POS Tagging For Code-Mixed Indian Social Media Text : Systems from IIIT-H for ICON NLP Tools Contest

Arnav Sharma
LTRC, IIIT Hyderabad

Raveesh Motlani
LTRC, IIIT Hyderabad

{arnav.s, raveesh.motlani}@research.iiit.ac.in

Abstract

This paper describes Part-of-Speech (POS) tagging systems submission for *POS Tagging For Code-mixed Indian Social Media Text* contest taking place at *ICON-15*. Our system uses linguistically motivated language specific as well as generic features to train the CRF sequence labeling algorithm. There were three language pairs, namely Hindi-English (Hi-En), Bengali-English (Bn-En) and Tamil-English (Ta-En), for each of which two submissions, constrained and unconstrained, could be made. Our main focus was on systems for Hi-En language pair. We submitted constrained systems for all the three language pairs and unconstrained system only for Hi-En. Our constrained system for Bn-En and Ta-En preformed the best while Hi-En performed third best. Our unconstrained system for Hi-En performed the best with 80.68% accuracy while the second best performed with 77.60%. Our submission was declared overall contest winner.

1 Introduction

Multilingual speakers tend to show code-mixing and code-switching in their day to day language. Code-Mixing is embedding of linguistic units such as phrases, words, morphemes of one language into an utterance of another language whereas code-switching refers to the co-occurrence of speech extracts belonging to two different grammatical system (Gumperz, 1982). Here we use code-mixing to refer to both unless explicitly stated.

The exponentially growing popularity of social media and user created web content is producing huge amounts of textual data. Vyas et al. (2014) noted that the complexity in analyzing these texts stems from non-adherence to a formal grammar, spelling variations, lack of annotated data and inherent conversational nature.

Code mixed content created on social media platforms can be called as code-mixed social media text (CMST). Code-mixing leads to presence of more than one language in the text and its social nature adds all the complexities mentioned above. Additionally, CMST is being generated at an enormous scale and there is a need to create special NLP tools for it as traditional NLP tools are trained on news text and have been shown to perform poorly on social content which ensures they will perform poorly on CMST. One of the most fundamental parts of linguistic pipeline is POS tagging, a basic form of syntactic analysis which has countless applications in NLP.

In this paper, we explore POS tagging of code-mixed Indian social media text using machine learning approaches. The text has English mixed with one of the three Indian languages, Hindi, Bengali and Tamil. We first build a POS tagger using only given data set and derive features from it, namely a *constrained system*. Then, we build a POS tagger using additional resources, namely an *unconstrained system*. For both type of systems we do a grid search over all the parameters of the features and measure our performance with ten fold cross-validation and use the model built using best performing parameter values as our submission. Our main focus was on Hi-En language pair as it was the first language of both the authors. Constrained system was developed by experimentation and analysis on Hi-En language pair. Then

the same system was run on Ta-En and Bn-En.

The rest of this paper is organized as follows: Section(2) discusses background and related work; Section(3) discusses about the data sets given, its metrics and analysis; Section(4) talks about our baseline; Section(5) discusses our constrained systems; Section(6) discusses experiments performed with additional resources result of which includes our unconstrained system; Section(7) presents observations and Section(8) concludes and discusses future work.

2 Background

POS tagging for monolingual data has been researched extensively. Systems have been developed with word level accuracies in the higher nineties. However, POS tagging on CMST is a very nascent research problem.

Solorio and Liu (2008) presented the results on the problem of POS tagging English-Spanish code-switched discourse by using preexisting taggers for both languages. The idea came from the evidence on studies of code-switching on different language pairs, which have shown code-switching to be grammatical according to both languages being switched. They performed experiments using different heuristics to combine POS tag information from existing monolingual taggers. They also explored the use of different language identification methods to select POS tags from the appropriate monolingual tagger. Their best results were achieved by a machine learning approach using features generated by monolingual POS taggers. They achieved the maximum accuracy of 93.48%. However, since their data was manually transcribed from recordings, they did not face the difficulties added by social media text.

Vyas et al. (2014) described their initial efforts to POS tag Hi-En CMST. They addressed the challenges presented by code-mixed nature of the text, transliteration and non-standard spellings, as well as lack of annotated data. They were the first to introduce the problem of transliteration in POS tagging of CMST. Their contributions include formalization of the problem and related challenges in processing Hi-En CMST, creation of annotation dataset and some initial experiments for language identification, transliteration, normalization and POS tagging CMST. A language identification system developed in Gella et al. (2013) was used to identify language of the words and then, using a simple heuristic, chunks of same language were formed. They applied CRF++ based Hi POS tagger for Hi and Twitter POS tagger (Owoputi et al, 2013) for En and map it to Universal tagset (Petrov et al, 2012). With gold language tags and transliteration, they achieved an accuracy on accuracy of 79% whereas without gold language tags their accuracy fell to 65%. Thus highlighting the importance of language identification and transliteration for POS tagging of CMST.

Since CMST is heavily influenced by its social nature, it would be valuable to look at POS taggers built just for social media text. None of these studies mentioned here consider code-mixing. Gimpel et al. (2011) proposed a POS tagger for English tweets. Their contributions was that they developed a POS tagset for Twitter using which they tagged 1,827 tweets. They used a CRF tagger with arbitrary local features in a log-linear model adaption. Their basic feature set included a feature for each word type, presence of digits and hyphens, and features looking at capitalization patterns in the word. The extended feature set also utilized external linguistic resources, the domain specific properties of data and unlabeled in-domain data. They reported an accuracy of 89.95%. Owoputi et al. (2013) proposed an improvement over this original Twitter POS tagger. Their feature set had lexical and unsupervised word clustering features which increased the accuracy from 90% to 93%.

3 Code-Mixed Datasets

CMST have some distinctive properties as compared to other datasets. In this section we have described these various properties and also analyzed the dataset provided to us for the shared-task.

3.1 Hi-En - Metrics

Vyas et al. (2014) used the concept of matrix language. They divided the sentence in a contiguous fragments of words such that each fragment has a unique matrix language. Matrix language governs the grammatical relation between constituents of the utterance. Here, we used a much simpler metric, the most frequent language the sentence to gain an understanding of the training data.

The training dataset comprised of 729 sentences. There were a total of 22 sentences which had no English words and 109 sentences which had no Hindi words. 409 sentences had more Hindi words than English words and 189 sen-

Tag	Frequency	Example
en	6178	experience
hi	5546	pasand
univ	3200	:(
ne	831	Rajasthanis
acro	180	IITB
undef	8	M
mixed	12	maha-backlogger

Table 1: Hi-En Training data language distribution

tences had more English words than Hindi words. The Hi-En data was POS tagged using BIS Tagset and some additional tags coming to a total of 39 tags.

There were a total of 15955 tokens present in the Hi-En dataset. The average length of a sentence was 21.88. Each word in the data was also assigned one of the language labels mentioned below.

- *en*: English
- *hi*: Hindi
- *univ*: Universal
- *ne*: Named Entity
- *acro*: Acronym
- *undef*: Undefined
- *mixed*: Word level code-mixing

Table 1 shows the distribution of these tokens based on their language tag.

Jamatia et al. (2015) introduced the concept of code-mixing index defined as

$$CMI = \begin{cases} 100 \times \left[1 - \frac{max\{w_i\}}{n-u}\right] & : n > u \\ 0 & : n = 0 \end{cases}$$

where w_i is the words tagged with each language tag (en, hi) hence excluding items tagged univ, acro, ne, undef, etc., while including those with language tags and language-based suffix tags) and maxw_i is the number of words of the most prominent language. Thus, for monolingual sentences CMI would be 0. CMI of Hi-En came out to be 14.14 if monolingual sentences are included and 14.22 otherwise.

3.2 Hi-En - Analysis of data

We did a through manual inspection of training data to better understand its nature and possible roadblocks ahead. Some of the problems we identified are properties of any CMST. Below are described the problems some of which we tried to tackle in our systems.

- *Tagging mistakes and inconsistencies*: Tokens like "I'm" and "I've" have been tagged as both *PRP* and *NNV* in similar contexts.
- *Incorrect assignment of language tags*: "He hi" , "machaoo en" , "kaptan en" , "firang en" etc.
- *Script inconsistencies*: Some of the Hindi words are written in Devanagari script, while most of them are written in Roman. Some of the English words have been written using symbols and special characters instead of Roman characters.
- *Language inconsistencies*: Both standard, newspaper like as well as non-standard, day-to-day texting like forms of language is used.
- *Inconsistent pre-processing of data*: Tokenization of sentences has a lot of inconsistencies. Words are merged with numbers, emoticons and punctuation. Most of the sentences were from Facebook posts. Thus they were composed of many small sentences. The fact that the average sentence length in data is quite high (21.88) supports the claim.

4 Problem Definition and Baseline System

Part-of-Speech tagging is the process of assigning a part-of-speech label to each word in a sentence, based on both, its definition as well as its context. A lot of algorithms can used for automating the task of Part-of-Speech tagging. Most of them fall into either of the two categories : Rule-based or Statistical. We have used a statistical sequence labeling algorithm called Conditional Random Fields (Lafferty et al, 2001).

4.1 Baseline

We created our baseline system which gave the most frequent tag of the word as the output. If the word was not present in the training set, it was given the most frequent tag in the training set.

5 Constrained System

Constrained system had the limitation that only the training set provided could be used for building the system. We defined the following set of features.

5.1 Features

- *Lowercase word (Lower)*: In the training data, a lot of same Hindi words were written with different cases. For example, *ishq/Ishq, wala/WaLa/Wala, Aur/aur/AUR*, etc. Capitalization does not give any semantic or syntactic meaning in Hindi. We created a feature that stored the words with all the characters in lowercase.

- *Gold Language Tag (Lang)*: The word level language tag present in the dataset.

- *Affixes*: (prefix_len, suffix_len) Adds prefixes of length ranging from [1, prefix_len] and suffixes of length ranging from [1, suffix_len] as features.

- *Word Context Window (Context)*: (start, end) Adds the words in the context window [start, end] of current word as features. If the window goes ahead from either side, nothing is added.

- *Numeral (Num)*: We define a regular expression which matches a word if the word has some sort of number in it. For example, the regular expression would match 1-4, =3 etc. This binary feature is *True* if the regular expression matches the current else it is *False*.

- *First Character in Uppercase (F Upper)*: Binary feature which is *True* if the first character of the given word is in uppercase else it is *False*.

- *Any Character in Uppercase (A Upper)*: Binary feature which is *True* if any character of the given word is in uppercase else it is *False*.

- *Symbol (Sym)*: Binary feature which is *True* if the given word is not alphanumeric and punctuation else it is *False*.

5.2 Experiments

We used pycrfsuite[1], scikit-learn (Pedregosa et al, 2011) and Amazon web services extensively to perform all our experiments. Our training set had 729 sentences on which we performed ten fold cross validation for all our experiments.

We built models incrementally by adding new features to the previous ones. Finally, we set gave each feature a range containing its possible values and did a ten fold cross validated grid search, using scikit's GridSearchCV, to find the best possible combination of parameters. Binary features like Numeral could either be added to the model or not, thus giving them two possible values. For *Context* we set the range from 0 to 2 (inclusive) for both start and end points of the window, thus giving it a total of 9 possible values. Similarly, for *Affixes* we gave it a range from 0 to 5 (inclusive) for both prefix and suffix length which totals to 36 possible values. We ignored *Lower* and *Lang* from this grid search. Thus, a total of $36 \times 9 \times 2 \times 2 \times 2 \times 2 \times 10 = 51,840$ models were to be trained and evaluated. Since, this was some huge amount of computation, we rented out a 40 core system on AWS to perform this. The best performing model was **M8**.

5.3 Results

Baseline model performed with an accuracy of 60.22% which was more than 10% better than our most basic CRF model which just had *Lower* as feature, **M1**.

M8 had an accuracy of 73.80%, 0.14% better than the best performing model before that - **M3**. The parameters of **M8** are listed in Table 2. We trained **M8** on Bn-En and Ta-En train datasets as well. It performed reasonably well with ten fold cross validated accuracies of 79.35% and 71.09%. Thus, we chose separately trained **M8** for each language as our constrained system and used it to create output of the test datasets provided.

After testing, our constrained system for Hi-En stood third with accuracy 75.04% just behind the submission which stood second by 0.50%. Our Bn-En submission stood first with 79.84% accuracy as did our Ta-En submission with 75.48% accuracy.

6 Unconstrained System

Unconstrained system had no restrictions as constrained system. We were free to utilize different resources and tools. We defined the following set of features on top of the features defined in the constrained section.

[1] https://github.com/jakevdp/pyCRFsuite

	Lower	Lang	Affixes [p, s]	Context	Num	F Upper	A Upper	Symbol	%
B	-	-	-	-	-	-	-	-	60.22
M1	+	-	-	-	-	-	-	-	49.70
M2	+	+	-	-	-	-	-	-	62.88
M3	+	+	[4, 4]	-	-	-	-	-	73.66
M4	+	+	[4, 4]	[2, 2]	-	-	-	-	73.52
M5	+	+	[4, 4]	[2, 2]	+	-	-	-	73.47
M6	+	+	[4, 4]	[2, 2]	+	+	+	-	73.38
M7	+	+	[4, 4]	[2, 2]	+	+	+	+	73.43
M8	+	+	[5, 5]	[1, 2]	-	-	-	+	73.80

Table 2: Constrained Models

6.1 Features

- *Hindi Transliterator and Normalizer (Normalizer)*: Vyas et al. (2014) mention that transliteration and normalization are the major roadblocks in doing POS tagging of CMST. We used python-irtrans[2] developed by Irshad Bhatt for transliteration of Hindi from Roman to Devanagari. We ran the output of transliteration through a spell checker to normalize it, the end product of which we gave as a feature. Though there was some difficulty in transliterating tokens that do not have vowels rk (ruk), sb (sab), rhi (rahi), phr (phir), pr (par) etc.

- *Twitter POS Tag (Twitter POS)*: For English we used the Twitter POS tagger (Owoputi et al, 2013) which is also freely available[3]. The output of the Twitter POS tagger was given as a feature as is. As mentioned before, preprocessing done on training dataset is inconsistent. The tagger is custom built to handle social media text and tackles such inconsistencies rather well.

- *English Hierarchical Word Clusters (Clusters)*: Owoputi et al, (2013) released Hierarchical Word Clusters produced by an unsupervised HMM using Brown clustering implementation on English tweets. We looked up the word in which cluster it belonged and gave its numeric cluster id as a feature.

- *Affixes of Normalized Hindi Words (Affixes-2)*: (prefix_len, suffix_len) Adds prefixes of length ranging from [1, prefix_len] and suffixes of length ranging from [1, suffix_len] of the normalized Hindi word created in the feature *Normalizer*.

[2] https://github.com/irshadbhat/python-irtrans

[3] http://www.ark.cs.cmu.edu/TweetNLP/

6.2 Experiments

We added each feature one by one on the constrained best performing model **M8**.

Like for constrained systems, we built models incrementally by adding features one by one and evaluating it using ten fold cross-validation. Lastly, we did a grid search using scikit's GridSearchCV, to find the best possible set of features along with its parameters. Normalizer, Twitter POS tagger and Word Clusters could either be present or absent whereas Normalized Affixes were given the range of 0 to 6 for both, prefix_len and suffix_len. The best performing model found from the grid search was defined as **M13** and its features with its parameters is described in the Table 3.

6.3 Results

All of the features had a positive impact except for English Hierarchical Word Clusters on addition of which accuracy fell by 0.08%. **M13**, the best model found after grid search had the accuracy of 77.36%. Thus, we chose **M13** as our unconstrained system and used it to tag the testing data.

On evaluation over testing data, **M13** performed with an accuracy of 80.68% which was huge improvement over the next best accuracy of 70.60%.

7 Observations

As we had mentioned earlier about the nature of dataset provided, that there were many instances of lengthy utterances (speeches, complete Facebook posts, long discourses etc.) being treated as a single sentence. We experimented with cleaner data by manually treating it and breaking these

	Normalizer	Twitter POS	Clusters	Affixes-2 [p, s]	%
M9	+	-	-	-	73.90
M10	+	+	-	-	77.08
M11	+	+	+	-	77.02
M12	+	+	+	[5, 5]	77.10
M13	+	+	-	[6, 0]	77.36

Table 3: Unconstrained Models

Tag	**Frequency**
en	8553
hi	411
univ	1837
ne	254
acro	155
undef	1
mixed	1
Total	11212

Table 4: Hi-En Testing data language distribution

lengthy utterances into shorter meaningful sentences (without adding any tokens by ourselves). To our utter surprise the accuracy decreased by 0.8% after we trained and tested our system on the cleaned data.

We observed that adding normalization to English as well as Hindi tokens makes significant difference in the performance of the tagger. There are various reasons why normalization helps the learning of a tagger. Normalization helps in learning in cases where ambiguities may exist. Such as when a word and its noisy form both exist in the data, then normalizing the noisy form to the standard one helps with better learning of both. Another such situation is where the standard form of a noisy word does not exist in the data but words with similar morphological properties and context exist.

We observed that the test data that was provided for evaluation was comparable in size to the training data as shown in Table 4 except for the fact that number of Hi tokens are very low. The average length of a sentence in testing data is 29.74. Usually the testing data is significantly lower as compared to training data. Despite this our model performed well. This robustness of our final model may be due to our evaluation metric being cross-validation, where we train and test the model on various distributions possible over training data.

8 Conclusion and Future Work

Thus for the contest a POS tagger for CMST was created using CRF. The text had sentences composed of a mixture of words in English and one of the three Indian languages, Hindi, Bengali and Tamil. Our main focus was on Hi-En (Hindi-English) language pair as it was first language of both the authors. We first build a POS tagger using only the given dataset, namely a *constrained system*, which gave an accuracy of 75.04% after being evaluated on unseen test dataset. Then, we built a POS tagger using additional resources, namely an *unconstrained system* which performed better than the constrained system and gave an accuracy of 80.68%. We employed ten-fold cross-validation method for training and testing of both type of systems. We also calculated the best model features values by doing a grid search over all the parameters of the features. We also developed and submitted constrained systems for the other two pairs, namely BN-EN (Bengali-English) and TA-EN (Tamil-English). Our accuracy was 79.84% and 75.48% respectively for these language pairs.

In the shared task, our HI-EN constrained system ranked 3^{rd} and unconstrained system ranked 1^{st}. Our constrained system for BN-EN and TA-EN also ranked 1^{st}. Our accuracy of constrained systems for BN-EN and TA-EN was more than the submissions of unconstrained systems for the respective language pairs. Our systems were declared the overall contest winner.

Normalization was one of the most crucial part of our system. In future, we would like to develop a better normalization system without any external dependencies. We would also like to collect and annotate more code-mixed data from various social media platforms and increase our dataset, so that we can get more training data and subsequently a better POS Tagger.

In our literature survey on POS Tagging for social code-mixed data, we had observed that Universal POS Tags (Petrov et al, 2012) were used

for POS Tagging by most of the papers. On the other hand, the datasets provided for this shared task were POS Tagged using BIS Tagset. In the future experiments we would try converting the dataset's POS tags from BIS Tagset to Universal tagset and observe the performance of the tagger.

In this shared task, we were benefited by presence of gold language labels for each token in the dataset. These language labels were manually annotated, which is difficult to obtain if the system is used on a real-time data. Therefore, we need an automatic language labeling system. In future, we plan to build such a system and incorporate it in our POS Tagger, so that we can tag any code-mixed sentence picked up from a social media platform at any time, without depending on a language label annotator.

Acknowledgments

We would like to thank the organizers for organizing this challenging shared task. We would also like to thank Riyaz Bhatt and Irshad Bhatt for helping us and sharing their thoughts, feedback and transliteration system. This publication reflects author's views only.

References

Yogarshi Vyas and Spandana Gella and Jatin Sharma and Kalika Bali and Monojit Choudhury. 2014. POS Tagging of English-Hindi Code-Mixed Social Media Content. *In Proceedings of the First Workshop on Computational Approaches to Code Switching, EMNLP*.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A Universal Part-of-Speech Tagset. *In Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*.

Thamar Solorio and Yang Liu. 2008. Parts-of-speech tagging for English-Spanish code-switched text. *In Proceedings of the Empirical Methods in natural Language Processing*.

Thamar Solorio, Elizabeth Blair, Suraj Maharajan, Steven Bethard, Mona Diab, Mahmoud Gohneim, Abdelati Hawwari, Fahad AlGhamdi, Julia Hirschberg, Alson Chang, and Pascale Fung. 2014. Overview for the first shared task on language identification in code-switched data. *In Proceedings of the First Workshop on Computational Approaches to Code Switching, EMNLP*.

Kalika Bali and Jatin Sharma and Monojit Choudhury and Yogarshi Vyas. 2014. "I am borrowing ya mixing ?" An Analysis of English-Hindi Code Mixing in Facebook. *In Proceedings of the First Workshop on Computational Approaches to Code Switching, EMNLP*.

Olutobi Owoputi, Brendan OConnor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith 2013. Improved part-of-speech tagging for online conversational text with word clusters. *In Proceedings of NAACL.*

Kevin Gimpel, N. Schneider, B. OConnor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. *In Proceedings of ACL.*

Spandana Gella, Jatin Sharma, and Kalika Bali. 2013. Query word labeling and back transliteration for indian languages: Shared task system description. *FIRE Working Notes*.

Anupam Jamatia and Amitava Das. 2014. Part-of-speech tagging system for Hindi social media text on Twitter. *In Proceedings of the First Workshop on Language Technologies for Indian Social Media, ICON*.

Anupam Jamatia, Björn Gambäck and Amitava Das. 2015. Part-of-Speech Tagging for Code-Mixed English-Hindi Twitter and Facebook chat messages. *In the Proceeding of 10th Recent Advances of Natural Language Processing (RANLP)*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *In Proceedings of the Eighteenth International Conference on Machine Learning (ICML '01)*.

John J. Gumperz. 1982. Discourse Strategies.. *Oxford University Press*.

Fabian Pedregosa, Gal Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot and Èdouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research.*