Hate Speech and Offensive Content Identification: LSTM Based Deep Learning Approach @ HASOC 2020

Baidya Nath Saha[a], Apurbalal Senapati[b]

[a]*Concordia University of Edmonton, 7128 Ada Blvd NW, Edmonton, Alberta, Canada, T5B 4E4*
[b]*Central Institute of Technology, Kokrajhar, BTAD, Assam, India, 783370*

Abstract
The use of hate speech and offensive words is growing around the world. It includes the way of expression in vocal or written form that attacks an individual or a community based on their caste, religion, gender, ethnic groups, physical appearance, etc. The popular social media like Twitter, Facebook, WhatsApp. Print media and visual media are being exploited as a platform for hate speech and offensive and increasingly found in the web. It is a serious matter for a healthy democracy, social stability, and peace. As a consequence, the social media platforms are trying to identify such content in the post for their preventing measure. FIRE 2020 organizes a track aiming to develop a system that will identify hate speech and offensive content in the document. In our system we (CONCORDIA_CIT_TEAM) have used the Long Short Term Memory (LSTM) for automatic hate speech and offensive content identification. Experimental results demonstrate that LSTM can successfully identify hate speech and offensive content in the documents of various languages successfully.

Keywords
Hate Speech, Offensive Content, Deep Learning, LSTM

1. Introduction

Since the last decade, social media has been attracting to the masses and growing exponentially. It has been becoming a great platform to communicate with people irrespective of their social status towards their democratic process [1]. With the significant rise of cheap and user-friendly handy devices such as smartphone, and tablets across the world, people are spending a significant amount of time on various social media like Twitter, Facebook, Instagram, and so on. As a public communication, there are various emotions reflected in the critical discourse [2]. As a result, web content is also growing rapidly, and some content include abusive languages in different forms [3].

2. Task Description

The FIRE 2020 track [4] focuses on the identification of hate speech and offensive content in Indo-European Languages. Particularly it aims to develop the system for English, German, and

FIRE '20, Forum for Information Retrieval Evaluation, December 16–20, 2020, Hyderabad, India.
baidya.saha@concordia.ab.ca (B. N. Saha); a.senapati@cit.ac.in (A. Senapati)

Hindi languages. The track consists of two sub-tasks that are mentioned below.

2.1. Sub-task A

This task is to identify the Hate speech and Offensive language for the English, German, and Hindi languages. It can be mapped to a two-class classification problem where system will be developed to classify texts into two classes, namely: Hate and Offensive (HOF) and Non- Hate and offensive (NOT).

- **(NOT) Non Hate-Offensive:** This post does not contain any Hate speech, profane, offensive content.

- **(HOF) Hate and Offensive:** This post contains Hate, offensive, and profane content.

2.2. Sub-task B

The sub-task B is a fine-grained classification for English, German, and Hindi texts. Hate-speech and offensive posts from the sub-task A are further classified into three categories:

- **(HATE) Hate speech:** Posts under this class contain Hate speech content.

- **(OFFN) Offenive:** Posts under this class contain offensive content.

- **(PRFN) Profane:** These posts contain profane words.

3. Related work

A number of literature [5] can be found on the topic related to hate speech and offensive content identification. It describes the definition of hate speech in a different aspect and the solution strategies. Some of them used the techniques like dictionaries, N-gram, bag of words along with the syntactic information. On the other hand, others used the machine learning, deep learning-based classifications. Several shared task is organized related to this issue providing the data, results in the public domain. FIRE 2019 organized a track [6] on "Hate Speech and Offensive Content Identification in Indo-European Languages" in three languages English, German, and Hindi. There were 321 experiments were submitted with different approaches. SemEval-2019 [7] organized the shared task on identifying and categorizing offensive language in various social media in which nearly 800 teams registered to participate in the task but finally, 115 of them submitted results.

The GermEval2018 [8] Shared Task was intended for the identification of offensive language. The task deals with the classification of German tweets from the twitter data. It comprises two tasks, a coarse-grained binary classification and the other one is a fine-grained multi-class classification task. There were 20 participants and they submitted 51 runs for the first category task and 25 runs for the second category task.

The TRAC-1 (First Workshop on Trolling, Aggression and Cyberbullying) @ COLING 2018 [9] focussed on the aggression and related activities like trolling, flaming, cyberbullying, and hate speech in both text (especially in the social media text) and speech.

4. Methodology

In the system implementation, we have used a special type of Recurrent Neural Network (RNN) based deep learning approach known as Long Short Term Memory (LSTM) [10, 11] to detect hate speech and identify offensive content in the text. RNN is capable to connect the past information in their learning process but there is a drawback that it could not process the previous relevant information and long dependencies. This was overcome by a special type of RNN called LSTM which ie being widely used nowadays.

4.1. LSTM networks

The standard RNN can be considered as a single layer chain of repeating modules of a neural network. The repeating module will have a simple structure, such as a single ***tanh*** function. The LSTMs architecture is also similar types of a chain-like structure, but the repeating module has a relatively complex structure. The key feature of an LSTMs is the cell state, the horizontal line running through the top of the cell diagram. The information passes along the cell with a minor change. The LSTM can remove or add information to the cell state which is regulated by a special structure called gate.

4.2. LSTM walk-through

The first step of the LSTM [12] is to decide what information is going to throw away from the cell state. It is made by a sigmoid layer called the forget gate. It takes h_{t-1} and x_t as input, and produce outputs a number between 0 and 1 for each number in the cell state C_{t-1}. The output 1 represents “completely keep this" while a output 0 represents “completely get rid of this." The output of the forget get is defined by f_t, the equation of which is illustrated Fig. 1.

The notations used to describe the LSTM is given below:

- $x_t \in \mathbb{R}^d$ is the input vector of the LSTM unit
- $f_t \in \mathbb{R}^h$ is activation vector (forget gate)
- $i_t \in \mathbb{R}^h$ are input and update of the gate vector
- $o_t \in \mathbb{R}^h$ is the exit of the activation
- $h_t \in \mathbb{R}^h$ are hidden state vectors (exit vector of the LSTM unit)
- $\widehat{C_t} \in \mathbb{R}^h$ are new candidate vectors for cell status
- W and b are the weight matrices and bias vectors that need to be learned during network training

The next step is to decide about the new information we're going to store in the cell state. It is incorporated by the sigmoid function and ***tanh*** function. First, a sigmoid layer called the "input gate layer" decides which values will be updated. Next, a tanh layer generates a vector of

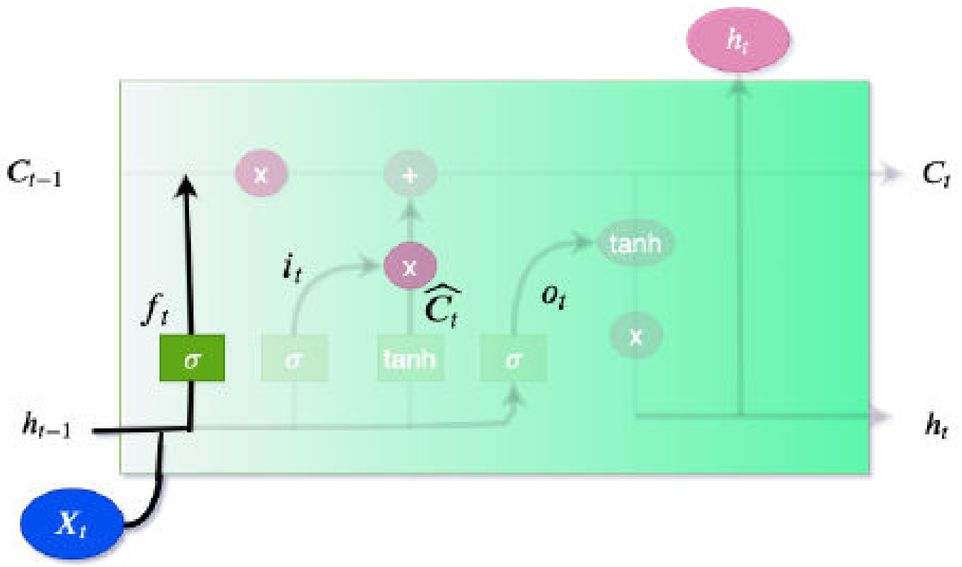

Figure 1: Forget gate layer. Inspired from [13].

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

new candidate values, $\widehat{C_t}$(Fig. 2), that could be added to the state. Finally, combine these two (Fig. 3) to create an update to the state.

Finally, it produces the output (Fig. 4) in the form of -1 or 1. This output will be based on the cell state but will be a filtered version. First, it runs a sigmoid layer which decides what parts of the cell state are going to output. Next, it put the cell state through the function ***tanh*** (values between 1 and 1) and multiply it by the output of the sigmoid gate, so that we only output the parts that we decide.

5. Data description

Dataset for the shared task is provided by FIRE 2020 organizers (HASOC 2020) [4]. They create a data set following the labeling scheme of OffensEval [7] and GermEval [8]. They provide the training and test data in English, German, and Hindi data. The data is described in the column format of five fields and the detail description is given below:

- **tweet_id** - unique ID for each piece of text
- **text** - the text of the tweet
- **task1** - the general label of the tweet (sub-task A)
- **task2** - the general label of the tweet (sub-task B)
- **ID** - unique ID generated by the system

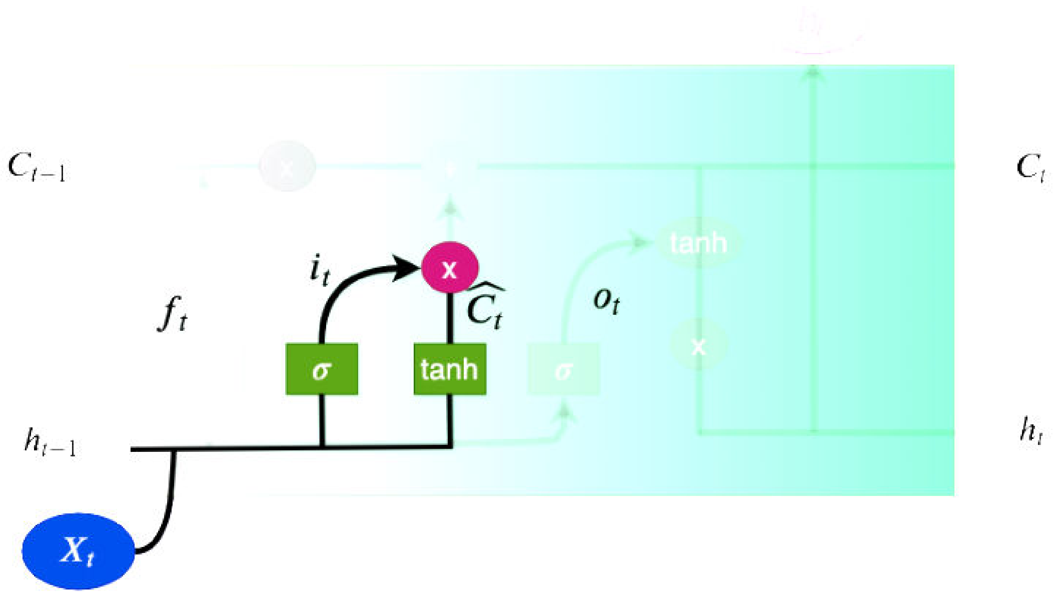

Figure 2: Update the layer. Inspired from [13].

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\widehat{C_t} = tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

6. Experimental results and discussions

As we explained about our implemented LSTM system, that exploited the following architecture:

word embedding $\rightarrow$ **LSTM with hidden layer** $\rightarrow$ **Dense layer with a neuron** $\rightarrow$ **sigmoid activation function**.

Vectors length 128 for embeddings layer, 128 neurons in each hidden layer, batch size 60, 10 number of epochs and a dropout of 20% were chosen for this experiment. In order to establish the convergence of the network, binary and categorical cross entropy type error function were used for two- and multi-class classification respectively. ADAM optimizer was used for all the tasks classification. We used the default parameters of Keras for ADAM optimizer.

The results of LSTM based deep RNN architecture for Sub-task A and Sub-task B of HASOC identification are shown in Table 1 and Table 2 respectively. The binary cross-entropy loss function is used for the binary sentence classification: hate and offensive (HOF) and non-hate-offensive classification (NOT) associated with Sub-task A. Categorical cross-entropy loss function is used for multiclass sentence classification associated with Sub-task B: hate speech (HATE), offensive (OFFN), profane (PRFN), and NOT. Table 1 and Table 2 demonstrate the

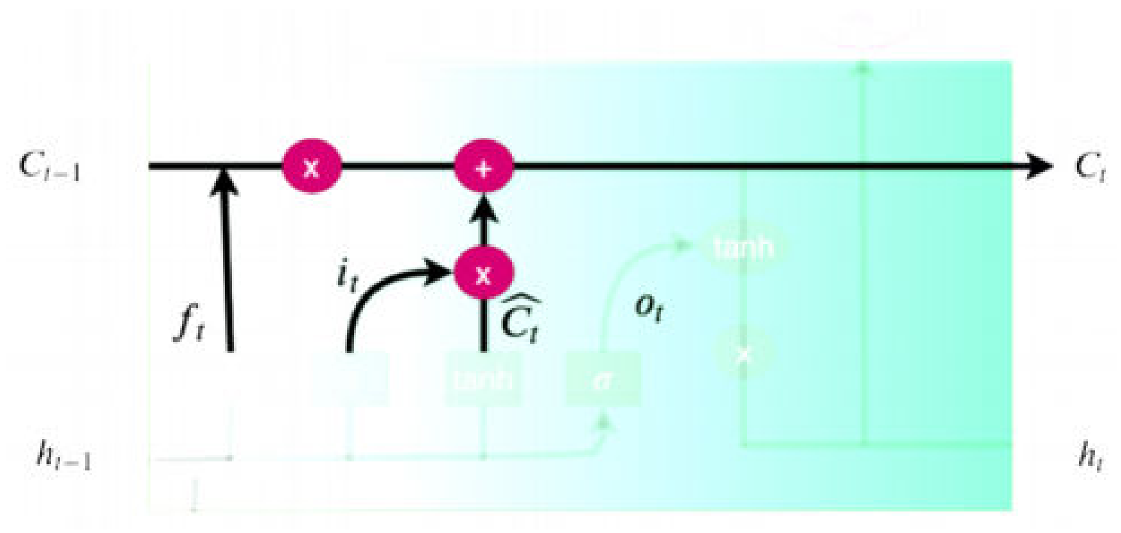

Figure 3: Updated layer. Inspired from [13].

$$C_t = f_t * C_{t-1} + i_t * \widehat{C_t}$$

Table 1
Classification results for Sub-task A.

Language	F1 Macro average	Highest score
Hindi	0.5027	0.5337
English	0.5078	0.5152
German	0.5200	0.5235

Table 2
Classification results for Sub-task B.

Language	F1 Macro average	Highest score
Hindi	0.2323	0.3345
English	0.2115	0.2652
German	0.2727	0.2943

performance of the LSTM based RNN classifier in terms of the weighted average of accuracy F1 Macro average. Results demonstrate that LSTM based RNN classifier performs better in English than the Hindi dataset for Sub-task A but opposite in Sub-task B.

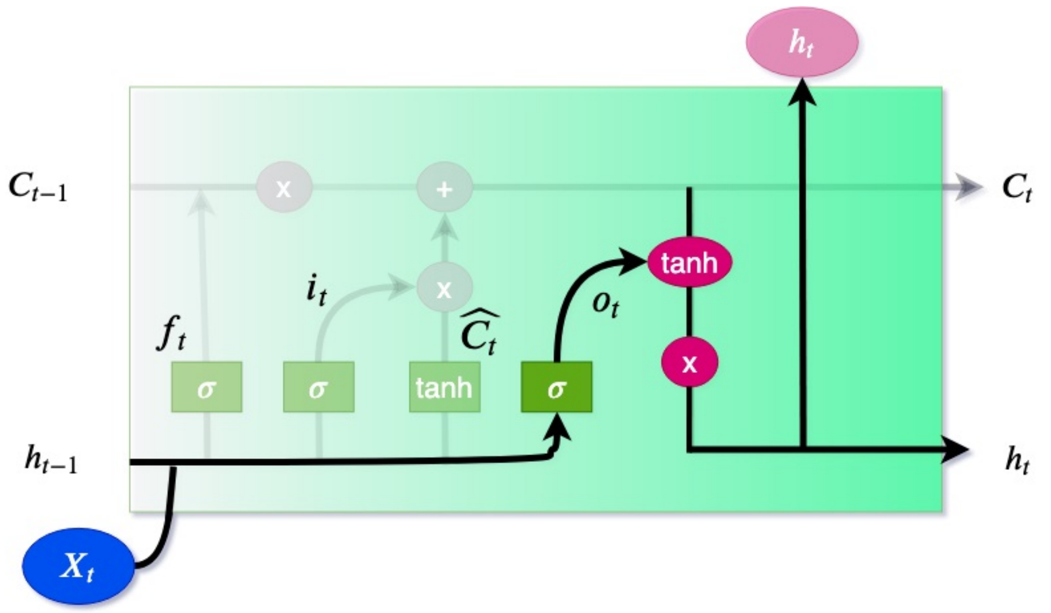

Figure 4: Output layer. Inspired from [13].

$$o_t = \sigma(W_o[h_{t-1}, h_t] + b_o)$$
$$h_t = o_t * tanh(C_t)$$

7. Conclusion

We proposed a Long Short Term Memory (LSTM) based Recurrent Neural Network (RNN) for automatic HASOC identification in three Indo-European languages: English, German, and Hindi. The advantages of the proposed methodology include: (a) it does not use any pre-trained model which offers a languag-agnostic solution; and (b)there is no feature engineering required for the proposed model. However, the bottlenecks of HASOC detection are that it is very subjective and context-dependent in nature. In future, we would like to implement other rich deep learning architectures for HASOC identification.

References

[1] G. Pitsilis, H. Ramampiaro, H. Langseth, Effective hate-speech detection in twitter data using recurrent neural networks., Applied Intelligence 48 (2018) 4730–4742. doi:10.1007/s10489-018-1242-y.
[2] J. Habermas, The Theory of Communicative Action, Volume 1: Reason and the Rationalization of Society, Boston: Beacon Press, 1984.
[3] Y. Chen, Y. Zhou, S. Zhu, H. Xu, Detecting offensive language in social media to protect adolescent online safety, in: 2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing, 2012, pp. 71–80.

[4] T. Mandl, S. Modha, G. K. Shahi, A. K. Jaiswal, D. Nandini, D. Patel, P. Majumder, J. Schäfer, Overview of the HASOC track at FIRE 2020: Hate Speech and Offensive Content Identification in Indo-European Languages), in: Working Notes of FIRE 2020 - Forum for Information Retrieval Evaluation, CEUR, 2020.

[5] P. Fortuna, S. Nunes, A survey on automatic detection of hate speech in text (2018). doi:10.1145/3232676.

[6] T. Mandl, S. Modha, P. Majumder, M. Dave, D. Patel, C. Mandlia, Jaiswal, A. Patel, Hasoc (2020) at fire 2019: Hate speech and offensive content identification in indo-european languages., in: Overview of the HASOC track at FIRE 2019: Hate Speech and Offensive Content Identification in Indo-European Languages, 2019.

[7] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, R. Kumar, Identifying and categorizing offensive language in social media (offenseval), in: In: Proceedings of the 13th International Workshop on Semantic Evaluation., 2019, pp. 75–86.

[8] M. Wiegand, M. Siegel, J. Ruppenhofer, Overview of the germeval 2018 shared task on the identfication of ffensive language (2018), in: Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018), 2019.

[9] R. Kumar, A. K. Ojha, S. Malmasi, M. Zampieri, Benchmarking aggression identification in social media, in: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018), 2018, pp. 1–11. URL: https://www.aclweb.org/anthology/W18-4401.

[10] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation, 9(8) 48 (1997) 1735–1780.

[11] A. S. Baidya Nath Saha, Cit kokrajhar team: Lstm based deep rnn architecture for hate speech and offensive content (hasoc) identification in indo-european languages, in: Proceedings in Forum for Information Retrieval Evaluation (FIRE). Forum for Information Retrieval Evaluation (FIRE), volume 2517, 2019, pp. 1–7.

[12] C. Olah, Understanding lstm networks, accessed: October, 2020 (2015). URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[13] Understanding lstm networks, https://colah.github.io/posts/2015-08-Understanding-LSTMs/, ???? Accessed:December, 2020.