Knowledge Extraction from Code Mixed Text using Subword Patterns

Sumanth Manduru, Rambabu Yadav, Raviteja Gowtham, Viswanath P, Amitava Das

Indian Institute of Information Technology, Sri City, AP, India, 517588
{sumanth.m15, rambabu.g15, raviteja.g15, viswanath.p, amitava.das}@iiits.in

***Abstract*— Usage of code mixing languages on social media platforms is being increased. Because of which several applications have been emerged. One among them is Knowledge Extraction(KE). There are many synonyms for KE as Sentimental Analysis, Opinion Mining etc. Presently, current users are fond of native language which makes them to use it in many online platforms like Twitter, Facebook etc as a tool for the purpose of communication. In a Multilingual society like India, people use code mixing for grasping the content faster. We present Telugu-English dataset of size 5K Tweets for performing the sentimental analysis of a code mixed text.**

In this paper, We implement two stacked Long Short Term Memory model which learns the subword level patterns by performing series of 1D convolutions in predicting the sentiment of a code mixed text[Te-En]. By doing this, the model gains the ability to interpret the sentiment quality of the morphemes present in the sentence. We also believe that the capacity of judging the content present in code mixed text have been increased by these subword patterns. Thereby, we achieved about 6-9% higher accuracy than the state-of-art models.

I. INTRODUCTION

Social media evolution brought many challenges along with new opportunities in accessing the information and also for language technologies to tackle the facing challenges. Monolingual content of English has been reducing in blogs, sites and social media platforms like Facebook, Twitter, WhatsApp and Informal chats. It is inferred by twitter that only 50% of the data is in English. It implies the usage of code-mixing in the present era. In the process of conveying the emotion, people use their native language along with English words in between for easy understanding. In exploring this situation, researchers have laid the foundation for code-mixing and code-borrowing. Code-Mixing is also called as Code-Switching. Code-Mixing can be highly observed in Indian Sub Continent, in which 22 official languages are being spoken, written in 13 different scripts, with over 720 dialects. Out of them, Telugu is the third most widely spoken language in India. Hindi and Bengali tops the list with first and second respectively. Telugu is mostly being spoken in Andhra Pradesh and Telagana.

We considered Telugu and English as our preferred languages. In this paper, First we discuss about the preparation of Te-En code-mixed dataset. Second we focus on the significance of sub-word level features. Finally we compare the experimentally evaluated result with state-of-the-art models. We illustrate the essence of subword level patterns compared with other representations like word level patterns, character level patterns. Introduction about code-mixing takes the first section in the paper. In section 2, we describe about our data collection and challenges faced to prepare such dataset. Coming to section 3, Illustration of our two stacked LSTM Model. Experimentation and Observations are placed in the section 4.

II. DATASET DESCRIPTION

A. Data Collection

We mentioned already Telugu-English are the preferred code-mixed languages. We manually prepared a dictionary consisting of nearly 8500 telugu spoken words. Using Twitter API, we collected code mixed tweets which contain these telugu spoken words. Finally, we collected about 7000 code-mixed tweets.

B. Data Cleaning

Data cleaning is an integral part so as to remove noise from the considered code mixed tweet. We focused on removing ReTweet(RT), Hashtag, URLs, @Das12(UserId), Emojis like :), ;), :(etc.

We validate each token such a way that it should start with alphabets rather than with special characters. For example

- $!!!Hurrah$ – Hurrah
- $\&bamboo$ – bamboo

We take care of multiple spaces. Converting more than 2 letter repetitions to 2 letter in a word is also a part of cleaning.

- Fooooood – Food
- Happppppy – Happy

Although the last point is applicable to the most of the cases, there are some cases where these conditions fail such as

- Goooooood – Good, What if user mean God ?

We considered the last statement so that we can bring all such different numerous repetitions to one platform.

C. Code Mixed Index

All retrieved tweets does not come under code mixed tweets because there may be cases like monolingual tweets, mixing is not done at proper ratio. We calculated Code Mixed Index(CMI) for each of these tweets and considered the tweets whose code mixing is above the threshold value of 0.33. To calculate CMI, we need to know the language at word level. We manually labelled each word present in each tweet. Word Level classes are Telugu(Te), English(En),

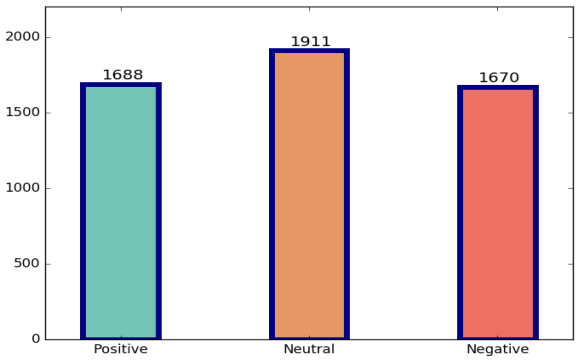

Fig. 1. Composition of Te-En Code mixed dataset

Ambiguous(Amb) and Universal(U). Names comes under Ambiguous where as Numbers, Special Characters comes under Universal Class. Due to the complexity in finding the Te-En tweets, we have kept the low threshold value.

$$CMI = \begin{cases} 100 * \left[1 - \frac{max(w_i)}{n-u}\right], n > u \end{cases} \quad (1)$$

where $max(w_i)$ implies the maximum tokens belong to a particular language, n refers to total no of tokens and u refers to languages independent classes such as Ambiguous, Universal in our case. For Monolingual Tweets, CMI = 0

D. Illustration of CMI

Below are some of the instances from our dataset. We show the code mixed tweets labelled at word level in order to calculate the CMI for the following three cases such as CMI = 0, CMI $<$ 0.33 and CMI $\geq$ 0.33.

- "Babai(Te) Swaroop(Amb) ,(U) Ee(Te) prapancham(Te) chala(Te) chinnadi(Te)" - CMI = 0 (Monolingual Tweet)
- "asalu(Te) evaru(Te) confirm(En) chesaru(Te) ?(U)" - CMI = 0.25 $<$ 0.33
- "Hyderabad(Amb) lo(Te) Iroju(Te) manaki(Te) Pakistan(Amb) vallaki(Te) One(En) Day(En) Cricket(Amb) Match(En) anta(Te)" - CMI = 0.37 $>$ 0.33

Using the formula mentioned in the equation 1, the above CMI's are calculated.
After performing CMI on collected tweets, we are able to prepare a code mixed dataset of size 5269. Labelling each tweet into any of the three classes like Positive, Neutral and Negative.

Our dataset is the first dataset, whose size is more than 5K, published in the domain of Telugu-English preferred code mixing languages. It has the tag of code-mixed and sentiment as well. Composition of dataset is shown in figure 1.

III. LEARNING PATTERNS

In this section, we explain the method of predicting the sentiment of a code mixed text. Subword level features have high impact on predicting the sentiment of a code mixed text

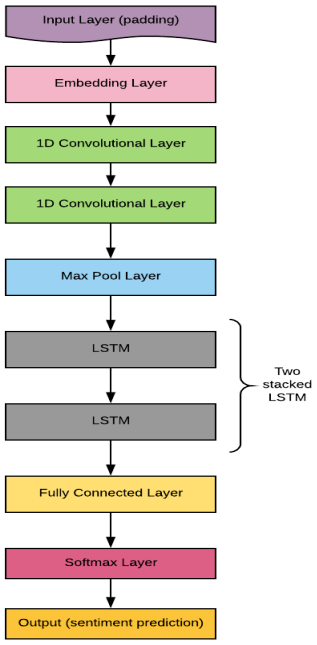

Fig. 2. Illustration of Model

rather than word level and character level. We try to explain the reason for choosing the subword level patterns over word and character level.

- Word Level Patterns : In word level patterns, there are assumptions like
  - Finite vocabulary. Though it performs well in the case of English but in code mixed, it is not valid. Finite vocabulary does not applicable in real scenarios as new words are being generated in telugu as well as english. It is tough to maintain finite size dictionary. Even if maintaining finite dictionary is possible, sparsity of word may cause disturbance in learning the word level patterns.
- Character Level Patterns : In character level patterns, there is no assumptions of finite vocabulary, which makes freedom to generate new words. But, there is no guarantee that words having sense are being generated all the time.

We hypothesize that subword level patterns can account for the production of new lexical structures and inturn, they can help in predicting the sentiment better than character level patterns.

A. Subword Level Patterns

Subword level representations are generated while performing the operation of convolution on given input. These patterns are helpful in increasing the performance of the model.

1) METHODOLOGY: Performing the 1-D convolution operation(s) on given input can generate subword level patterns. We perform convolution 1-D on the given text for two times, one after the other. The reason is we found

that performing convolution on already convolved input brings out the better performance. There by we notice that there are some features which are generated during second time convolution, help in increasing the performance of the prediction model.

Let D represents the code mixed[Te-En] dataset comprises of many code mixed sentences. Let I refers to the given code mixed input sentence consists of n characters $[c_1, c_2, ..., c_n]$ and I ϵ D. We apply padding to the given sentence so as to bring out all input sentences to same level. Let p be the padding size of the input sentence. Therefore the size of input sentence is p. Hence the input matrix can be represented as A ϵ R^{mxp} where m implies the length of character embedding vector for the character in the given input sequence $[c_1, c_2, ..., c_p]$ of length p.

We apply 1-D convolution on input matrix A with a filter G ϵ R^{mxz} of size z. Later we add bias to the resultant and pass it through non-linear layer to generate a subword level feature. Repeat the same procedure on already convolved input for one more time to generate much stronger feature map of subword level. Let f be the feature map obtained and it is represented as f ϵ R^{p-2z+2}. We apply max pool layer of size z for the generated feature patterns and thus obtain the maximal values.

B. Model Description

The Model comprises of two stacked LSTM layers followed by fully connected layer. After the max pool layer, the obtained feature maps are passed into the two stacked lstm layers. LSTM are known for remembering the useful information for long time. It is free from vanishing gradients. LSTM consists of three gates namely Forget, Update and Output gates. It has a memory cell state. Lets describe the functionality of each gate in detail. we feed feature map as input to the cell state at time $t-1$

- Forget Gate : It calculates the amount of information from time $t-1$ need to be stored in the cell state at time t. The equation as follows :

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{2}$$

  W_f, b_f represents to weight and bias assigned to forget gate respectively, h_{t-1} is the activation output at time $t-1$ and x_t is the input feature map at time t. Output of 1 represents completely keep the information. 0 implies get rid of the information present in cell state at time t.
- Update Gate : This gate helps in what new information we need to store in the cell state at time t.

$$i_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \tag{3}$$

$$C`_t = \tanh W_c.[h_{t-1}, x_t] + b_c \tag{4}$$

  i_t implies input gate, $C`_t$ implies candidate value for the cell state. W_i, b_i represents to weight and bias assigned to input gate respectively. Similarly W_c, b_c represents to weight and bias assigned to calculate candidate value for the cell state respectively.

$$C_t = f_t * C_{t-1} + i_t * C`_t \tag{5}$$

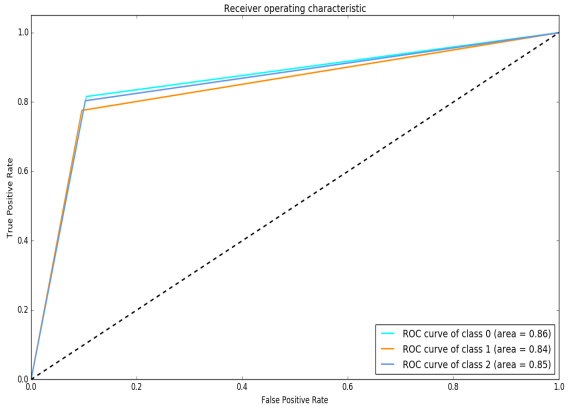

Fig. 3. ROC Curve

  C_t is updated cell state at time t. C_{t-1} represents the cell state information at time $t-1$
- Output Gate : Finally, we need to decide what were going to output. This output will be based on our cell state.

$$O_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \tag{6}$$

  W_o, b_o represents to weight and bias assigned to output gate respectively

$$h_t = O_t * \tanh C_t \tag{7}$$

  h_t indicates the activation output at time t which is passed into next LSTM cell at time $t+1$.

Overview of our model is shown in the above figure . The Functions and Metric we have used goes as follows : The input layer is a row vector of size 200, including padding. Each input sentence is passed through tokenization and also lowering each character, before letting the input through input layer for padding. The Embedding layer we have used is of 128 dimensions. LSTM cells are of dimensions 128 having 0.2 as dropout metric. Activation function that are mentioned in convolution layers is ReLu. Loss Metric Function is Categorical Cross Entropy and where as optimizer is Adamax. We have implemented this model using Keras, Deep Learning Library.

IV. EXPERIMENTAL RESULTS

On performing series of two 1-D convolution operations on given input, our model achieved an accuracy of 79.4%. Further, Joshi proposed the subword LSTM with one convolution layer, has achieved an accuracy of 69.7%. It shows that our model has shown good performance by a large margin of 10%. We also found that there is no extension of adding further more convolution layers because the model goes into overfitting state upon adding further convolution layers.

In the figure 3 shown above, it represents the Receiver Operating Characteristic curve, in short as ROC curve. The area covered by Negative class is 0.86. Neutral Class covers an area of 0.84 and final class, Positive holds an area of 0.85.Training and Testing accuracies of the model are shown

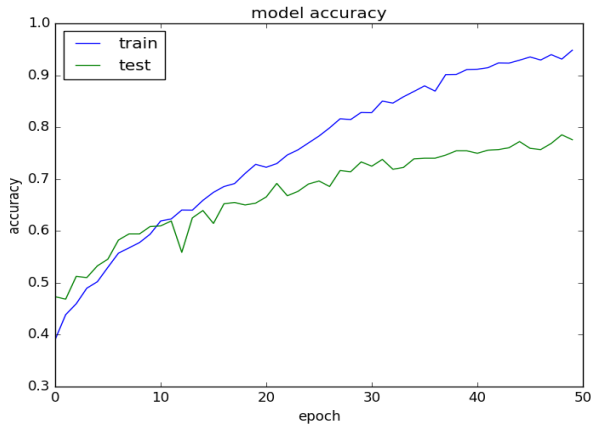

Fig. 4. Training and Testing Accuracies

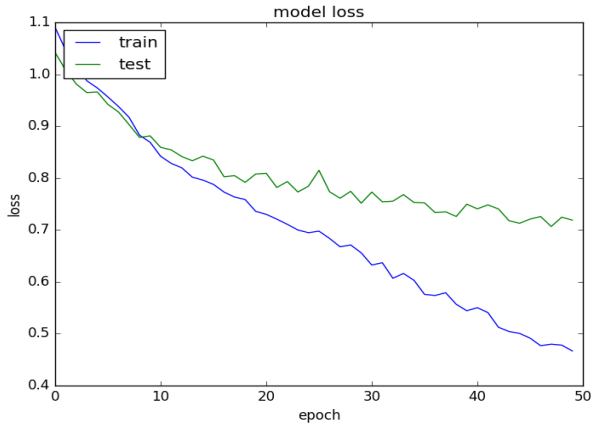

Fig. 5. Loss variations

in the figure 4. Loss variations of the model is also shown in the figure 5.

As mentioned earlier, our dataset size is of 5269. We split it into three sets namely train data, cross-validation data and finally test data. Train data is of size 3372 where as cross-validation holds 843 samples. Test data contains 1054 code mixed sentences. In the figure 6, We have shown confusion matrix with normalization. Out of 1054 Test data, 326 belongs to Negative, 366 goes into Neutral class and remaining 362 comes under Positive class.
Out of 326 Negative statements, 266 were correctly predicted by the model as Negative, 28 as Neutral and 32 as Positive. Similarly out of 366 Neutral statements, 284 classified correctly and 43 as Negative and 39 as Positive. 291 samples are correctly predicted as positive by the model out of 362 positive samples and 33 as Negative and 38 as Neutral. F1-score of the model is noted as 0.797 where as previous state-of-art model has an F1-score of 0.658

In the above Table1, we have shown the output dimension of the input matrix at each stage of the training process. The stride for both convolution layers is 1 where as for max-pooling layer, the stride is 3.

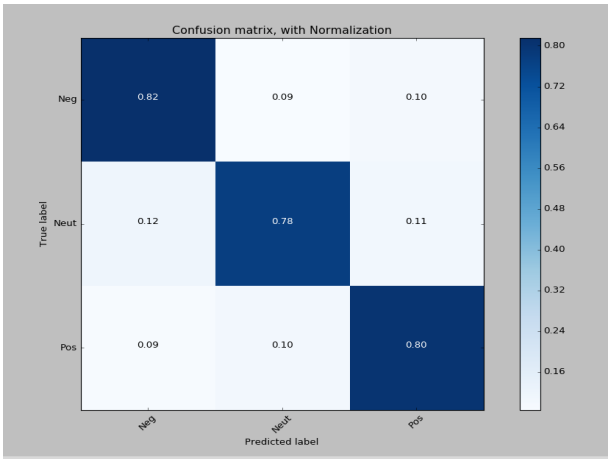

Fig. 6. Confusion Matrix with Normalization

TABLE I
NETWORK ARCHITECTURE OF THE PROPOSED SUBWORD LSTM MODEL

Layer Type	Filter Size	Output Dimension
Input	-	1 x 200
Embedding Layer	-	1 x 200 x 128
Convolution 1D -1	3	1 x 198 x 128
Convolution 1D -2	3	1 x 196 x 128
Maxpooling 1D	3	1 x 65 x 128
LSTM - 1	-	1 x 65 x 128
Dropout - 1	-	1 x 65 x 128
LSTM - 2	-	1 x 128
Dropout - 2	-	1 x 128
Dense	-	3

TABLE II
CLASSIFICATION REPORT

Class	Precision	Recall	F1-Score
Negative	0.78	0.81	0.79
Neutral	0.79	0.78	0.78
Positive	0.81	0.79	0.80
Avg/Total	0.79	0.79	0.79

TABLE III
PERFORMANCE METRICS-I

Class	Sensitivity(TPR)	Specificity(TNR)
Negative	0.81	0.89
Neutral	0.78	0.89
Positive	0.79	0.90

TABLE IV
PERFORMANCE METRICS-II

Class	False Positive Rate	False Negative Rate
Negative	0.10	0.19
Neutral	0.11	0.22
Positive	0.10	0.21

In the Table II, we have shown the classification report of the proposed model. Classification Report basically covers the common evaluation metrics such as precision, recall and F1-Score of each class.

In Table III and Table IV, we have mentioned True Positive

Rate(TPR), True Negative Rate(TNR) and False Positive Rate, False Negative Rate respectively for each class.

V. CONCLUSIONS

Even though, Telugu is holding the third place in the section of widely spoken languages in India. It is quite tough to collect real and good in content code mixed data on online platforms like Twitter, Facebook and many more. Before making out subword patterns by the help of convolutional layers, we would like to present the Telugu-English code mixed data. Due to unavailability of any other Te-En dataset, we did not perform comparison. In this paper, we have presented the modified version in increasing the performance of a model in predicting the sentiment of a code mixed text. The model is enough trained to predict the sentiment in a noisy-dataset of code mixed languages, Telugu-English.

REFERENCES

[1] Joshi, A.; Prabhu, A.; Shrivastava, M.; and Varma, V. 2016. Towards sub-word level compositions for sentiment analysis of hindi-english code mixed text. In COLING, 24822491.

[2] U. Kumar, V.S.Rana, C.Andrew, S.Gongidi, and A. Das. Consonant-Vowel Sequences as Subword Units for Code-Mixed Languages. In the student session of the AAAI 2018, Louisiana, USA. Intelligence 2018

[3] Sharma, S.; Srinivas, P.; and Balabantaray, R. C. 2015. Text normalization of code mix and sentiment analysis. In Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on, 14681473. IEEE.

[4] U. Barman, A. Das, J. Wagner, and J. Foster. Code-Mixing: A Challenge for Language Identification in the Language of Social Media. The 1st Workshop on Computational Approaches to Code Switching, EMNLP 2014 , pages 13-23, October, 2014, Doha, Qatar.

[5] A. Das and B. Gambck. Identifying Languages at the Word Level in Code-Mixed Indian Social Media Text. The 11th International Conference on Natural Language Processing (ICON-2014) , December, 2014, Goa, India.

[6] Nguyen, D., Dogruoz, A.S.: Word level language identification in online multilingual communication. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. pp. 857862. ACL, Seattle, Washington (Oct 2013).

[7] Sharma, S.; Srinivas, P.; and Balabantaray, R. C. 2015. Text normalization of code mix and sentiment analysis. In Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on, 14681473. IEEE

[8] Shruti Rijhwani, Royal Sequiera, Monojit Choudhury, Kalika Bali, Chandra Shekhar Maddila: Estimating Code-Switching on Twitter with a Novel Generalized Word-Level Language Detection Technique. ACL (1) 2017: 1971-1982

[9] Gokul Chittaranjan, Yogarshi Vyas, Kalika Bali, Monojit Choudhury: Word-level Language Identification using CRF: Code-switching Shared Task Report of MSR India System. CodeSwitch@EMNLP 2014: 73-79

[10] A. Jamatia, B. Gambck, and A. Das. Collecting and Annotating Indian Social Media Code-Mixed Corpora. In the proceeding of the 17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLING), April 3-9, 2016, Konya, Turkey.