De-Mixing Sentiment from Code-Mixed Text

Yash Kumar Lal[*†] **Vaibhav Kumar**[*‡] **Mrinal Dhar**[*∘]
Manish Shrivastava[◇] **Philipp Koehn**[†]
[†]Johns Hopkins University, [‡]Carnegie Mellon University, [∘]Bloomberg LP,
[◇]International Institute of Information Technology
{yash, phi}@jhu.edu,
vaibhav2@andrew.cmu.edu, mrinaldhar@gmail.com,
manish.shrivastava@iiit.ac.in

Abstract

Code-mixing is the phenomenon of mixing the vocabulary and syntax of multiple languages in the same sentence. It is an increasingly common occurrence in today's multilingual society and poses a big challenge when encountered in different downstream tasks. In this paper, we present a hybrid architecture for the task of Sentiment Analysis of English-Hindi code-mixed data. Our method consists of three components, each seeking to alleviate different issues. We first generate subword level representations for the sentences using a CNN architecture. The generated representations are used as inputs to a Dual Encoder Network which consists of two different BiLSTMs - the Collective and Specific Encoder. The Collective Encoder captures the overall sentiment of the sentence, while the Specific Encoder utilizes an attention mechanism in order to focus on individual sentiment-bearing sub-words. This, combined with a Feature Network consisting of orthographic features and specially trained word embeddings, achieves state-of-the-art results - 83.54% accuracy and 0.827 F1 score - on a benchmark dataset.

1 Introduction

Sentiment Analysis (SA) is crucial in tasks like user modeling, curating online trends, running political campaigns and opinion mining. The majority of this information comes from social media websites such as Facebook and Twitter. A large number of Indian users on such websites can speak both English and Hindi with bilingual proficiency. Consequently, English-Hindi code-mixed content has become ubiquitous on the Internet, creating the need to process this form of natural language.

Code-mixing is defined as "the embedding of linguistic units such as phrases, words and morphemes of one language into an utterance of another language" (Myers-Scotton, 1993). Typically, a code-mixed sentence retains the underlying grammar and script of one of the languages it is comprised of.

Due to the lack of a formal grammar for a code-mixed hybrid language, traditional approaches don't work well on such data. The spelling variations of the same words according to different writers and scripts increase the issues faced by traditional SA systems. To alleviate this issue, we introduce the first component of our model - it uses CNNs to generate subword representations to provide increased robustness to informal peculiarities of code-mixed data. These representations are learned over the code-mixed corpus.

Now, let's consider the expression: *x* **but** *y*, where *x* and *y* are phrases holding opposite sentiments. However, the overall expression has a positive sentiment. Standard LSTMs work well in classifying the individual phrases but fail to effectively combine individual emotions in such compound sentences (Socher et al., 2013). To address this issue, we introduce a second component into our model which we call the Dual Encoder Network. This network consists of two parallel BiLSTMs, which we call the Collective and Specific Encoder. The Collective Encoder takes note of the overall sentiment of the sentence and hence, works well on phrases individually. On the other hand, the Specific Encoder utilizes an attention mechanism which focuses on individual sentiment bearing units. This helps in choosing the correct sentiment when presented with a mixture of sentiments, as in the expression above.

Additionally, we introduce a novel component, which we call the Feature Network. It uses surface features as well as monolingual sentence vector representations. This helps our entire system work well, even when presented with a lower amount of

[*] Equal Contribution

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, pages 371–377
Florence, Italy, July 28 - August 2, 2019.

training examples as compared to established approaches.

We perform extensive experiments to evaluate the effectiveness of this system in comparison to previously proposed approaches and find that our method outperforms all of them for English-Hindi code-mixed sentiment analysis, reporting an accuracy of 83.54% and F1 score of 0.827. An ablation of the model also shows the effectiveness of the individual components.

2 Related Work

Mohammad et al. (2013) employ surface-form and semantic features to detect sentiments in tweets and text messages using SVMs. Keeping in mind a lack of computational resources, Giatsoglou et al. (2017) came up with a hybrid framework to exploit lexicons (polarized and emotional words) as well as different word embedding approaches in a polarity classification model. Ensembling simple word embedding based models with surface-form classifiers has also yielded slight improvements (Araque et al., 2017).

Extending standard NLP tasks to code-mixed data has presented peculiar challenges. Methods for POS tagging of code-mixed data obtained from online social media such as Facebook and Twitter has been attempted (Vyas et al., 2014) . Shallow parsing of code-mixed data curated from social media has also been tried (Sharma et al., 2016). Work has also been done to support word level identification of languages in code-mixed text (Chittaranjan et al., 2014).

Sharma et al. (2015) tried an approach based on lexicon lookup for text normalization and sentiment analysis of code-mixed data. Pravalika et al. (2017) used a lexicon lookup approach to perform domain specific sentiment analysis. Other attempts include using sub-word level compositions with LSTMs to capture sentiment at morpheme level (Joshi et al., 2016), or using contrastive learning with Siamese networks to map code-mixed and standard language text to a common sentiment space (Choudhary et al., 2018).

3 Model Architecture

An overview of this architecture can be found in Figure 3. Our approach is built on three components. The first generates sub-word level representations for words using Convolutional Neural Networks (CNNs). This produces units that are more atomic than words, which serve as inputs for a sequential model.

In Section 3.2, we describe our second component, a dual encoder network made of two Bidirectional Long Short Term Memory (BiLSTM) Networks that: (1) captures the overall sentiment information of a sentence, and (2) selects the more important sentiment-bearing parts of the sentence in a differential manner.

Finally, we introduce a Feature Network, in Section 3.3, built on surface features and a monolingual vector representation of the sentence. It augments our base neural network to boost classification accuracy for the task.

3.1 Generating Subword Representations

Word embeddings are now commonplace but are generally trained for one language. They are not ideal for code-mixed data given the transcription of one script into another, and spelling variations in social media data. As a single character does not inherently provide any semantic information that can be used for our purpose, we dismiss character-level feature representations as a possible choice of embeddings.

Keeping in mind the fact that code-mixed data has innumerable inconsistencies, we use characters to generate subword embeddings (Joshi et al., 2016). This increases the robustness of the model, which is important for noisy social media data. Intermediate sub-word feature representations are learned by filters during the convolution operation.

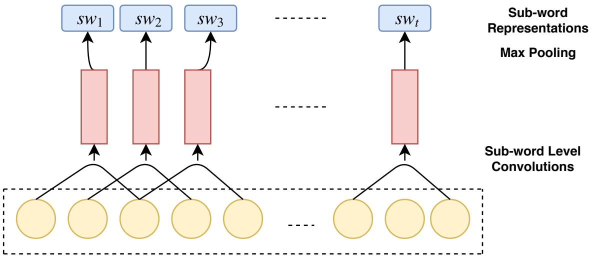


Figure 1: CNNs for Generating Sub-Word Representations

Let S be the representation of the sentence. We generate the required embedding by passing the characters of a sentence individually into 3 layer 1-D CNN. We perform a convolution of S with a filter F of length m, before adding a bias b and applying a non-linear activation function g. Each such operation generates a sub-word level feature map f_{sw}.

$$f_{sw}[i] = g((S[:, i : i + m - 1] * F) + b) \quad (1)$$

Here, $S[:, i : i + m - 1]$ is a matrix of $(i)^{th}$ to $(i + m - 1)^{th}$ character embeddings and g is the ReLU activation function. Thereafter, a final max-pooling operation over p feature maps generates a representation for individual subwords:

$$sw_i = max(f_{sw}[p * (i : i + p - 1)]) \quad (2)$$

A graphical representation of the architecture can be seen in Figure 1

3.2 Dual Encoder

We utilize a combination of two different encoders in our model.

3.2.1 Collective Encoder

The collective encoder, built over a BiLSTM, aims to learn a representation for the overall sentiment of the sentence. A graphical representation of this encoder is in Figure 2(A). It takes as input the subword representation of the sentence. The last hidden state of the BiLSTM i.e. h_t, encapsulates the overall summary of the sentence's sentiment, which is denoted by cmr^c.

3.2.2 Specific Encoder

The specific encoder is similar to the collective encoder, in that it takes as input a subword representation of the sentence and is built over LSTMs, with one caveat - an affixed attention mechanism. This allows for selection of subwords which contribute the most towards the sentiment of the input text. It can be seen in Figure 2(B).

Identifying which subwords play a significant role in deciding the sentiment is crucial. The specific encoder generates a context vector cmr^s to this end. We first concatenate the forward and backward states to obtain a combined annotation $(k_1, k_2, \ldots, k_t)$. Taking inspiration from (Yang et al., 2016), we quantify the significance of a sub-word by measuring the similarity of an additional hidden representation u_i of each sub-word against a sub-word level context vector X. Then, a normalized significance weight α_i is obtained.

$$u_i = tanh(W_i k_i + b_i) \quad (3)$$

$$\alpha_i = \frac{exp(u_i^T X)}{\sum exp(u_i^T X)} \quad (4)$$

The context vector X can be looked at as a high-level representation of the question ”is it a significant sentiment-bearing unit” evaluated across the sub-words. It is initialized randomly and learned jointly during training. Finally, we calculate a vector cmr^s as a weighted sum of the sub-word annotations.

$$cmr^s = \sum \alpha_i k_i \quad (5)$$

Using such a mechanism helps our model to adaptively select the more important sub-words from the less important ones.

3.2.3 Fusion of the Encoders

We concatenate the outputs obtained from both these encoders and use it as inputs to further fully connected layers. Information obtained from both the encoders is utilized to come up with a unified representation of sentiment present in a sentence,

$$cmr^{sent} = [cmr^c; cmr^s] \quad (6)$$

where cmr^{sent} represents the unified representation of the sentiment. A representation of the same can be found in Figure 3.

3.3 Feature Network

We also use linguistic features to augment the neural network framework of our model.

- Capital words: Number of words that have only capital letters

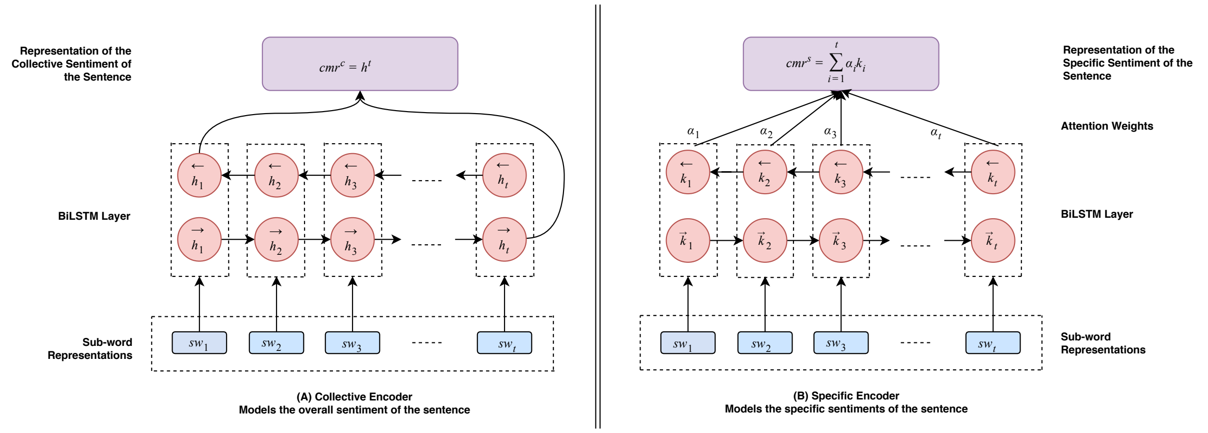
- Extended words: Number of words with multiple contiguous repeating characters.

- Aggregate positive and negative sentiments: Using SentiWordNet (Esuli and Sebastiani, 2006) for every word bar articles and conjunctions, and combining the sentiment polarity values into net positive aggregate and net negative aggregate features.

- Repeated exclamations and other punctuation: Number of sets of two or more contiguous punctuation.

- Exclamation at end of sentence: Boolean value.

- Monolingual Sentence Vectors: Bojanowski et al. (2017)'s method is used to train word vectors for Hindi words in the code-mixed sentences.


Representation of the Collective Sentiment of the Sentence
$cmr^c = h^t$
BiLSTM Layer
$\overleftarrow{h}_1$ $\overleftarrow{h}_2$ $\overleftarrow{h}_3$ $\overleftarrow{h}_t$
$\overrightarrow{h}_1$ $\overrightarrow{h}_2$ $\overrightarrow{h}_3$ $\overrightarrow{h}_t$
Sub-word Representations
sw_1 sw_2 sw_3 sw_t
$cmr^s = \sum_{i=1}^{t} \alpha_i k_i$
Representation of the Specific Sentiment of the Sentence
Attention Weights
α_1 α_2 α_3 α_t
$\overleftarrow{k}_1$ $\overleftarrow{k}_2$ $\overleftarrow{k}_3$ $\overleftarrow{k}_t$
$\vec{k}_1$ $\vec{k}_2$ $\vec{k}_3$ $\vec{k}_t$
BiLSTM Layer
Sub-word Representations

(A) Collective Encoder
Models the overall sentiment of the sentence

(B) Specific Encoder
Models the specific sentiments of the sentence

Figure 2: Parts of the Dual Encoder Network

Method	Accuracy	F1-score
SVM (Uni+Bigram) (Pang and Lee, 2008)	52.96%	0.3773
NBSVM (Uni+Bigram) (Wang and Manning, 2012)	62.5%	0.5375
MNB (Uni+Bigram) (Wang and Manning, 2012)	66.36%	0.6046
MNB (Tf-Idf) (Wang and Manning, 2012)	63.53%	0.4783
Lexicon Lookup (Sharma et al., 2015)	51.15%	0.252
Char-LSTM (Joshi et al., 2016)	59.8%	0.511
Subword-LSTM (Joshi et al., 2016)	69.7%	0.658
FastText (Joulin et al., 2017)	46.39%	0.505
SACMT (Choudhary et al., 2018)	77.3%	0.759
CMSA (Proposed)	**83.54%**	**0.827**

Table 1: Comparing against previous approaches

3.4 CMSA

The entire model is shown in Figure 3. Sub-word representations are fed into both the specific and the collective encoder. The outputs of the encoders are concatenated with each other, and further with the result of the Feature Network. Subsequently, these are passed through multiple fully connected layers to make the prediction. This architecture allows us to capture sentiment on the morphemic, syntactic and semantic levels simultaneously and learn which parts of a sentence provide the most value to its sentiment.

This system enables us to combine the best of neural networks involving attention mechanisms with surface and semantic features that are traditionally used for sentiment analysis.

4 Experiments And Results

4.1 Dataset

We use the dataset released by (Joshi et al., 2016). It is made up of 3879 code-mixed English-Hindi sentences which were collected from public Facebook pages popular in India.

4.2 Baselines

We compare our approach with the following:

- **SVM** (Pang and Lee, 2008): Uses SVMs with ngrams as features.
- **NBSVM** (Wang and Manning, 2012): Uses Naive Bayes and SVM with ngrams as features.
- **MNB** (Wang and Manning, 2012): Uses Multinomial Naive Bayes with various features.
- **Lexicon Lookup** (Sharma et al., 2015): The code-mixed sentences are first transliterated from Roman to Devanagari. Thereafter, sentiment analysis is done using a lexicon based approach.
- **Char-LSTM and Sub-word-LSTM** (Joshi et al., 2016): Character and sub-word level

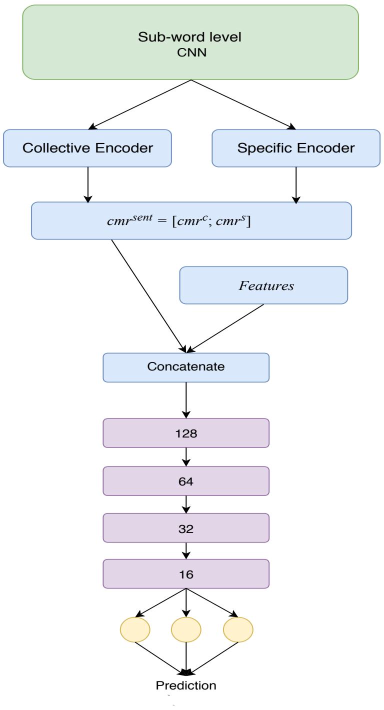


Figure 3: Complete Architecture of CMSA.

embeddings are passed to LSTM for classification.

- **FastText** (Joulin et al., 2017): Word Embeddings utilized for classification. It has the ability to handle OoV words as well.
- **SACMT** (Choudhary et al., 2018): Siamese networks are used to map code-mixed and standard sentences to a common sentiment space.

5 Results and Analysis

From Table 1, it is clear that that CMSA outperforms the state-of-the-art techniques by 6.24% in accuracy and 0.068 in F1-score. We observe that sentence vectors (FastText) alone are not suitable for this task. A possible reason for this is the presence of two different languages.

There is a significant difference in accuracy between Subword-LSTM and Char-LSTM, as seen in Table 1, which confirms our assumption about subword-level representations being better for this task as compared to character-level embeddings. Amongst the baselines, SACMT performed the best. However, mapping two sentences into a common space does not seem to be enough for sentiment classification. With a combination of different components, our proposed method is able to overcome the shortcomings of each of these baselines and achieve significant gains.

5.1 Effect of different Encoders

Method	Accuracy	F1-score
Specific Encoder	80.2%	0.801
Collective Encoder	77.3%	0.795
Specific + Collective (CMSA)	**83.54%**	**0.827**

Table 2: Different encoding mechanisms with Feature Network

We experiment with different encoders used in the dual encoder network. From Table 2, we can see that CMSA > Collective Encoder > Specific Encoder, for both accuracy and F1 score. A combination of the two encoders provides a better sentiment classification model which validates our initial hypothesis of using two separate encoders.

5.2 Effect of Different Model Components

System	Accuracy	F1-score
Dual Encoder	75.74%	0.705
Feature Network only	57.9%	0.381
CMSA	**83.54%**	**0.827**

Table 3: Effect of different model components

From Table 3, we see the performance trend as follows: CMSA > Dual Encoder > Feature Network. Although the feature network alone does not result in better overall performance, it is still comparable to Char-LSTM (Table 2). However, the combination of feature network with dual encoder helps in boosting the overall performance.

5.3 Effect of Training Examples

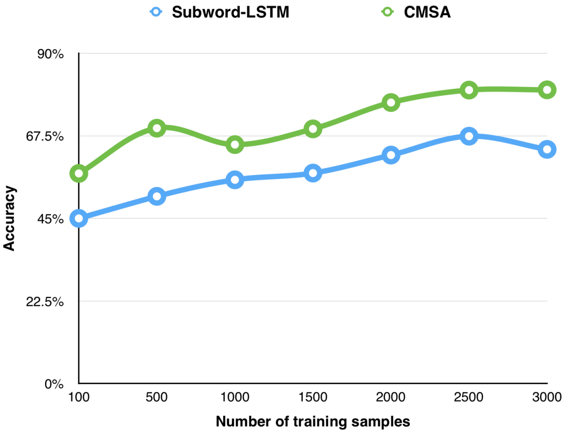


Figure 4: Performance on varying training data size

We experimented with varying numbers of training examples used for learning the model parameters. Different models were trained on 500, 1000, 1500 and 2000 examples. We observed that the performance of CMSA was greater than Subword-LSTM and SACMT at each point. This can be seen in Figure 4. One of the reasons for the same is the feature network in CMSA which helps in better performance even with lesser number of training examples.

6 Conclusion

We propose a hybrid approach that combines recurrent neural networks utilizing attention mechanisms, with surface features, yielding a unified representation that can be trained to classify sentiments. We conduct extensive experiments on a real world code-mixed social media dataset, and demonstrate that our system is able to achieve an accuracy of 83.54% and an F1-score of 0.827, outperforming state-of-the-art approaches for this task. In the future, we'd like to look at varying the attention mechanism in the model, and evaluating how it performs with a larger training set.

References

Oscar Araque, Ignacio Corcuera-Platas, J. Fernando Snchez-Rada, and Carlos A. Iglesias. 2017. Enhancing deep learning sentiment analysis with ensemble techniques in social applications. *Expert Systems with Applications*, 77:236 – 246.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Gokul Chittaranjan, Yogarshi Vyas, Kalika Bali, and Monojit Choudhury. 2014. Word-level language identification using crf: Code-switching shared task report of msr india system.

Nurendra Choudhary, Rajat Singh, Ishita Bindlish, and Manish Shrivastava. 2018. Sentiment analysis of code-mixed languages leveraging resource rich languages.

Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC06*, pages 417–422.

Maria Giatsoglou, Manolis G. Vozalis, Konstantinos Diamantaras, Athena Vakali, George Sarigiannidis, and Konstantinos Ch. Chatzisavvas. 2017. Sentiment analysis leveraging emotions and word embeddings. *Expert Systems with Applications*, 69:214 – 224.

Aditya Joshi, Ameya Prabhu, Manish Shrivastava, and Vasudeva Varma. 2016. Towards sub-word level compositions for sentiment analysis of hindi-english code mixed text. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING)*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431.

Saif M. Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. *CoRR*, abs/1308.6242.

Carol Myers-Scotton. 1993. Common and uncommon ground: Social and structural factors in codeswitching. *Language in society*, 22(4):475–503.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135.

A. Pravalika, V. Oza, N. P. Meghana, and S. S. Kamath. 2017. Domain-specific sentiment analysis approaches for code-mixed social network data. In *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–6.

A. Sharma, S. Gupta, R. Motlani, P. Bansal, M. Srivastava, R. Mamidi, and D. M. Sharma. 2016. Shallow Parsing Pipeline for Hindi-English Code-Mixed Social Media Text.

Shashank Sharma, PYKL Srinivas, and Rakesh Chandra Balabantaray. 2015. Text normalization of code mix and sentiment analysis. *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1468–1473.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Yogarshi Vyas, Spandana Gella, Jatin Sharma, Kalika Bali, and Monojit Choudhury. 2014. Pos tagging of english-hindi code-mixed social media content. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Sida Wang and Christopher D. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. In *HLT-NAACL*.