Code-Switching Patterns Can Be an Effective Route to Improve Performance of Downstream NLP Applications: A Case Study of Humour, Sarcasm and Hate Speech Detection

Srijan Bansal[1], **G Vishal**[2], **Ayush Suhane**[3], **Jasabanta Patro**[4], **Animesh Mukherjee**[5]
Indian Institute of Technology, Kharagpur, West Bengal, India - 721302
{[1]srijanbansal97, [2]vishal_g, [3]ayushsuhane99 }@iitkgp.ac.in,
[4]jasabantapatro@iitkgp.ac.in, [5]animeshm@cse.iitkgp.ac.in

Abstract

In this paper we demonstrate how code-switching patterns can be utilised to improve various downstream NLP applications. In particular, we encode different switching features to improve humour, sarcasm and hate speech detection tasks. We believe that this simple linguistic observation can also be potentially helpful in improving other similar NLP applications.

1 Introduction

Code-mixing/switching in social media has become commonplace. Over the past few years, the NLP research community has in fact started to vigorously investigate various properties of such code-switched posts to build downstream applications. The author in (Hidayat, 2012) demonstrated that inter-sentential switching is preferred more than intra-sentential switching by Facebook users. Further while 45% of the switching was done for real lexical needs, 40% was for discussing a particular topic and 5% for content classification. In another study (Dey and Fung, 2014) interviewed Hindi-English bilingual students and reported that 67% of the words were in Hindi and 33% in English. Recently, many down stream applications have been designed for code-mixed text. (Han et al., 2012) attempted to construct a normalisation dictionary offline using the distributional similarity of tokens plus their string edit distance. (Vyas et al., 2014) developed a POS tagging framework for Hindi-English data.

More nuanced applications like humour detection (Khandelwal et al., 2018), sarcasm detection (Swami et al., 2018) and hate speech detection (Bohra et al., 2018) have been targeted for code-switched data in the last two to three years.

1.1 Motivation

The primary motivation for the current work is derived from (Vizcaíno, 2011) where the author notes – “The switch itself may be the object of humour”. In fact, (Siegel, 1995) has studied humour in the Fijian language and notes that when trying to be comical, or convey humour, speakers switch from Fijian to Hindi. Therefore, humour here is produced by the *change of code* rather than by the *referential meaning or content of the message*. The paper also talks about similar phenomena observed in Spanish-English cases.

In a study of English-Hindi code-switching and swearing patterns on social networks (Agarwal et al., 2017), the authors show that when people code-switch, there is a strong preference for swearing in the dominant language. These studies together lead us to hypothesize that the patterns of switching might be useful in building various NLP applications.

1.2 The present work

To corroborate our hypothesis, in this paper, we consider three downstream applications – (i) humour detection (Khandelwal et al., 2018), (ii) sarcasm detection (Swami et al., 2018) and (iii) hate speech detection (Bohra et al., 2018) for Hindi-English code-switched data. We first provide empirical evidence that the switching patterns between native (Hindi) and foreign (English) words distinguish the two classes of the post, i.e., humour vs non-humour or sarcastic vs non-sarcastic or hateful vs non-hateful. We then featurise these patterns and pump them in the state-of-the-art classification models to show the benefits. We obtain a macro-F1 improvement of 2.62%, 1.85% and 3.36% over the baselines on the tasks of humour detection, sarcasm detection and hate speech detection respectively. As a next step, we introduce a modern deep neu-

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 1018–1023
July 5 - 10, 2020. ©2020 Association for Computational Linguistics

ral model (HAN - Hierarchical Attention Network (Yang et al., 2016)) to improve the performance of the models further. Finally, we concatenate the switching features in the last hidden layer of the HAN and pass it to the softmax layer for classification. This final architecture allows us to obtain a macro-F1 improvement of 4.9%, 4.7% and 17.7% over the original baselines on the tasks of humour detection, sarcasm detection and hate speech detection respectively.

2 Dataset

We consider three datasets consisting of Hindi (hi) - English (en) code-mixed tweets scraped from Twitter for our experiments - Humour, Sarcasm and Hate. We discuss the details of each of these datasets below.

	+	-	Tweets	Tokens	Switching*
Humour	1755	1698	3453	9851	2.20
Sarcasm	504	4746	5250	14930	2.13
Hate	1661	2914	4575	10453	4.34

Table 1: Dataset description (* denotes average/tweet).

Humour: Humour dataset was released by (Khandelwal et al., 2018) and has Hindi-English code-mixed tweets from domains like 'sports', 'politics', 'entertainment' etc. The dataset has uniform distribution of tweets in each category to yield better supervised classification results (see Table 1) as described by (Du et al., 2014). Here the positive class refers to humorous tweets while the negative class corresponds to non-humorous tweet. Some representative examples from the data showing the point of switch corresponding to the start and the end of the humour component.

- women can crib on things like $\underline{humour_{start}}$ bhaiyya ye shakkar bahot zyada meethi hai $\underline{humour_{end}}$, koi aur quality dikhao[1]
- shashi kapoor trending on mothersday how apt, $\underline{humour_{start}}$ mere paas ma hai $\underline{humour_{end}}$[2]
- political journey of kejriwal, from $\underline{humour_{start}}$ mujhe chahiye swaraj $\underline{humour_{end}}$ to $\underline{humour_{start}}$ mujhe chahiye laluraj $\underline{humour_{end}}$[3]

Sarcasm: Sarcasm dataset released by (Swami et al., 2018) contains tweets that have hashtags #sarcasm and #irony. Authors used other keywords such as 'bollywood', 'cricket' and 'politics' to collect sarcastic tweets from these domains. In this case, the dataset is heavily unbalanced (see Table 1). Here the positive class refers to sarcastic tweets and the negative class means non-sarcastic tweets. Some representative examples from our data showing the point where the sarcasm starts and ends.

- said aib filthy pandit ji, $\underline{sarcasm_{start}}$ aap jo bol rahe ho woh kya shuddh sanskrit hai $\underline{sarcasm_{end}}$? irony shameonyou[4]
- irony bappi lahiri sings $\underline{sarcasm_{start}}$ sona nahi chandi nahi yaar toh mila arre pyaar kar le $\underline{sarcasm_{end}}$[5]

Hate speech: (Bohra et al., 2018) created the corpus using the tweets posted online in the last five years which have a good propensity to contain hate speech (see Table 1). Authors mined tweets by selecting certain hashtags and keywords from 'politics', 'public protests', 'riots' etc. The positive class refers to a hateful tweets while the negative class means non-hateful tweets[6]. An example of hate tweet showing the point of switch corresponding to the start and the end of the hate component.

- I hate my university, $\underline{hate_{start}}$ koi us jagah ko aag laga dey $\underline{hate_{end}}$ [7].

3 Switching features

In this section, we outline the key contribution of this work. In particular, we identify how patterns of switching correlate with the tweet text being humorous, sarcastic or hateful. We outline a synopsis of our investigation below.

3.1 Switching and NLP tasks

In this section, we identify how switching behavior is related to the three NLP tasks at our

[1] Gloss: women can crib on things like *brother the sugar is a little more sweet, show a different quality*.

[2] Gloss: shashi kapoor trending on mothersday how apt, *I have my mother with me*.

[3] Gloss: political journey of kejriwal, from *I want swaraj* to *I want laluraj*.

[4] Gloss: said aib filthy pandit ji, *whatever you are telling is it pure sanskrit*? irony shameonyou.

[5] irony bappi lahiri sings *Gloss: doesn't matter you do not get gold or silver, you have got a friend to love*.

[6] The dataset released by this paper only had the hate/non-hate tags for each tweet. However, the language tag for each word required for our experiments was not available. Two of the authors independently language tagged the data and obtained an agreement of 98.1%. While language tagging, we noted that the dataset is a mixed bag including hate speech, offensive and abusive tweets which have already been shown to be different in earlier works (Waseem et al., 2017). However, this was the only Hindi-English code-mixed hate speech dataset available.

[7] Gloss: I hate my university. *Someone burn that place*.

hand. Let $\mathcal{Q}$ be the property that a sentence has en words which are surrounded by hi words, that is there exists an English word in a Hindi context. For instance, the tweet *koi*_hi *to*_hi *pray*_en *karo*_hi *mere*_hi *liye*_hi *bhi*_hi satisfies the property $\mathcal{Q}$. However, *bumrah*_hi *dono*_hi *wicketo*_hi *ke*_hi *beech*_hi *gumrah*_hi *ho*_hi *gaya*_hi does not satisfy $\mathcal{Q}$.

We performed a statistical analysis to determine the correlation between the switching patterns and a classification task at hand (represented by $\mathcal{T}$). Let us denote the probability that a tweet belongs to a positive class for a task $\mathcal{T}$ given that it satisfies property $\mathcal{Q}$ by $p(\mathcal{T}|\mathcal{Q})$. Similarly, let $p(\mathcal{T}| \sim \mathcal{Q})$ be the probability that the tweet belongs to the positive class for task $\mathcal{T}$ and does not satisfy the property $\mathcal{Q}$.

Further let $avg(\mathcal{S}|\mathcal{T})$ be the average switching in positive samples for the task $\mathcal{T}$ and $avg(\mathcal{S}| \sim \mathcal{T})$ denote the average switching in negative samples for the task $\mathcal{T}$.

	$\mathcal{T}$: Humour	$\mathcal{T}$: Sarcasm	$\mathcal{T}$: Hate
$p(\mathcal{T}\|\mathcal{Q})$	0.56	0.28	0.36
$p(\mathcal{T}\| \sim \mathcal{Q})$	0.50	0.42	0.41
$avg(\mathcal{S}\|\mathcal{T})$	7.84	0.60	1.49
$avg(\mathcal{S}\| \sim \mathcal{T})$	6.50	0.89	1.54

Table 2: Correlation of switching with different classification tasks.

The main observations from this analysis for the three tasks – humour, sarcasm and hate are noted in Table 2. For the humour task, $p(humour|\mathcal{Q})$ dominates over $p(humour| \sim \mathcal{Q})$. Further the average number of switching for the positive samples in the humour task is larger than the average number of switching for the negative samples. Finally, we observe a positive Pearson's correlation coefficient of 0.04 between a text being humorous and the text having the property $\mathcal{Q}$. This together indicates that the switching behavior has a positive connection with a tweet being humorous.

On the other hand $p(sarcasm| \sim \mathcal{Q})$ as well as $p(hate| \sim \mathcal{Q})$ respectively dominate over $p(sarcasm|\mathcal{Q})$ and $p(hate|\mathcal{Q})$. Moreover the average number of switching for the negative samples for both these tasks is larger than the average number of switching for the positive samples. The Pearson's correlation between a text being sarcastic (hateful) and the text having the property $\mathcal{Q}$ is negative: -0.17 (-0.04). This shows there is an overall negative connection between the switching behavior and sarcasm/hate speech detection tasks.

Feature name	Description
en_hi_switches	The number of en to hi switches in a sentence
hi_en_switches	The number of hi to en switches in a sentence
V	The total number of switches in a sentence
fraction_en	Fraction of English words in a sentence
fraction_hi	Fraction of Hindi words in a sentence
mean hi_en	Mean of hi_en vector
stddev hi_en	Standard deviation of hi_en vector
mean en_hi	Mean of en_hi vector
stddev en_hi	Standard deviation of en_hi vector

Table 3: Description of the switching features.

While we have tested on one language pair (Hindi-English), our hypothesis is generic and has been already noted by linguists earlier (Vizcaíno, 2011).

3.2 Construction of the feature vector

Motivated by the observations in the previous section we construct a vector **hi_en**$[i]$ that denotes the number of Hindi (hi) words before the i^{th} English (en) word and a vector **en_hi**$[i]$ that denotes the number of English (en) words before the i^{th} Hindi (hi) word. This can also be interpreted as the run-lengths of the Hindi and the English words in the code-mixed tweets. Based on these vectors we define nine different features that capture the switching patterns in the code-mixed tweets[8].

An example feature vector computation: Consider the sentence - *koi*_hi *to*_hi *pray*_en *karo*_hi *mere*_hi *liye*_hi *bhi*_hi.

hi_en : $[0, 0, 2, 0, 0, 0, 0]$

en_hi : $[0, 0, 0, 1, 1, 1, 1]$

Feature vector: $[1, 1, 2, \frac{1}{7}, \frac{6}{7}, \frac{2}{7}, 0.69, \frac{4}{7}, 0.49]$

4 Experiments

4.1 Pre-processing

Tweets are tokenized and punctuation marks are removed. All the hashtags, mentions and urls are stored and converted to string 'hashtag', 'mention' and 'url' to capture the general semantics of the tweet. Camel-case hashtags were segregated and included in the tokenized tweets (see (Belainine et al., 2016), (Khandelwal et al., 2017)). For example, *#AadabArzHai* can be decomposed into three distinct words: *Aadab*, *Arz* and *Hai*. We use the same pre-processing for all the results presented in this paper.

[8] We tried with different other variants but empirically observe that these nine features already subsumes all the necessary distinguishing qualities.

4.2 Machine learning baselines

Humour baseline (Khandelwal et al., 2018): Uses features such as n-grams, bag-of-words, common words and hashtags to train the standard machine learning models such as SVM and Random-Forest. The authors used character n-grams, as previous work shows that this feature is very efficient in classifying text because they do not require expensive text pre-processing techniques like tokenization, stemming and stop words removal. They are also language independent and can be used in code-mixed texts. In their paper, the authors report the results for tri-grams.

Sarcasm baseline (Swami et al., 2018): This model also uses a combination of word n-grams, character n-grams, presence or absence of certain emoticons and sarcasm indicative tokens as features. A sarcasm indicative score is computed and chi-squared feature reduction is used to take the top 500 most relevant words. These were incorporated into features used for classification. Standard off-the-shelf machine learning models like SVM and Random Forest were used.

Hate baseline (Bohra et al., 2018): The hate speech detection baseline also consists of similar features such as character n-grams, word n-grams, negation words [9] and a lexicon of hate indicative tokens. Chi-squared feature reduction method was used to decrease the dimensionality of the features. Once again SVM and Random Forest based classifiers were used for this task.

Switching features: We plug in the nine switching features introduced in the previous section to the three baseline models for humour, sarcasm and hate speech detection.

4.3 Deep learning architecture

In order to draw the benefits of the modern deep learning machinery, we build an end-to-end model for the three tasks at hand. We use the Hierarchical Attention Network (HAN) (Yang et al., 2016) which is one of the state-of-the-art models for text and document classification. It can represent sentences in different levels of granularity by stacking recurrent neural networks on character, word and sentence level by attending over the words which are informative. We use the GRU implementation of HAN to encode the text representation for all the three tasks.

Handling data imbalance by sub-sampling: Since the sarcasm dataset is heavily unbalanced we sub-sampled the data to balance the classes. To this purpose, we categorise the negative samples into those that are easy or hard to classify. Hypothesizing that if a model can predict the hard samples reliably it can do the same with the easy samples. We trained a classifier model on the training dataset and obtained the softmax score which represents $p(sarcastic|text)$ for the test samples. Those test samples which have a score less than a very low confidence score (say 0.001) are removed imagining them to be easy samples. The dataset thus got reduced and more balanced. It is important to note that positive samples are never removed. We validated this hypothesis through the test set. Our trained HAN model achieves an accuracy of 94.4% in classifying the easy (thrown out) samples as non-sarcastic thus justifying the sub-sampling.

Switching features: We include the switching features to the pre-final fully-connected layer of HAN to observe if this harnesses additional benefits (see Figure 1).

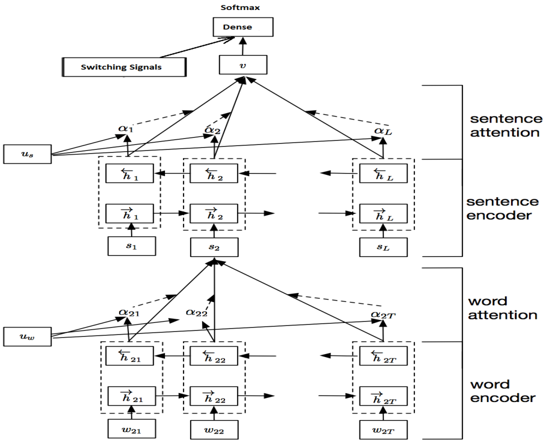

Figure 1: The overall HAN architecture along with the switching features in the final layer.

4.4 Experimental Setup

Train-test split: For all datasets, we maintain a train-test split of 0.8 - 0.2 and perform 10-fold cross validation.

Parameters of the HAN: BiLSTMs: no dropout, early stopping patience: 15, optimizer = 'adam' (learning rate = 0.001, beta_1 = 0.9), loss = binary cross entropy, epochs = 200, batch_size = 32, pre-trained word-embedding size = 50, hidden size: $[20, 60]$, dense output size (before concatenation):

[9] see Christopher Pott's sentiment tutorial: http://sentiment.christopherpotts.net/lingstruc.html

Model	Humour	Sarcasm	Hate
Baseline (B)	69.34	78.4	33.60
Baseline + Feature (BF)	**71.16**	**79.85**	**34.73**
HAN (H)	72.04	81.36	38.78
HAN + Feature (HF)	**72.71**	**82.07**	**39.54**

Table 4: Summary of the results from different models in terms of macro-F1 scores. M-W U test shows all improvements of HF over B are significant.

[15, 30].

Pre-trained embeddings: We obtained pre-trained embeddings by training GloVe from scratch using the large code-mixed dataset (725173 tweets) released by (Patro et al., 2017) plus all the tweets (13278) in our three datasets.

5 Results

We compare the baseline models along with (i) the baseline + switching feature-based models and (ii) the HAN models. We use macro-F1 score for comparison all through. The main results are summarized in Table 4. The interesting observations that one can make from these results are – (i) inclusion of the switching features always improves the overall performance of any model (machine learning or deep learning) for all the three tasks, (ii) the deep learning models are always better than the machine learning models. Inclusion of switching features into the machine learning models (indicated as BF in Table 4) allows us to obtain a macro-F1 improvement of 2.62%, 1.85% and 3.36% over the baselines (indicated as B in Table 4) on the tasks of humour detection, sarcasm detection and hate speech detection respectively. Inclusion of the switching feature in the HAN model (indicated as HF in Table 4) allows us to obtain a macro-F1 improvement of 4.9%, 4.7% and 17.7% over the original baselines (indicated as B in Table 4) on the tasks of humour detection, sarcasm detection and hate speech detection respectively.

Success of our model: Success of our approach is evident from the following examples. For instance, as we had demonstrated earlier, humour is positively correlated with switching, a tweet having a switching pattern like - *anurag*_hi *kashyap*_hi *can*_en *never*_en *join*_en *aap*_hi *because*_en *ministers*_en *took*_en *oath*_en, "*main*_hi *kisi*_hi *anurag*_hi *aur*_hi *dwesh*_hi *ke*_hi *bina*_hi *kaam*_hi *karunga*_hi" which was not detected as humorous by the baseline (B) but was detected so by our models (BF and HF). Note that the author of the above tweet seems to have categorically switched to Hindi to express the humour; such observations have also been made in (Rudra et al., 2016) where opinion expression was cited as a reason for switching.

Sarcasm being negatively correlated with switching, a tweet without having switching is more likely to be sarcastic. For instance, the tweet *naadaan*_hi *baalak*_hi *kalyug*_hi *ka*_hi *vardaan*_hi *hai*_hi *ye*_hi, which bears no switching was labeled non-sarcastic by the baseline. Our models (BF and HF) have rectified it and correctly detected it as sarcastic.

Similarly, hate being negatively correlated with switching, a tweet with no switching - *shilpa*_hi *ji*_hi *aap*_hi *ravidubey*_hi *jaise*_hi *tuchho*_hi *ko*_hi *jawab*_hi *mat*_hi *dijiye*_hi *ye*_hi *log*_hi *aap*_hi *ke*_hi *sath*_hi *kabhi*_hi *nahi*_hi was labeled as non-hateful by the baseline, was detected as hateful by our methods (BF and HF).

6 Conclusion

In this paper, we identified how switching patterns can be effective in improving three different NLP applications. We present a set of nine features that improve upon the state-of-the-art baselines. In addition, we exploit the modern deep learning machinery to improve the performance further. Finally, this model can be improved further by pumping the switching features in the final layer of the deep network.

In future, we would like to extend this work for other language pairs. For instance, we have seen examples of such switching in English-Spanish[10] and English-Telugu[11] pairs also. Further we plan to investigate other NLP applications that can benefit from the simple linguistic features introduced here.

References

P. Agarwal, A. Sharma, J. Grover, M. Sikka, K. Rudra, and M. Choudhury. 2017. I may talk in english but gaali toh hindi mein hi denge : A study of english-hindi code-switching and swearing pattern on social networks. In *2017 9th International Conference on Communication Systems and Networks (COMSNETS)*, pages 554–557.

[10] Don't forget $\underline{humour_{start}}$ *la toalla cuando go to le playa* $\underline{humour_{end}}$; Gloss: Don't forget the towel when you go to the beach.

[11] Votes kosam quality leni liquor bottles supply chesina politicians $\underline{sarcasm_{start}}$ *ee roju quality gurinchi matladuthunnaru* $\underline{sarcasm_{end}}$; Gloss:Politicians who supplied low quality liquor bottles for votes are talking about quality today.

Billal Belainine, Alexsandro Fonseca, and Fatiha Sadat. 2016. Named entity recognition and hashtag decomposition to improve the classification of tweets. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 102–111, Osaka, Japan. The COLING 2016 Organizing Committee.

Aditya Bohra, Deepanshu Vijay, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. A dataset of Hindi-English code-mixed social media text for hate speech detection. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, pages 36–41, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Anik Dey and Pascale Fung. 2014. A Hindi-English code-switching corpus. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*.

Mian Du, Matthew Pierce, Lidia Pivovarova, and Roman Yangarber. 2014. Supervised classification using balanced training. pages 147–158.

Bo Han, Paul Cook, and Timothy Baldwin. 2012. Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 421–432, Jeju Island, Korea. Association for Computational Linguistics.

Taofik Hidayat. 2012. An analysis of code switching used by facebookers (a case study in a social network site).

Ankush Khandelwal, Sahil Swami, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2017. Classification of spanish election tweets (COSET) 2017 : Classifying tweets using character and word level features. In *Proceedings of the Second Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2017) co-located with 33th Conference of the Spanish Society for Natural Language Processing (SEPLN 2017), Murcia, Spain, September 19, 2017*, pages 49–54.

Ankush Khandelwal, Sahil Swami, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. Humor detection in english-hindi code-mixed social media content : Corpus and baseline system. *CoRR*, abs/1806.05513.

Jasabanta Patro, Bidisha Samanta, Saurabh Singh, Abhipsa Basu, Prithwish Mukherjee, Monojit Choudhury, and Animesh Mukherjee. 2017. All that is English may be Hindi: Enhancing language identification through automatic ranking of the likeliness of word borrowing in social media. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2264–2274, Copenhagen, Denmark. Association for Computational Linguistics.

Koustav Rudra, Shruti Rijhwani, Rafiya Begum, Kalika Bali, Monojit Choudhury, and Niloy Ganguly. 2016. Understanding language preference for expression of opinion and sentiment: What do hindi-english speakers do on twitter? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1131–1141.

Jeff Siegel. 1995. How to get a laugh in fijian: Code-switching and humor. *Language in Society*, 24(1):95–110.

Sahil Swami, Ankush Khandelwal, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. A corpus of english-hindi code-mixed tweets for sarcasm detection. *CoRR*, abs/1805.11869.

María José García Vizcaíno. 2011. Association humor in code-mixed airline advertising.

Yogarshi Vyas, Spandana Gella, Kalika Bali, and Monojit Choudhury. 2014. Pos tagging of english-hindi code-mixed social media content. pages 974–979.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding abuse: A typology of abusive language detection subtasks. In *Proceedings of the First Workshop on Abusive Language Online*, pages 78–84, Vancouver, BC, Canada. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.