# Dependency Parser for Bengali-English Code-Mixed Data enhanced with a Synthetic Treebank

**Urmi Ghosh**
MT & NLP Lab, KCIS
LTRC, IIIT-H
Hyderabad, India
urmi.ghosh@research.iiit.ac.in

**Dipti Misra Sharma**
MT & NLP Lab, KCIS
LTRC, IIIT-H
Hyderabad, India
dipti@iiit.ac.in

**Simran Khanuja**
BITS, Pilani
KK Birla Goa Campus
Goa, India
khanuja.simran7@gmail.com

## Abstract

The development of code-mixing (CM) NLP systems has significantly gained importance in recent times due to an upsurge in the usage of CM data by multilingual speakers. However, this proves to be a challenging task due to the complexities created by the presence of multiple languages together. The complexities get further compounded by the inconsistencies present in the raw data on social media and other platforms. In this paper, we present a neural stack based dependency parser for CM data of Bengali and English by utilizing pre-existing resources for closely related Hindi and English CM treebank as well as monolingual treebanks for Bengali, Hindi and English. To address the issue of scarcity of annotated resources for Bengali-English CM pair, we present a rule based system to computationally generate a synthetic code-mixing treebank for Bengali and English (Syn-BE) which is used to further improve the accuracy of our dependency parser. For evaluation purpose, we present a dataset of 500 Bengali-English tweets annotated under Universal Dependencies scheme.

## 1 Introduction

Code-mixing refers to the mixing of various linguistic units (morphemes, words, modifiers, phrases, clauses and sentences) primarily from two participating grammatical systems within a sentence (Bhatia and Ritchie, 2008). This is essentially different from code-switching which refers to the co-occurrence of speech extracts belonging to two different grammatical systems (Gumperz, 1982). The occurrence can be both inter-sentential or intra-sentential, however there are strict phrasal boundaries and within one lexical unit, the syntax of only one language is maintained. Since the more recent works have not focused on the differences between the two phenomena, we will use these two terms interchangeably.

Recently, code-mixing which was often only observed in speech, has pervaded almost all forms of communication due to the growing popularity and usage of social media platforms by multilingual speakers (Rijhwani et al., 2017). Therefore, there has been considerable effort in building CM NLP systems such as language identification (Nguyen and Dogruoz, 2013; Solorio et al., 2014; Barman et al., 2014; Rijhwani et al., 2017), normalization and back-transliteration (Dutta et al., 2015). Part-of-speech (POS) and chunk tagging for code-mixing data for various South Asian languages with English have been attempted with promising results (Sharma et al., 2016; Nelakuditi et al., 2016). Ammar et al. (2016) developed a single multilingual parser trained on multilingual set of treebanks that outperformed monolingually-trained parsers for several target languages. In the CoNLL 2018 shared task, several participating teams developed multilingual dependency parsers that integrated cross-lingual learning for resource-poor languages and were evaluated on monolingual treebanks belonging to 82 unique languages (Zeman et al., 2018). However, none of these multilingual parsers have been evaluated on code-mixed data or adapted specifically for CM parsing.

The Bengali-English code-mixing is found in abundance as Bengali is widely spoken in India and Bangladesh. It is the second most widely spoken language in India after Hindi (Bhatia, 1982). Because of inherent structural and semantic similarity between Bengali and Hindi, we observe a close proximity between Bengali-English and Hindi-English code-mixing as well. Both of these language pairs deal with

the challenges of mixing different typologically diverse languages; SOV word order[1] for Hindi/Bengali and SVO word order for English. A dependency parser for Hindi-English code-mixing has been presented by Bhat et al. (2018). In comparison, Bengali-English code-mixing is left relatively unexplored barring significant works on language identification (Das and Gambäck, 2014) and POS tagging (Jamatia et al., 2015) which serve as preliminary tasks for more advanced parsing applications down the pipeline. The main hindrance to the development of parsing technologies for Bengali-English stem from the lack of annotated resources for the code-mixing of this language pair. In this paper, we try to utilize the pre-existing resources for widely available monolingual Bengali, Hindi and English as well as Hindi-English code-mixing and adapt them for Bengali-English dependency parsing. We also propose a rule based system to synthetically generate Bengali-English code-mixing data. An attempt has been made to generate code-mixing data for the Spanish-English language pair (Pratapa et al., 2018) but none for the Hindi-English or Bengali-English language pair as these pairs pose special challenges due to their different word orders which commonly violate most code-mixing theories (Sinha and Thakur, 2005). We further present a method to project dependency annotations to our Bengali-English CM data from monolingual Bengali and Hindi-English CM treebank and generate a synthetic treebank for Bengali-English (Syn-BE) which helps improve the accuracy of our dependency parser. For evaluation purpose, we present a dataset of 500 Bengali-English tweets annotated under Universal Dependencies scheme.

## 2 Universal Dependencies for Bengali-English

### 2.1 Data Preparation and Annotation

We prepared a dataset of 500 Bengali-English tweets by crawling over Twitter using Tweepy[2] - an API wrapper for Twitter. We identify the Bengali-English tweets by running the tweets through a language identification system (Bhat et al., 2018) trained on the dataset provided by ICON 2015.[3] We select only those tweets which satisfy a minimum code-mixing ratio of 30:70(%). Here, code-mixing ratio is defined as:

$$\frac{1}{n}\sum_{s=1}^{n}\frac{E_s}{M_s+E_s}$$

where $n$ is the number of sentences in the dataset, $M_s$ and $E_s$ are the number of words in the matrix and embedded language in sentence $s$ respectively. Next, we manually select 500 tweets from the resulting tweets and normalize and/or transliterate each word before annotating them using Universal Dependency guidelines (Nivre et al., 2016) for POS and dependency tags. The language tags are annotated based on the tag set defined in (Solorio et al., 2014; Jamatia et al., 2015).

Figure 1 illustrates the conventions followed by our annotators for unique code-mixed constructions. Bengali verbification of English verb *start* by adding a Bengali light verb *hobe* ("will be") leads to a *hybrid* compound verb *start hobe* ("will be"). Here, *start* is POS tagged as 'NOUN' instead of 'VERB' as it functions as a noun in this CM lexical unit and verbal inflection is observed only by the light verb *hobe* ("will be"). Also, #BOSS2 is tagged as 'PROPN' instead of 'X' as it is a syntactic token in this context. These annotations are consistent with the annotations for Hindi-English CM (Bhat et al., 2018).

The resulting dataset is split into three sets consisting of 200 tweets for testing, 160 for tuning and a third set of 140 tweets to be used as the training set in our stacking model for dependency parsing. The Bengali-English CM dataset is available at `https://github.com/urmig/UD_bn-en`.

### 2.2 Code Mixing Data Synthesis

Based on the token-level data distribution in Table 1, we observe that the matrix language in the majority of CM sentences is Bengali. The same is observed for the Hindi-English CM Data (Sharma et al., 2016). With this assumption, we proceed with the synthetic data generation by mixing English linguistic elements into the matrix of Bengali sentences. A frequently observed phenomenon in CM data is replacement of noun phrases in one language by the corresponding noun phrase in the other language

[1] Subject, Object and Verb Order in transitive sentences

[2] http://www.tweepy.org/

[3] http://ltrc.iiit.ac.in/icon2015/

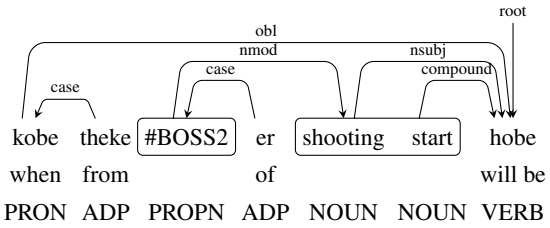


Figure 1: An example to illustrate Bengali-English Code-Mixed tweets

| Language Tags | Token Count | |
|---|---|---|
| Bengali | 2840 | (46.73%) |
| English | 1781 | (29.3%) |
| Rest (univ,acro,ne) | 1457 | (23.97%) |

Table 1: Token-level Data Distribution on 500 Bengali-English tweets.

| Language | POS | UAS | LAS |
|---|---|---|---|
| Hindi | 97.65 | 94.36 | 91.02 |
| Bengali | 93.26 | 87.07 | 80.1 |
| Hindi-English | 91.90 | 74.16 | 64.11 |

Table 2: POS and parsing results of neural-stacking model for different languages

partially or in entirety (Dey and Fung, 2014). Sinha and Thakur (2005) had previously discussed CM constraints for Hindi-English and came to the conclusion that the phenomenon of code-mixing for this language pair is not entirely arbitrary. In our code-mixing method, we will be closely following *the Closed Class Constraint* which states that the matrix language elements within the closed class of grammar (possessives, ordinals, determiners, pronouns) are not allowed in code-mixing (Sridhar and Sridhar, 1980; Joshi, 1982).

(1) ***Bengali:*** *(Apnar "your" PRON) (chokher "of eyes" NOUN) (dekhashonar "care" VERB) (jonye "for" ADP) (aapni "you" PRON) (kotota "how much" DET) (icchuk "willing" ADJ) ?*

(2) ***English:*** *(How ADV) (aware ADJ) (are VERB) (you PRON) (about ADP) (the DET) (care NOUN) (of ADP) (your PRON) (eyes NOUN) ?*

(3) ***Incorrect CS:*** *(*Your PRON) (chokher "of eyes" NOUN) (*about ADP) (*the DET) (dekhashona "care" NOUN) (*you PRON) (*how ADV) (icchuk "willing" ADJ)?*

(4) ***Correct CS:*** *(Apnar "your" PRON) (eyes NOUN) (er "of" ADP) (care NOUN) (er "of" ADP) ( jonno "for" ADP) (aapni "you" PRON) (kotota "how much" DET) (aware ADJ) ?*

Example (3) demonstrates an unnatural and uncommon code-mixed construction and thus we can conclude that the two mixing constraints hold true for Bengali-English CM text as well. We extend these constraints to *question words* which can fall in the POS category of ADV and PRON as well as for *adpositions* (prepositions and postpositions). We note that the example (4) results in an acceptable code-mixed sentence as the closed class elements from the matrix language Bengali are retained.

**The Code-Mixing Process**

The pipeline for our code-mixing script is as shown in Figure 2. The script takes shallow-parsed English and Bengali parallel corpora as inputs. Consistency across chunks in parallel sentences is imperative for direct replacement of chunks for code-mixing. However, there are various structural differences in constituency parsing obtained for English by the Stanford Parser (Klein and Manning, 2003) and shallow parsing obtained for Bengali by the Shallow Parser by TDIL Program, Department Of IT Govt. Of India.[4]. The first module, *chunk harmonizer* handles the issue of structural differences in English and

[4]http://ltrc.iiit.ac.in/analyzer/bengali/

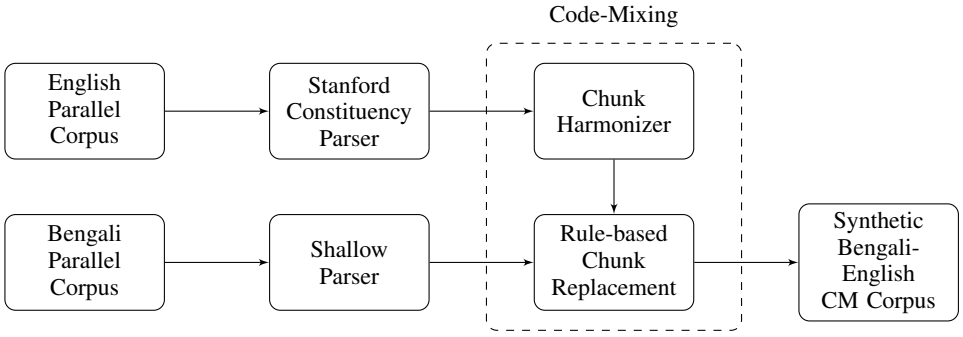


Figure 2: Schematic diagram of the Code-Mixing process

Bengali chunks by modifying the English chunks based on the following set of rules:

1. Separate the *coordinating conjunction* and its conjuncts into different chunks as they are treated separately in Bengali.

2. Combine the *adverbs of degree* (also, too, so, very etc.) with the preceding noun phrase (NP) as they are classified in Bengali as *particles* (*o* ("too"), *i* ("only") etc.) and *intensifiers* (*bhishon* ("extreme"), *khub* ("very") etc.) and grouped with NP.

3. Convert *prepositional phrase* (PP) to NP by making the head noun of the succeeding NP as the head and separating it from the preceding verb phrase (VP).

4. Split NP at *genitives* into separate NPs as genitives are considered as separate chunks in Bengali.

The rules are demonstrated by the example below:

(5) *(NP Your self-confidence) (ADVP also) (VP increases (PP with (NP teeth))) → (NP Your) (NP self-confidence also) (VP increases) (NP with teeth)*

which now consistently maps to the corresponding chunks in the parallel Bengali sentence:

(6) *(NP daanter "teeth" jonyo "for") (NP aapnaar "your") (NP aatmaviswas "self-confidence" o "also") (VP baadhe "increases")*

Along with harmonizing the chunks, this module marks the heads of each chunk in both languages using generalized rules defined by Sharma et al. (2006). For clarity, we have mapped the POS tags from Penn Treebank POS tagsets (Marcus et al., 1993) for English and Bureau Of Indian Standard (BIS) POS tagset (Choudhary and Jha, 2011) for Bengali to the Universal Dependency Tagset (Nivre et al., 2016).

The second module in the pipeline facilitates *rule-based chunk replacement* by taking the chunk-harmonized parallel Bengali and English sentences as inputs and replacing some selected Bengali chunks with English according to the rules discussed in 2.2. First, the chunks, each represented by the head element, are aligned using word alignments obtained from Giza++ (Och and Ney, 2003). Next, we replace the Bengali noun chunks (NP) and adjectival chunks (JJP) with the corresponding English chunks. By keeping the verbal chunks (VP) intact, we ensure that Bengali is retained as the matrix language of the code-mixed sentence. Hybrid compound verbs (see section 2.1) are a common occurrence in Bengali-English code-mixing and we can succesfully synthesize them by replacing the NP/JJP preceding Bengali light verbs. For eg: (JJP *porishkaara* ("clean")) (VP *koruna* ("do")) → (JJP clean) (VP *koruna* ("do")). We also retain Bengali post-positions and drop English prepositions associated with the heads.

Mixing the Bengali sentence (6) with the parallel English sentence (5) will generate:

(7) *(NP teeth er "of" jonyo "for" ) (NP aapnaar "your" ) (NP self-confidence o "also" ) (VP baadhe "increases" )*

This is one of the acceptable combinations of the two sentences to form a CM sentence. We use the parallel corpora for English, Bengali and Hindi provided by Indian Languages Corpora Initiative (ILCI) (Jha, 2010) belonging to the *health* domain. We select a subset of 10,000 parallel sentences from each language and generate code-mixed sentences for both Bengali-English and Hindi-English language pair following the constraints in 2.2 . Thus, we have a parallel corpora for code-mixed Bengali-English and Hindi-English along with parallel corpora for Bengali, Hindi and English. We obtain only 5,063 code-mixed sentences with a minimum CM ratio of 30:70(%). The reason for this is attributed to the non-alignment of a few heads in many Bengali and Hindi sentences to the heads of corresponding English sentence. In spite of strictly following these rules, we generated a few erroneous sentences with word repetitions due to inconsistent chunking of multi-word expressions. We try to mitigate those errors in the post-processing step by carefully removing repeated words at code-mixing points. We attain this by calculating cosine similarity between the words represented by their *cross lingual embeddings* (see section 4). Eg: *chiniyukta* ("sugared") sugared gums → sugared gum

### 2.3 Synthetic Bengali-English Treebank

Cross-lingual annotation projection makes use of parallel data to project annotations from the source language to the target language through automatic word alignment. Hwa et al. (2002) proposed some basic projection heuristics to deal with different kinds of word alignments. Tiedemann (2014) proposed improvements in the annotation scheme by adding heuristics to remove unnecessary dummy nodes that are introduced in the target treebank to deal with problematic word alignments. We investigate the utility of annotation projection from the Hindi-English CM treebank (HE) and the Bengali monolingual treebank (B) to Bengali-English (BE). HE is created by parsing the Hindi-English CM data generated in the section 2.2 using the neural stacking dependency parser for Hindi-English by Bhat et al. (2018).[5] BE is generated by parsing the parallel Bengali sentences using the same neural stacking dependency parser trained on a monolingual Bengali dependency treebank. The POS tagging and parsing accuracy of these two parsers are mentioned in Table 2.

The basic setup for annotation projection is as follows:

1. Project annotations from B to BE for the matching head word nodes in Bengali and its dependent Bengali nodes.

2. Project annotations from HE to BE for the matching head word nodes in English and its dependent English nodes.

3. For each matching English dependent node in HE and BE with a Hindi head, find the aligned Bengali node in B. If the cosine similarity between the two is above a certain threshold (0.5), project annotations from B to BE.

4. For each matching Bengali dependent node in B and BE with an English head, find the aligned Hindi node in HE. If the cosine similarity between the two is above a certain threshold (0.5), project annotations from HE to BE.

In Figure 3, we demonstrate this with an example where the annotation for the BE tree is generated by both HE (in blue) and B (in red). Since the sentences in BE, HE and B are essentially parallel, we get one-to-one mapping and do not need to introduce any dummy nodes. We select 3643 completely annotated trees for our Syn-BE.

## 3 Dependency Parsing

We adapt the neural dependency parser by Bhat et al. (2018) which is based on a transition-based parser (Kiperwasser and Goldberg, 2016) and enhanced by neural stacks to incorporate monolingual syntactic knowledge with the CM model. The model jointly learns POS-tagging as well as parsing by adapting feature level neural stacks (Zhang and Weiss, 2016; Chen et al., 2016). The input layer for both the

[5]https://github.com/irshadbhat/csnlp

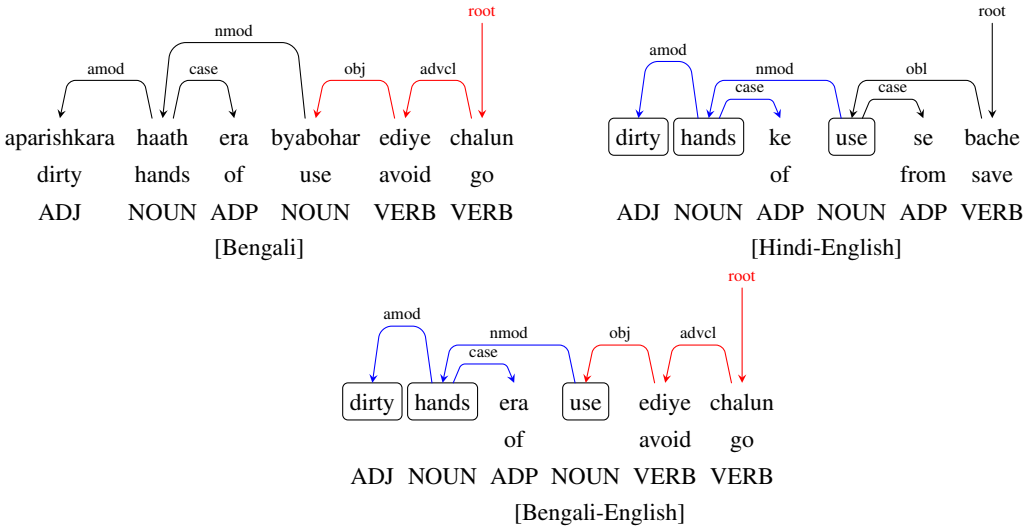


Figure 3: An example of annotation projection from Bengali and Hindi-English to Bengali-English

tagger and the parser encodes the input sentence into word and character embeddings and passes it to the shared bidirectional LSTM (Bi-LSTM). Bhat et al. (2018) demonstrates augmenting the final multi-layer perceptron (MLP) layer of a bilingual model trained on Hindi and English treebanks (bilingual source model) into the MLP layer of the model trained on Hindi-English CM data (CM model) achieves state-of-the-art results for Hindi English code-mixing.

## 4 Experiments

Our models are trained on English and Hindi UD-v2 treebanks.[6] Due to the absence of a Bengali UD treebank, we converted the Paninian annotation scheme (Begum et al., 2008) present in the Bengali treebank[7] to UD by slightly modifying the rules (Tandon et al., 2016) for Hindi. The characters are represented by 32-dimensional character embeddings while the words in each language are represented by 64 dimensional word2vec vectors (Mikolov et al., 2013) learned using the skip-gram model. The hidden dimensions and learning hyperparameters are consistent with those in Bhat et al. (2018).

For our baseline model, we train the neural stacking model (Bhat et al., 2018) for Bengali-English by training the source model on both Bengali and English treebanks and stacking it on a CM model trained on 140 Bengali-English CM (Gold-BE) sentences in our training set. Even though the size of the training set is limited, we benefit from the presence of unique CM grammar as well as syntactic information of social media elements. Our bilingual source model serves to transfer both POS tagging and parsing information to the CM model.

In our next experiment, we train the CM stacking model with 1448 Hindi-English CM data (Gold-HE) as provided by Bhat et al. (2018) in addition to our 140 Gold-BE sentences. In order to fully capture the Hindi syntactic information in the CM data, we fortify the bilingual source model with the Hindi treebank resulting in a trilingual source model. We try to reduce the differences in data representations belonging to Hindi and Bengali by using:

1. *Cross Lingual Word Embeddings* for Hindi and Bengali by projecting the word2vec embeddings for the two languages into the same space by using the projection algorithm of Artetxe et al. (2016) and using a bilingual lexicon from ILCI parallel corpora.

2. *WX notation*[8] to represent words from the two languages and using a common 32-dimensional character embedding space.

[6] https://github.com/UniversalDependencies
[7] Developed as a part of the Indian Languages Treebanking Project by Jadavpur University
[8] http://wiki.apertium.org/wiki/WX_notation

| Embeddings | POS | UAS | LAS |
|---|---|---|---|
| Monolingual | 84.86 | 71.32 | 56.93 |
| Crosslingual | 85.62 | 71.94 | 57.41 |
| Crosslingual + WX notation | 87.43 | 74.42 | 60.04 |

Table 3: Effect of embeddings on POS and Parser results for the Trilingual + Gold-(HE + BE) model

| Stacking Models | POS | UAS | LAS |
|---|---|---|---|
| (Bilingual) + Gold-BE | 79.39 | 62.78 | 49.38 |
| (Trilingual) + Gold-(HE + BE) | 87.43 | 74.42 | 60.04 |
| (Trilingual + Syn-BE) + Gold-(HE + BE) | 89.63 | 76.24 | 61.41 |

Table 4: POS and Parser results of different neural-stacking models for Bengali-English.

For our final experiment, we augment our Synthetic Code-Mixed Bengali-English Treebank (Syn-BE) to the trilingual source model generated in the previous experiment and stack that on our CM model.

## 5 Results

We present our final results in Table 4. The baseline model adapted from Bhat et al. (2018) for Hindi-English gives us 62.78% UAS and 49.38% LAS points. The POS results give 79.39% accuracy. The lower accuracy for the model is expected due to the small training set for Bengali-English (140) when compared with Hindi-English (1448). Moreover, the significantly lower parser accuracy (a difference of ~9% LAS points) for Bengali in comparison to Hindi negatively impacts the performance of the source model (See Table 2).

Our next model that fortifies the baseline model with Hindi monolingual and CM data with Hindi-English improves all the three measurements significantly because it enables us to utilize the relatively large Hindi-English CM UD-annotated data. The UAS and LAS show an improvement in accuracy by 11.64% and 10.66% points respectively. The improvement in POS accuracy is ~8%. In this model, we slightly modify the word and character embedding representations in order to mitigate the lexical differences between Hindi and Bengali by using cross-lingual embeddings and a common character space. From Table 3, we observe that using cross-lingual embeddings improves the accuracy of tagging by 0.76%, UAS by ~0.6% points and LAS by ~0.5% points. Using a common character space by using WX notation further improves the accuracy of both tagging and parsing by ~1.8% and ~2.5% points respectively. The significant improvements in the results confirm the inherent similarity between the code-mixing grammar of Hindi and Bengali with English as both of these language pairs deal with mixing of two typologically diverse languages.

Our final model utilizes our Syn-BE CM treebank by augmenting it to the trilingual source model and stacking it on the CM model trained on our Gold-HE and Gold-BE datasets. We observe an improvement in the Bengali-English parser accuracy by 1.82% UAS points, 1.37% LAS points and POS tagging accuracy by 2.2% . This improvement is satisfactory considering the errors propagated into our Syn-BE treebank by annotating projections from automatically parsed Bengali and Hindi-English treebanks. We must also note that the the domain of Syn-BE (*health*) lacks certain social media elements and constructs present in the evaluation set.

## 6 Conclusion

Our neural stacking model utilizing monolingual, gold and synthetic CM resources has shown significant improvement of 10.24% for POS, 13.76% improvement in UAS and ~12% improvement in LAS points when compared with the baseline model. The stacking model augmented by the Syn-BE CM treebank improves the POS tagging accuracy by 2.2% points and parser accuracy by 1.82% UAS points and 1.37% LAS points respectively. The Syn-BE CM data can be used in other NLP systems like machine translation, question-answering etc. to further improve their systems. There is scope for extending the Syn-BE corpus by including more CM constructions like intra-sentential switching and CM sentences with English as the matrix language. Our evaluation dataset consisting of 500 UD-annotated Bengali-English tweets provides for a valuable resource for research on code-mixing.

## References


Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A Smith. 2016. Many languages, one parser. *Transactions of the Association for Computational Linguistics*, 4:431–444.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2016. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2289–2294.

Utsab Barman, Amitava Das, Joachim Wagner, and Jennifer Foster. 2014. Code mixing: A challenge for language identification in the language of social media. In *Proceedings of the first workshop on computational approaches to code switching*, pages 13–23.

Rafiya Begum, Samar Husain, Arun Dhwaj, Dipti Misra Sharma, Lakshmi Bai, and Rajeev Sangal. 2008. Dependency annotation scheme for indian languages. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*.

Irshad Ahmad Bhat, Riyaz Ahmad Bhat, Manish Shrivastava, and Dipti Misra Sharma. 2018. Universal dependency parsing for hindi-english code-switching. *arXiv preprint arXiv:1804.05868*.

Tej Bhatia and William Ritchie. 2008. *The Handbook of Bilingualism*. 01.

TEJ K. Bhatia. 1982. English and the Vernaculars of India: Contact and Change1. *Applied Linguistics*, III(3):235–245, 10.

Hongshen Chen, Yue Zhang, and Qun Liu. 2016. Neural network for heterogeneous annotations. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 731–741.

Narayan Choudhary and Girish Nath Jha. 2011. Creating multilingual parallel corpora in indian languages. In *Language and Technology Conference*, pages 527–537. Springer.

Amitava Das and Björn Gambäck. 2014. Identifying languages at the word level in code-mixed indian social media text. In *Proceedings of the 11th International Conference on Natural Language Processing*, pages 378–387.

Anik Dey and Pascale Fung. 2014. A hindi-english code-switching corpus. In *LREC*, pages 2410–2413.

Sukanya Dutta, Tista Saha, Somnath Banerjee, and Sudip Kumar Naskar. 2015. Text normalization in code-mixed social media text. In *2015 IEEE 2nd International Conference on Recent Trends in Information Systems (ReTIS)*, pages 378–382. IEEE.

John J. Gumperz. 1982. *Discourse Strategies*. Studies in Interactional Sociolinguistics. Cambridge University Press.

Rebecca Hwa, Philip Resnik, Amy Weinberg, and Okan Kolak. 2002. Evaluating translational correspondence using annotation projection. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 392–399. Association for Computational Linguistics.

Anupam Jamatia, Björn Gambäck, and Amitava Das. 2015. Part-of-speech tagging for code-mixed english-hindi twitter and facebook chat messages. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 239–248.

Girish Nath Jha. 2010. The tdil program and the indian langauge corpora intitiative (ilci). In *LREC*.

Aravind K Joshi. 1982. Processing of sentences with intra-sentential code-switching. In *Proceedings of the 9th conference on Computational linguistics-Volume 1*, pages 145–150. Academia Praha.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.

Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 423–430. Association for Computational Linguistics.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Kovida Nelakuditi, Divya Sai Jitta, and Radhika Mamidi. 2016. Part-of-speech tagging for code mixed english-telugu social media data. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 332–342. Springer.

Dong-Phuong Nguyen and A. Seza Dogruoz. 2013. Word level language identification in online multilingual communication. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 857–862, United States, 10. Association for Computational Linguistics (ACL).

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, may. European Language Resources Association (ELRA).

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Adithya Pratapa, Gayatri Bhat, Monojit Choudhury, Sunayana Sitaram, Sandipan Dandapat, and Kalika Bali. 2018. Language modeling for code-mixing: The role of linguistic theory based synthetic data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1543–1553.

Shruti Rijhwani, Royal Sequiera, Monojit Choudhury, Kalika Bali, and Chandra Shekhar Maddila. 2017. Estimating code-switching on twitter with a novel generalized word-level language detection technique. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1971–1982, Vancouver, Canada, July. Association for Computational Linguistics.

Dipti Misra Sharma, Rajeev Sangal, Lakshmi Bai, Rafiya Begam, and KV Ramakrishnamacharyulu. 2006. Anncorra: Treebanks for indian languages (version–1.9). Technical report, Technical report, Language Technologies Research Center IIIT, Hyderabad, India.

Arnav Sharma, Sakshi Gupta, Raveesh Motlani, Piyush Bansal, Manish Srivastava, Radhika Mamidi, and Dipti M Sharma. 2016. Shallow parsing pipeline for hindi-english code-mixed social media text. *arXiv preprint arXiv:1604.03136*.

R Mahesh K Sinha and Anil Thakur. 2005. Machine translation of bi-lingual hindi-english (hinglish) text. *10th Machine Translation summit (MT Summit X), Phuket, Thailand*, pages 149–156.

Thamar Solorio, Elizabeth Blair, Suraj Maharjan, Steven Bethard, Mona Diab, Mahmoud Ghoneim, Abdelati Hawwari, Fahad AlGhamdi, Julia Hirschberg, Alison Chang, et al. 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.

Shikaripur N Sridhar and Kamal K Sridhar. 1980. The syntax and psycholinguistics of bilingual code mixing. *Canadian Journal of Psychology/Revue canadienne de psychologie*, 34(4):407.

Juhi Tandon, Himani Chaudhry, Riyaz Ahmad Bhat, and Dipti Sharma. 2016. Conversion from paninian karakas to universal dependencies for hindi dependency treebank. In *Proceedings of the 10th Linguistic Annotation Workshop held in conjunction with ACL 2016 (LAW-X 2016)*, pages 141–150.

Jörg Tiedemann. 2014. Rediscovering annotation projection for cross-lingual parser induction. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1854–1864.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, Brussels, Belgium, October. Association for Computational Linguistics.

Yuan Zhang and David Weiss. 2016. Stack-propagation: Improved representation learning for syntax. *arXiv preprint arXiv:1603.06598*.