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 870–878, Association for Computational Linguistics, Tehnografia Digital Press, Athens, Greece, 2009.
[47] Y. Yao and Z. Huang, Bi-directional LSTM recurrent neural network for Chinese word segmentation, in: *Proceedings of the 23rd International Conference on Neural Information Processing*, 4, Springer, Kyoto, Japan, October 2016.
[48] Y.-L. Yeong and T.-P. Tan, Applying grapheme, word, and syllable information for language identification in code switching sentences, in: *Asian Language Processing (IALP), 2011 International Conference on*, pp. 111–114, IEEEXplore, Penang, Malaysia, 2011.
[49] R. Zazo, A. Lozano-Diez, J. Gonzalez-Dominguez, D. T. Toledano and J. Gonzalez-Rodriguez, Language identification in short utterances using long short-term memory (LSTM) recurrent neural networks, *PLoS One* **11** (2016), e0146917.
[50] R. Zazo, A. Lozano-Diez and J. Gonzalez-Rodriguez, Evaluation of an LSTM-RNN system in different NIST language recognition frameworks, in: *Proceedings of Speaker Odyssey*, Bilbao, Spain, 2016.