Social Network Analysis and Mining (2022) 12:100
https://doi.org/10.1007/s13278-022-00929-1

ORIGINAL ARTICLE

FA-Net: fused attention-based network for Hindi English code-mixed offensive text classification

Shikha Mundra[1,2] · Namita Mittal[1]

Received: 28 February 2022 / Revised: 27 June 2022 / Accepted: 12 July 2022 / Published online: 3 August 2022

Abstract
Widespread usage of social media platforms like Twitter, Facebook, and YouTube allows sharing of opinions and suggestions across countries. On the contrary, these platforms are often misused to disseminate hate speech and offensive content. Moreover, in a multilingual society such as India, many users resort to code-mixing while typing on social media. Thus, we have focused on Hindi English (Hi–En) Code-Mixed hate speech and offensive text classification. Recently, numerous approaches have emerged, and most of these approaches use CNN and LSTM in a stacked manner to extract local and sequential semantic features. However, these arrangements diminish the comprehensive effect of local and sequential features. In addition, deep framework suffers from issue of vanising gradient. Therefore, in our work, we have proposed, local and sequential knowledge aware Fused Attention-based Network (FA-Net), which introduces a fusion of attention mechanism of collective and mutual learning between local and sequential features. The proposed network (FA-Net) is lower in depth more in breadth in comparison to the existing architectures. It has three building blocks: Code Mixed Hybrid Embedding, Locally Driven Sequential Attention-2 (LDSA-2), Locally Driven Sequential Attention-3 (LDSA-3). CMHE is developed using customized Hi-En code mixed data, aiming the network to initialize with relevant weights. LDSA-2 and LDSA-3 equip the model to build a comprehensive representation having past, future, and local contextual knowledge w.r.t any location in the sentence. Extensive experimentation on two benchmark datasets shows that FA-Net has outperformed other state of the art.

Keywords Embedding · Fusion · Comprehensive representation · Hindi English code-mixed · Offensive

1 Introduction

With the phenomenal growth of the internet and social media platforms, incidents of abuse, hate speech, and offensive language has increased at a faster rate. Usage of body-shaming, sexually abusive, and hurtful language have become common these days. Patchin and Hinduja (2018) analysed that people who have been harassed or bullied online are nearly twice as likely to attempt suicide. Monto et al. (2018) stated that approximately 18% of youth report self-harming at least once due to online insulting and bullying comments. According to Bulao (2022), statistics reveal that every human creates 1.7 MB of data every second. Every day, 5 lakh new tweets were posted in 2020. This shows that users' engagement through social media is rising, which in turn increases the harassment incidents. Further, we have demonstrated survey reports which depict that cyberbullying events are exponentially rising in India.

As indicated by Shetty (2008), Microsoft conducts a survey on global youth online behaviour reveals that India is third-largest country suffering from offensive and cyberbullying incidents. A study by feminismindia Pasricha (2016) observed that nearly 50% of women in India's prime locations have been targeted for online abusive and body-shaming language. It is also reported that most of the users hurt themselves or suffered from depression problems. Along with this, multilingualism possesses another challenge in text understanding. Diversity in language across the globe results in a variety of linguistic patterns on social media. In such scenarios, identification of offensive incidents is not feasible with one global system due to variation in

✉ Shikha Mundra
a.shikha1990@gmail.com

Namita Mittal
nmittal.cse@mnit.ac.in

1 Malaviya National Institute of Technology (MNIT), Jaipur, India

2 Manipal University Jaipur, Jaipur, India

linguistic patterns. Hence, we need an automated system that can understand the language, its linguistic pattern, and its hate speech or offensive content features. Amongst multiple patterns, usage of code-mixed Hi-En is growing on social media, since Hindi is a widely spoken language in India Bali et al. (2015). Therefore, in our work, we have focused on Hi-En code-mixed language in the perspective of offensiveness detection. Further, we have explained about Hi-En code mixed language and its challenges in text understanding view.

1.1 Hi-En code-mixed language

According to Speaker (1995), code-mixing is a phenomenon where multilinguals borrow words from a second language while writing a comment in the first language. Nowadays, it has become usual to write in code-mixed manner during informal writing. On social media platforms, the Hi-En code-mixed language is growing exponentially, since Hindi is widely spoken by Indians and English is the most common language amongst all languages (Hinglish 2022). For instance,

Code-mixed. $Hypocrisy_{eng} ki_{hi} bhi_{hi} seema_{hi} tod_{hi} di_{hi} librandus_{slang} ne_{hi} or_{eng} feminist_{eng} ne_{hi}$

Translation. Feminist or liberals have crossed the limit of hypocrisy. (Only for reading purposes)

Here, *hi* represent romanized Hindi (Hinglish), and *eng* represent English and *slang* is English word in Hindi format with aggressive emotion. Hypocrisy and feminist are commonly used English words, hence easy to substitute while speaking or writing in Hindi. ki, bhi, seema are Hindi words written in Hinglish since the QUERTY keyboard is easy to use instead of Devanagari. Thus, the trend of using Hi-En code-mixed is rising on the internet Bali et al. (2015).

1.2 Challenges

Past work Ma et al. (2019) Dharma et al. (2022) Singh et al. (2022) depict that text representation using pretrained embedding has significantly improved the performance. Pretrained vector representations are lower in dimension and effectively capture contextual information. However, such pretrained embedding resources are limited to monolingual text Mikolov et al. (2013). Lack of availability of such resources for Hi-En code-mixed text is one of the key challenges since it is a mixture of Devanagari, English, and Hinglish words. Additionally, it suffers from the problem of spelling variation. Hence, it is challenging to map variously spelled words to original words. Another challenge is the limited contextual information in short-length sentences. Identifying offensive text in short-length code mixed sentences suffers from a lack of context. In such cases, local or global level features, independently, will not be effective. Thus, we need an automated system that can mutually understand code mixed offensive text at local and sequential level. Therefore, in this work, a new fused attention weight mechanism is proposed to develop comprehensive representation by extracting relevant sequential and local contextual features.

Our main contribution in this work is:

1. Code Mixed Hybrid Embedding(CMHE) is created at word and character n-gram level to initialize the network with relevant weights.
2. BiLSTM and convolutional structure based Fused Attention Network(FA-Net) is proposed in order to extract comprehensive information in sentences.
3. Empirical results on two benchmark datasets demonstrate that the proposed model has performed better against state-of-the-art approaches.

2 Related work

This section focuses on common feature extraction methods used in machine learning and deep learning for Hi-En code-mixed text classification. Analysis of related work is categorized in statistical and linguistic approaches, CNN-based approaches, RNN based approaches, and other combined approaches. Amongst these, statistical and linguistic features use machine learning classifier while CNN and RNN are concentrated on modelling local and sequential features, respectively. In order to get the combination of local and sequential knowledge, combined approaches have been studied. Along with this, transformer-based approaches have been analysed for comparative study.

2.1 Statistical and linguistic features

Considerable work has been done using statistical features like n-gram, bag of words, term frequency-inverse document frequency (tf-idf) for hate speech, and offensive language detection. Samghabadi et al. (2018) have experimented with char-n gram and word n-gram as features in order to overcome the issue of spell variation. These features were classified using logistic regression and multinomial naïve Bayes. Their work depicts that concatenation of char n-gram, word unigram feature performed best with logistic regression. In addition, they highlighted the need for word embedding for Hi-En code-mixed language. Sharma et al. (2015) have tried to normalize the Hi-En code mixed text to Devanagari text using the back transliteration approach. After this, they calculated the sentiment score from Hindi sentiwordnet lexicon as a feature. However, it is observed that back transliteration, being a rule-based method, unable to map variously spelled words to one Devanagari word. Thus, it results in an increase in out of vocabulary words. Si et al. (2019) created aggressive word lexicon and concatenated it with tf-idf, sentiment score, part of speech as linguistic features and found the best performance using XG Boost Ensemble learning method. Bohra et al. (2018) have used char n-gram, POS tag and other capital words as features and observed satisfactory performance using support vector machine with radial basis kernel. Overall, it has been observed that statistical feature based on character n-gram is mostly used in order to overcome spelling variation issues and linguistic features are used to enhance text understanding.

2.2 RNN based approaches

Well known embedding like glove, word2vec and random is investigated by Badjatiya et al. (2017) , they finetuned these pretrained embedding with LSTM and concluded that random embedding with LSTM performs best for English Hate Speech detection. Koufakou et al. (2020) augmented the dataset by merging various hate speech-related datasets exist in language Hindi and in language English. Further, the augmented dataset is initialized randomly and classified using the softmax classifier. However, the augmentation did not converge well due to the random allocation of weights. Santosh and Aravind (2019) has used phonic subword embedding and finetuned it with a hierarchical LSTM approach in order to obtain sequential feature. Majumder et al. (2022) uses word and character level embedding in order to understand Hi-En code mixed-text and used BiLSTM to extract forward and backward sequence dependency features for Hi-En code mixed named entity recognition.

2.3 CNN based approaches

Mathur et al. (2019) transliterated code-mixed Hi-En text to English text using rule based Hindi to English word mapping dictionary. In their work, the author didn't consider the word order, which in turn ignored the semantic meaning of the sentence. In addition to it, a manual profanity word list has been created to map abusive words to English words. Further, CNN and softmax are used to finetune and classify the features. Kapil and Ekbal (2020) has used multitask learning approach to classify hate speech, racism, sexism, offensive text in language English. Their model is based on learning and sharing weights from related task. They extracted and combined features from 5 related datasets at 2,3,4-gram level and finetuned them using concatenated CNN. Kim (2014) assumed a sentence has regional information instead of sequential knowledge. Hence, they proposed a randomly initialized CNN for sentence classification.

2.4 Combined approaches

Sasidhar et al. (2020) build a customized Hi-En code mixed embedding to generate Hi-En textual feature and finetuned with CNN and BILSTM along with attention layer in stacked arrangement to analyze sentiment. Joshi et al. (2016) proposed subword LSTM for HI-EN sentiment analysis. In, Subword LSTM architecture, LSTM is stacked on top of 3-gram char CNN in order to fetch sequential features considering local 3-gram as words. Paul et al. (2022) concatenated pretrained English word2vec and pretrained Hindi word2vec to initialize the network. Furthermore, they used ensemble of multilayer Perceptron, CNN, BiLSTM, BERT Devlin et al. (2018) to detect cyberbullying in code switched data in covid19 scenario. However, it is observed that pretrained word embedding cannot allocate vector to the words present in Hinglish or romanized Hindi language. Malte and Ratadiya (2019) has applied pretrained Mbert for the classification of aggression. They finetune MBert with 1 neural network layer and applied softmax for classification. However, this approach work well only if Hindi and English sentences share the same ordering of words.

Our approach differs from the related work significantly since CNN based approaches emphasize on local features while RNN based approaches is focused to sequential feature extraction. Most of the combined approaches extract local and sequential features but are aligned in a stacking manner which increases the depth of the framework. Past few works Too et al. (2019) Abuqaddom et al. (2021) demonstrated that deep framework suffers from the problem of vanishing gradient and the accuracy degrades as the network layers increases. Hence, we have proposed a fusion of local and sequential features, which increases the breadth of the network without increasing the depth of the network and produces a comprehensive representation focused on relevant local and global components.

3 Proposed methodology

3.1 Problem definition and formulation

A comment posted on social media can be termed as offensive if it conveys insult or shame. Offensive content can be defined as follows.

Definition 1 Given a supervised dataset having N number of labelled comments. A comment $c \in N$ consists of sentences and words in Hi-En code-mixed or Hinglish languages. Assume a comment c_i consists of j number of sentences $(1 \le s_{ij} \le j$) and each s_{ij} is represented by w_{ij} number of words. The definition not only include abusive sentence as offensive, but it also considers sentences that are not abusive

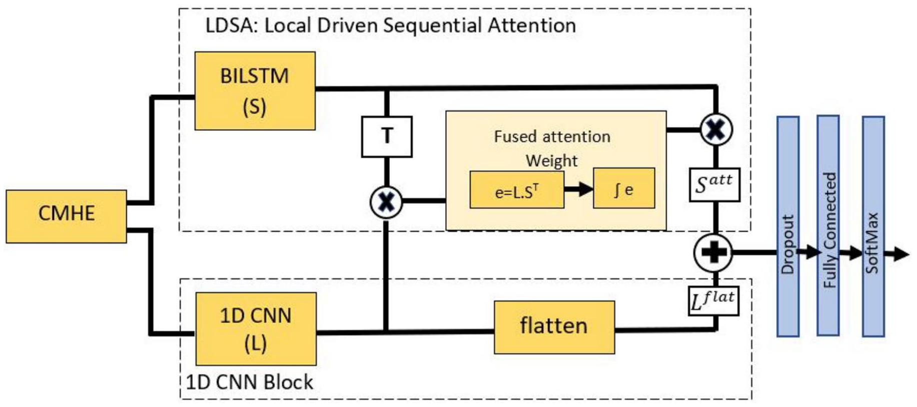


Fig. 1 FA-Net architecture

but insulting as offensive. These sentences are often misunderstood as non-offensive due to absence of any insulting words. Thus, the objective of this work is to develop a method M that can understand Hi-En code-mixed and correctly classify these comments.

3.2 Fused attention based network (FA-Net)

As shown in Fig. 1, overall framework of FA-Net comprises three building blocks as Code Mixed Hybrid Embedding (CMHE), Locally Driven Sequential Attention-2 (LDSA-2) and Locally Driven Sequential Attention-3 (LDSA-3). CMHE aim to assigns a meaningful weight vector to each word present in Hi-En code-mixed sentences. It is based on word2vec (continuous bag of word) and fasttext word embedding architecture. CMHE is shared in LDSA-2 and LDSA-3 block. LDSA-2 and LDSA-3 block capture the long dependency sequential feature along with relevant local 2-gram and 3-gram features. After this, LDSA-2 and LDSA-3 are concatenated and fed to a fully connected layer. Finally, softmax layer is employed for classification purpose.

3.2.1 Step by step processing of FA-Net

1. Construct CMHE by concatenating word and character n-gram embedding for each vocabulary words;
2. Context Sequential Features (S) : Employ BILSTM on CMHE to obtain contextual features for forward and backward text sequences;
3. Local Semantic Features (L2) : Employ 1D Convolutional layer with filter size 2 on CMHE to extract local 2-gram based features;
4. Local Semantic Features (L3) : Employ 1D Convolutional layer with filter size 3 on CMHE to extract local 3-gram based features;
5. LDSA2: Construct attention based sequential features (Satt2) by fusion of context sequential feature (S) and local semantic feature (L2) and employ self-attention mechanism to it in order to extract relevant sequential feature;
6. LDSA3: Construct attention based sequential features (Satt3) by fusion of context sequential feature (S) and local semantic feature (L3) and employ self-attention mechanism to it in order to extract relevant sequential feature;
7. Concatenate the final representation obtained by LDSA2 (step 5) and LDSA3 (step 6) to create a comprehensive representation as R= [Satt2, Satt3] ;
8. Feed the comprehensive representation (R) to fully connected layer using ReLU activation function and employ softmax classifier to get the class labels;
9. While training, learn weights using cross-entropy loss function through Adam optimizer.

Further, Sect. 3.3 to 3.7 explain each component of FA-Net in detail.

3.3 Code-mixed hybrid embedding (CMHE)

In this section, we have demonstrated the process to build code-mixed hybrid embedding using Hi-En code-mixed unsupervised corpus. The past few works Ma et al. (2019) Dharma et al. (2022) Singh et al. (2022) have shown that pretrained word embedding has significantly improved the performance in comparison to random embedding. In other words, initializing a network with a meaningful weight is better than random initialization. However, no such pretrained embedding is available for nonstandard language like Hi-En code-mixed. Therefore, to initialize our proposed network (FA-Net) with meaningful weights, we developed CMHE. Further, the process to build CMHE is demonstrated in sequence as data collection, data preprocessing, and construction of CMHE, as shown in Fig 2.

Table 1 Characteristics of collected data to build CMHE

Dataset characteristics	Size
Number of tweets	1,35,000
Total number Of words	27,99,402
Corpus vocabulary	2,09,093

3.3.1 Data collection

In order to accumulate a bulk amount of Hi-En code-mixed data, we utilized 102 offensive and hate-inducing words provided by Singh et al. (2018) as seed words e.g. kutte, harami, gadha etc. Furthermore, we used these seed words and GetOldTweets3 API (GetOldTweets3 0.0.11 2019) to collect tweets from Twitter. We have used seed words in Hinglish only in order to retrieve maximum comments in Hinglish. The region for scraping was set to India to restrict the scraping of similar words in different languages. All tweets are scraped in timeline of 2020-01-01 to 2020-06-01. Finally, we have collected 1,35,000(one lakh thirty-five thousand) tweets having 27,99,402(twenty-seven lakh ninety-nine thousand four hundred two) total words and 2,09,093(two lakh nine thousand ninety-three) unique words, as shown in Table 1. After this, the collected data is preprocessed to eradicate irrelevant data.

3.3.2 Data preprocessing

Since collected data comprises user-generated content, it contains a lot of irrelevant text that is not required for classification. Therefore, we followed the given steps to preprocess collected unsupervised corpus.

i We used the Indictrans Bhat et al. (2015), a transliteration library to convert Devanagari script to Romanized Hindi script.
ii We utilized regular expressions to eliminate all URLs and numeric data .
iii All emoticons were removed because they were misleading the actual information.
iv All words were converted to lowercase so that, similar words with spell variation could be assigned to the same word vector. For instance, Atanki, ATANKI converted to atanki.
v All of the @(e.g., @ xyz) stated have been condensed to a common term as user.
vi Stop words were not removed because they can disrupt the grammatical flow of the language.
vii Elongated words, commonly used to describe screaming expressions, were shortened to their normal form. For instance, wowwwww to wow.

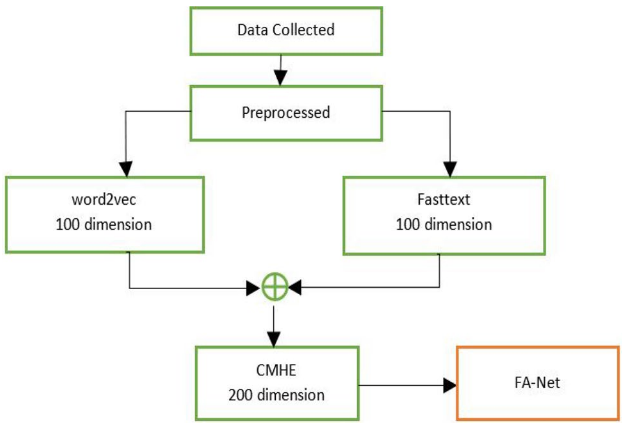


Fig. 2 Construction of CMHE

3.3.3 Construction of CMHE

Word2vec is proficient in extracting contextual knowledge at the word level and fasttext can capture morphological variation since it is based on char-n gram; therefore, to build CMHE, we trained word2vec and fasttext separately, with above collected data. For training, gensim library (GENSIM 2011) has been used. To generate contextual representation at word level, continuous bag of word (CBOW) model of word2vec with 100 dimensions and window size 5 are used as parameter Mikolov et al. (2013). For capturing spelling variation, fasttext is trained with 100 dimensions and char 3-gram as parameters Bojanowski et al. (2017). Furthermore, the resultant vector of word2vec and fasttext embedding is concatenated to obtain code mixed hybrid embedding as shown in Fig. 2. After concatenation, the resultant CMHE of 200 dimensions is used as the embedding layer to initialize the FA-Net with relevant weights.

3.4 Context sequential feature (S)

Long Short Term Memory belong to the family of Recurrent Neural Network (RNN) and is widely applied for extracting long dependency feature in text sequence Hochreiter and Schmidhuber (1997). However, single directional LSTM is not sufficient to extract feature of backward sequence. Hence, BiLSTM have been employed to extract feature from forward as well as backward direction. At last, the output vector of both directions is concatenated at each time step t. To begin with, LSTM is broadly divided into 4 components. The first is (I), which stands for an input gate that regulates the amount of memory that can be added. The forget gate (F), on the other hand, keeps track of memory that needs to be forgotten and helps to consume memory for relevant data only. The output gate (shown by O) modulates the output memory content, whereas cell memory (represented by C)

Table 2 Equations of LSTM

$$F_t = \sigma(p_t * W_{xf} + h_{t-1} * W_{hf})$$
$$c_t = tanh(p_t * W_{xc} + h_{t-1} * W_{hc})$$
$$I_t = \sigma(p_t * W_{xi} + h_{t-1} * W_{hi})$$
$$O_t = \sigma(p_t * W_{xo} + h_{t-1} * W_{ho})$$
$$C_t = F_t * C_{t-1} + I_t * c_t$$
$$h_t = O_t * tanh(C_t)$$

recall values over an arbitrary time interval. Initially, let d be the dimension of hidden state of one directional LSTM. The hidden state $h_t \in M_d$ at time step t in single directional LSTM. The mathematical form of LSTM is shown in Table 2.

Here, p_t is the input sequence at time step t and W is the learning weight of hidden layers. F_t, C_t, O_t, I_t are the representation of F, C, O, I at time step t. $\sigma(..)$ and tanh (..) signifies sigmoid and hyperbolic tangent activation function. Further, we have demonstrated the concatenation of forward and backward hidden states in BiLSTM.

Given the input sentence, P = p_1 , p_2 ,p_3 ....p_T , the hidden states h_f and h_b of the forward and backward LSTM outputs, respectively, are computed in both directions. They are concatenated as $s_t = [h_f;h_b]$ and used as a word representation of each word token. Here t=1,2,3 ....T and s_t represent sequential dependency at time step t. After this, all the calculated hidden state are placed into a matrix which is defined as $S = [s_1, s_2, s_3, ...s_T]$.

Here, $S \in M^{T\times2d}$ where T represents the rows and indicates the sequential dependency in the input sequence at corresponding position and d is the dimension of hidden state in LSTM. Dimension (d) is transformed to twice due to the concatenation of forward and backward hidden states in Bidirectional LSTM.

3.5 Local semantic feature (L)

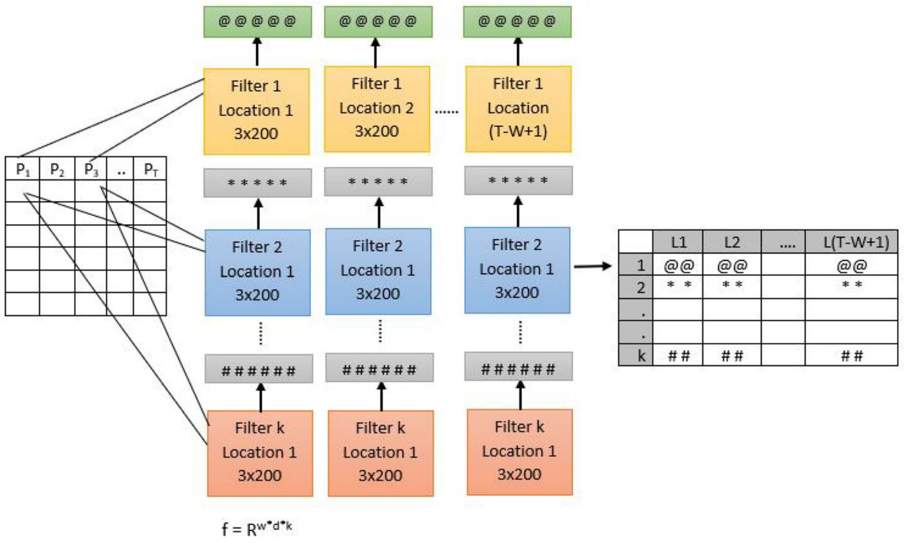
As shown in Fig. 3, a one-dimensional convolutional neural network (1D CNN) is employed to extract local semantic features from a textual sequence Zhang and LeCun (2015). In this, convolutional operation takes place in one direction. It involves filter vector f as $f \in R^{w\times d\times N_f}$. Here, R is a real number system, w represents the width of convolutional filter, d is the height of filter, which is correspondent to the dimension of each word in word vector (CMHE, 200 dimension) and K represents the number of convolutional filters, respectively. The filter vector (f) slide over text sequences $P = p_1, p_2, p_3....p_T$ and perform convolution operation simultaneously, as shown in Equ. 1.

$$L_i = z(W_f * P_{i:i+w-1} + b) \tag{1}$$

Here, * is the convolution operation. W_f is the weight associated with each filter getting convolved with window $P_{i:i+w-1}$ and generating feature vector L_i at i^{th} location. Bias vector is represented by b, and z(.) signifies nonlinear activation function, rectified linear unit (ReLU). After multiple convolutions, contextual vector generated at i^{th} word location can be stated as $L_i = L_i^1, L_i^2, L_i^3, L_i^4, ...L_i^k \in R^k$ Where L_i^k is feature vector generated using k^{th} filter at i^{th} location word. Here $1 \le i \le T - w + 1$. At last, contextual feature vector generated for complete sequence of length T words can be expressed as $L_u = [L_1, L_2, ..., L_{T-w+1}] \in R^{(T-w+1)\times K}$.

3.6 Locally driven sequential attention (LDSA-2)

In LDSA, sequential attention features (S^{att2}) are obtained from the contextual sequential feature (S) guided by local semantic feature (L_2) with filter width 2 , as shown in Equ.



@@@@@
Filter 1 Location 1 3x200
Filter 1 Location 2 3x200
Filter 1 Location (T-W+1)
......
*****
Filter 2 Location 1 3x200
######
Filter k Location 1 3x200
P1 P2 P3 .. PT
L1 L2 .... L(T-W+1)
1 @@
2 * *
k ##
f = R^{w*d*k}

Fig. 3 1D convolutional neural network architecture

4. In particular, the dot product of S and L_2 is computed, as shown in Equ. 2. After this, a self-attention mechanism Bahdanau et al. (2015) is employed to extract relevant weights from sequential features based upon local features, as shown in Equ. 3.

$$e = dot(L_2, S^T) \tag{2}$$

$$\alpha_{ij} = \frac{exp(e_{ij})}{\Sigma_{k=1}^{T_x} exp(e_{ik})} \tag{3}$$

$$S^{att2} = \alpha_{ij}.S \tag{4}$$

Here, α_{ij} is fused attention weight extracted by the mutual combination of sequential and local features. S^T is the transpose of sequential feature matrix and L is the local feature matrix with filter width 2.

3.7 Locally driven sequential attention (LDSA-3)

In LDSA-3, sequential attention features (S^{att3}) are obtained from the contextual sequential feature (S) guided by local semantic feature (L_3) with filter width 3 as shown in Equ. 7. In particular, dot product of S and L_3 is computed as shown in Equ. 5. After this, a self-attention mechanism is employed to extract relevant weights from sequential features based upon 3-gram local features, as shown in Equ. 6.

$$g = dot(L_3, S^T) \tag{5}$$

$$\beta_{ij} = \frac{exp(g_{ij})}{\Sigma_{k=1}^{T_x} exp(g_{ik})} \tag{6}$$

$$S^{att3} = \beta_{ij}.S \tag{7}$$

Here, β_{ij} is fused attention weight extracted by the mutual combination of sequential and local features. S^T is the transpose of sequential feature matrix and L is the local feature matrix with filter width 3.

After this, the output vector of LDSA-2 and LDSA-3 are concatenated in order to increase the feature space by combining high level and low level features as suggested by Du et al. (2020). Finally, a comprehensive context representation is obtained as, $R = [S^{att2}, S^{att3}]$ which is fed to fully connected layer using ReLU activation function. At last, softmax layer is employed to calculate the conditional probabilities over the label space to achieve classification. Currently, cross-entropy is a widely used loss function to measure the classification performance of model. In our model, adam optimizer is selected to optimize the loss function of the network. Training of FA-Net is based on the concept of backpropagation algorithm in which weights are adjusted until it reaches minimum loss. Hence, to analyze error or loss, we have employed cross-entropy as loss function and adam optimizer to optimize the loss function.

The main contribution and originality of FA-Net are as follows:

1. CMHE trained on large unsupervised corpus at word and character level for gathering contextual knowledge and capturing spelling variation, respectively. Further, FA-Net is regularized with CMHE, which in turn initializes the network with relevant weights.
2. BiLSTM is employed to extract low dimensional sequential features considering forward and backward sequences. Convolution structure is integrated to extract local n-gram based semantic features from the sentence.
3. In prior work, the attention mechanism employed with BiLSTM focuses on relevant sequential features, while the attention mechanism with CNN focuses on extraction of local features only. In addition, stacking of CNN LSTM increase the depth of the network and diminish the mutual contribution of local and sequential feature. Hence, in this work, fusion of sequential attention feature driven by the local n-gram, results in, extraction of mutually significant representation. Moreover, the fused attention mechanism in FA-Net makes the understanding of text semantics more accurate since it obtains diversified features.

4 Experiments and results

Experiments are carried out on two benchmark datasets in order to assess the efficacy of the proposed methodology for the classification of offensive text. In this section, dataset description, baseline models followed by results and discussion are demonstrated.

4.1 Dataset description

Experiments are performed on benchmark datasets: TRAC 2- 2020 Hindi English code-mixed dataset Bhattacharya et al. (2020) and hate speech dataset Bohra et al. (2018).

4.1.1 TRAC 2-2020

Trolling, Aggression and Cyberbullying (TRAC) is a multiclass supervised dataset. Its training set contains 3984, and the test set contains 1200 YouTube comments in Hi-En code-mixed language. Each sentence is annotated in one of 3 classes as Covertly Aggressive (CAG), Non-Aggressive (NAG), and Overtly Aggressive (OAG).

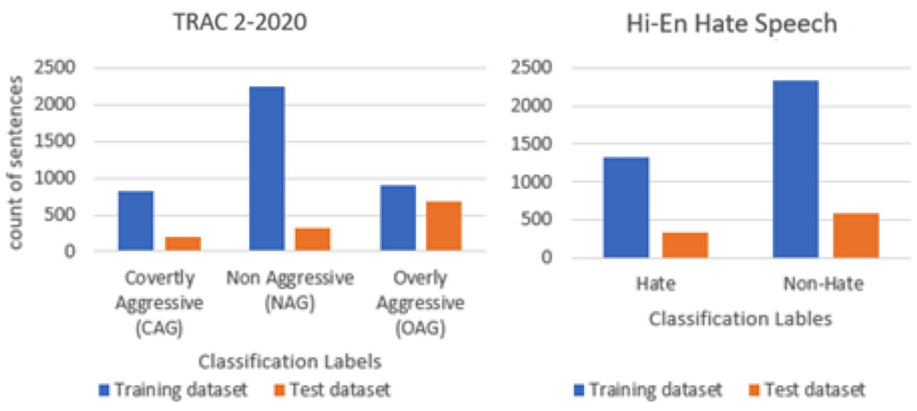


Fig. 4 Statistical visualization of trac-2 2020 and Hi-En hate speech datasets

4.1.2 Hate speech

It is a binary dataset having Hi-En code mixed text. In this dataset, each row is classified in one of the two classes, as Hate and Non-Hate. Since there is no standard test set available thus, we used stratified sampling to split the whole dataset in 20 % and 80 % as test and training dataset respectively. In the training dataset, 1328 Comments belong to Hate, and 2331 belongs to Non-Hate. Along with this, their statistical visualization and characterization is also shown in Fig. 4 and Table 3.

4.2 Baseline models

We compared FA-Net with some baseline models. A brief description of these models are mentioned below:

- *Logistic Regression*: In this, weighted tf-idf based unigram and bigram sequence of word is adopted to construct feature vector.Further, it is classified using logistic regression.

- *Xtreme Gradient Boosting (XGBoost)*: Using this method, weighted tf-idf based unigram and bigram word sequential feature vector is classified using tree based ensemble XGBoost classifier.

- *Char-CNN* Kim (2014):Each character in character vocabulary is randomly initialized with 100 dimension and later finetuned using CNN with filter size 3. This approach ignore sequential dependency between words.

- *Word-CNN* Kim (2014): To construct feature at word level, each word is initialized randomly with 100 dimension and finetune it using CNN of filter size 2.

- *Subword LSTM* Joshi et al. (2016) :In this network, the LSTM layer is stacked on top of the CNN layer in order to fetch local and sequential feature.Hence, we used character 3-gram CNN to build subwords and then fed to LSTM to develop a sequential feature for long text sequences.

- *Fine-tune M-BERT* Pires et al. (2019): M-Bert has already been trained in 104 languages, including Hindi and English.We used M-Bert on experimental datasets and finetuned it with a fully connected neural network and a 20% dropout to avoid overfitting to examine its performance on Hi-En code-mixed data.

4.3 Evaluation metrics

To analyse the performance of the proposed model, we adopted weighted average precision, recall, f1 score and accuracy as evaluation metrics Novaković et al. (2017). We calculated these metrics[1] using classification report feature of scikit-learn library Pedregosa et al. (2011). Further, these metrics are briefly explained below.

Accuracy:Accuracy is defined as total number of correctly classified data to overall data as shown in Equ. 8. However, this metrics is not enough to measure performance if data is imbalanced.Hence, we have focused on weighted average f1 score, precision and recall.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

Precision: Ratio of correctly classified as positive to the total predicted as positive, as shown in Equ. 9. It mainly focuses on the correct identification of positive input.

Table 3 Statistical characterization of datasets

Dataset 1	Training set	Test set
Covertly aggressive(CAG)	829	191
Non aggressive(NAG)	2245	325
Overtly aggressive (OAG)	910	684
Average length of sentence	16.06	20.31
Dataset 2		
Hate	1328	332
Non-hate	2331	583
Average length of sentence	90	87

[1] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.

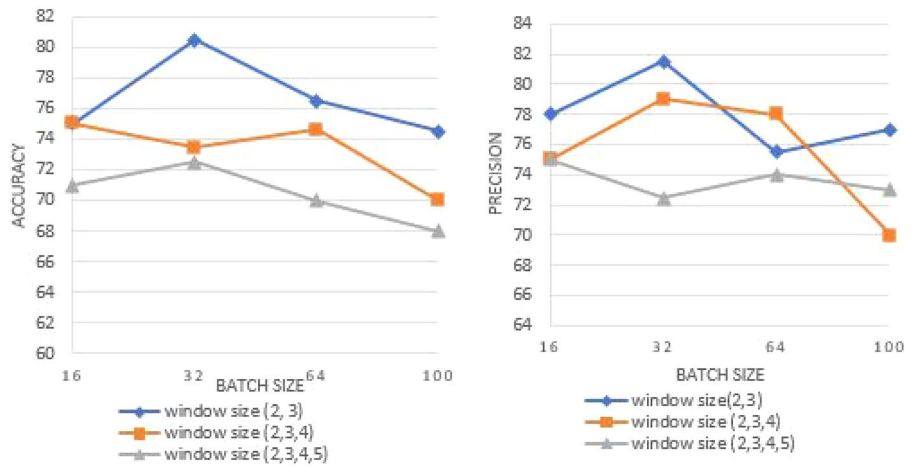


Fig. 5 Experimental Results of FA-Net with various hyperparameters

$$precision = \frac{TP}{TP + FP} \tag{9}$$

Recall : Ratio of correctly classified as positive to the total actual positive, as shown in Equ. 10.

$$recall = \frac{TP}{TP + FN} \tag{10}$$

f1-Score: f1-score is the harmonic mean of the precision and recall as shown in Equ. 11. Hence, it is recommended during experimentation with asymmetrical dataset.

$$f1 - score = \frac{2 \times precision \times recall}{precision + recall} \tag{11}$$

Here, TP= True Positive; FP= False Positive; TN= True Negative; FN= False Negative.

4.4 Experimental setup and parameter setting

Firstly, datasets are preprocessed using steps mentioned in Sect. 3.3 (step 2). After preprocessing, a vocabulary index is used to transform each sentence into a numeric sequence. In order to maintain an equal length of input numeric sequence, padding method is used. Since we do not want to lose contextual information of lengthy text, we select the maximum length parameter as 50 words for TRAC 2-2020 and 90 words for hate speech dataset, considering the average length of the dataset as mentioned in Table 3. According to this, if a sentence length is less than the maximum length, prepadding is used, and if the sentence is longer than the maximum length, pruning is done at the beginning. For experiment purposes, a well-known python library (Keras 2015) was used with tensorFlow Abadi et al. (2016) as a backend, and scikit-learn library Pedregosa et al. (2011) is used for machine learning models. We performed 5-fold cross-validation on the training dataset and evaluated the final model on the test datasets. FA-Net was trained for eight epochs, which was found to be nearly optimal after several experiments. As discussed in Sect. 3.3, Our model used CMHE of dimension 200 to initialize the network. The memory dimension of BiLSTM is set to 120. We used 120 number of filters to extract local features. The memory dimension of the BLSTM for each dataset is set the same as the number of filters in order to apply fusion in local representations with intermediate sentence representation. A backpropagation algorithm with adam optimizer and categorical cross-entropy loss function is used to train the network with a learning rate of 0.001 and a dropout rate of 0.2. Zhang and Wallace (2017) claimed that filter width size and batch size is important parameter and can affect the performance of the model; thus, to select right batch size and filter width size, TRAC-2 2020 dataset is experimented using FA-Net. We reused the discussed parameter and analysed validation accuracy and precision of FA-Net for (2,3), (2,3,4), (2,3,4,5) filter width sizes as shown in Fig. 5.

From the result, following findings have been observed.

- Validation accuracy of FA-Net with window size (2,3) is significantly better than other window size irrespective of batch size. Furthermore, the highest accuracy is attained with batch size 32 and for filter width (2,3).
- Precision of FA-Net with filter width (2,3) is significantly better than other window sizes. For filter width (2,3) the highest precision is attained for batch size 32.
- It also shows that probability of co-occurrence of 2 words and 3 words are frequent in pattern. In addition, degradation in performance is observed while increasing the filter width, which depict that the presence of longer sequential pattern are rare in text.

Thus, for the performance analysis of FA-Net, filter window size (2,3) and batch size 32 have been selected in this work.

Table 4 Comparison results of the proposed approach with baseline models (in %)

Approaches	trac-2,2020				Hate speech			
	Precision (weighted avg)	Recall (weighted avg)	Accuracy	f1-score (weighted avg)	Precision (weighted avg)	Recall (weighted avg)	Accuracy	f1-score (weighted avg)
Logistic regression	75.78	70	70	69.93	64.78	68.89	69	65.01
XGBoost	73	67	66.87	66.89	66.98	68.89	69	66.98
Naïve Bayes	77.66	47	50	47	73	70	70	65
Char CNN	64.39	60.41	60.41	69.30	61	60	60	61
Word CNN	62.06	60.23	60.21	60.17	56	57	58	56
Subword LSTM	63.06	61.23	61.23	60.87	76.63	47.89	69.80	47.87
Finetuned MBert	63.33	73.66	72	72	68.89	71.63	71	71
CMHE-$BiLSTM_{attn}$	76.50	77.32	77.54	77.09	70.09	74.12	75.23	73.34
CMHE-CNN_{attn}	74.5	76.32	75.54	75.89	74.69	74.92	74.63	74.44
FA-Net	**82.89**	**81.56**	**81.09**	**82.67**	**80.76**	**81.56**	**80.89**	**81.35**

Bold face indicates the best value

4.5 Results and ablation study

The proposed methodology discussed in Sect. 3 and baseline models discussed in section 4.2 has been used to examine the FA-Net performance. Table 4 shows the comparative results of FA-Net against baseline models. From the results, it can be inferred that:

- Proposed model FA-Net has outperformed other baseline models. It has attained an accuracy of 81.09%, 80.89%, and f1 score of 82.67%, 81.35% for TRAC-2,2020 and Hate Speech datasets. It has been observed that initializing network with relevant weights and comprehensive representation of local and sequential features has significantly improved the performance.
- Naive Bayes produced a high precision but a low recall of 47 percent for TRAC 2,2020. As a result, the model is underfitting and getting biased in favor of the majority class.
- Subword LSTM earned high precision with low recall and low f1 score for hate speech. However, the model has not performed significantly well for TRAC-2, 2020.
- Classical machine learning models, logistic regression, and the XGBoost ensemble model outperformed random embedding-based deep learning methods; thus, a neural network initialized with random weights does not converge well. In both datasets, random weight vectors did not surpass the statistical feature model performance.
- Finetuned MBert, a transformer-based model, performed comparatively better than classical ML model and random embedding models in terms of recall, accuracy, and F1 score. It attained 72 percent and 71 percent of f1 score for TRAC 2-2020 and hate speech datasets, respectively. [28] demonstrated that Pretrained MBert is based on extraction of cross-lingual features from multiple monolingual corpora in parallel. However, it performed poorly, when two language lacks similar word sequencing. This disruption of word sequencing phenomena occurs in Hi-En code mixed language due to repetitive switching between Hindi and English, resulting in a significant drop in Mbert's Performance.
- CMHE with BiLSTM attention and CMHE with CNN attention has performed comparatively better than other baseline model, which shows that initializing a network with relevant weight is crucial. Specifically, in code mixed Hi-En language, word and character n-gram based vectors supported in capturing contextual and morphological variation in text.

Further, we have conducted an ablation study to quantify the contribution of each component of FA-Net on TRAC-2,2020 and Hate Speech datasets.

W/O CMHE: FA-Net is initialized randomly and finetuned with a fused attention mechanism.

W/O LDSA-3: In this model, FA-Net is initialized with CMHE . After this, it's associated weights are finetuned by fusion of BiLSTM and local 2-gram based attention representation.

W/O LDSA-2: In this model, FA-Net is initialized with CMHE. After this, learning weights are updated using the fusion of BiLSTM and local 3-gram based attention representation.

As indicated in Table 5, performance of an individual ablative model in terms of weighted f1 score and comparison to FA-Net is depicted by (delta), δ= FA-Net - $ablativemodel_i$.

Table 5 Ablation studies on a different component of the proposed model (CMHE-AN) in terms of weighted average f1 score

	W/O CMHE	δ	W/O LDSA-3	δ	W/O LDSA-2	δ	FA-Net	δ
Trac-2,2020	72	(+10.67)	79.73	(+2.94)	78.88	(+3.79)	82.67	–
Hate speech	71.45	(+9.9)	78.56	(+2.79)	77.89	(+3.46)	81.35	–

Table 6 Comparative performance against the state of art

Source	Method adopted	Features/embedding	f1-score(trac-2,2020)	Accuracy (hate speech)
Datta et al. (2020)	Gradient Boosted Machine	Concetenation of tf-idf, aggressive word lexicon, sentiment score, part of speech	59.45	–
Santosh and Aravind (2019)	Hierarchical attention LSTM	Phonic subword embedding	–	66.6
Kumari et al.(2021)	3 layer of LSTM autoencoder	Random embedding	74	–
Bohra et al.(2018)	Support vector machine (rbf)	Char-ngram, linguistic feature	–	71.7
Koufakou et al. (2020)	LSTM applied to merged augmented code mixed datasets	Random embedding	72.6	–
Mathur et al. (2019)	Transfer learning using CNN	Transliteration to English and transfer of supervised feature		71
Kapil and Ekbal (2020)	Multitask learning	Code Mixed hybrid Embedding (CMHE)	74.98	72.76
Majumder et al. (2022)	BiLSTM	(Word+character) Random embedding	70.6	69.56
Paul et al. (2022)	Ensemble of CNN+LSTM +BERT	Word2vec (Hi)+ word2vec(Eng)	75.89	73.76
Paul et al. (2022) using FA-Net	BiLSTM-CNN fused attention network	Word2vec(Hi)+ word2vec(Eng)	76.12	73.67
FA-Net(proposed)	BiLSTM-CNN fused attention network	Code Mixed hybrid Embedding (CMHE)	**82.67**	**80.89**

Bold face indicates the best value

W/O CMHE reveals that initializing the FA-Net with relevant weights has significantly improved the performance by 10.67% and 9.9% in terms of F1 score. As discussed in Sect. 1 (challenges), Hi-En code-mixed suffers from spell variation problem; therefore, CMHE based on word2vec and fasttext mapped similarly spelled and contextually related words in closed proximity. In other words, similarly spelled words have been assigned a similar vector representation results in a meaningful representation of the sentence. W/O LDSA-3 and W/O LDSA-2 observed a significant decrease in f1 score, which shows that local 2 and 3-gram combinely enable FA-Net to extract more efficient comprehensive representation than 2-gram and 3-gram separately.

4.6 Comparison with the existing state of the art

In this section, our model (FA-Net) classification performance is compared with the existing state-of-the-art, as given in Table 6. The results of Datta et al. (2020), Santosh and Aravind (2019), Kumari et al. (2021), Bohra et al. (2018), Koufakou et al. (2020) are extracted from their original papers due to unavailability of manual dictionaries and code. Mathur et al. (2019)and Kapil and Ekbal (2020) has been reimplemented with publicly available code. For Majumder et al. (2022) Paul et al. (2022), they were implemented by ourselves and best results are reported. From Table 6, it is observed that Datta et al. (2020) has used statistical and linguistic features along gradient boosted classifier. Their model reported 59.45% of f1-score on TRAC-2, 2020 dataset. This approach did not work well because of the absence of semantic features. Santosh and Aravind (2019) used subword embedding at phonic level and used hierarchical LSTM to extract sequential, but performed poorly with 66.6% on hate speech dataset. Kumari et al. (2021) used the concept of reconstruction loss to classify TRAC-2 2020 and attained f1 score of 74%. Kapil and Ekbal (2020) used a multitasking approach to gather combined features from multiple datasets. However, on experimenting with this approach, we observed contamination of features, which in turn decreased the performance. As explained by Majumder et al. (2022), we concatenated randomly initialized word and character embedding and used BiLSTM to extract sequential features, but the attained result are not satisfactory. As demonstrated by Paul et al. (2022), we concatenated pretrained Hindi word2vec and pretrained English word2vec and experimented with ensemble of multiple layer perceptron, CNN, BiLSTM, BERT and achieved 75.89% f1 score and 73.76% accuracy for TRAC-2,2020 and Hate Speech datasets. We have used the same embedding to train FA-Net and obtained result of 76.12% accuracy and 73.67% f1-score which is comparable to Paul et al. (2022). Thus, we observed that pretrained embedding could not capture Hinglish nonstandard words, which in turn decreased the performance of both the networks. From Table 6, it can be observed that the proposed model (FA-Net) trained on CMHE has outperformed against the existing state-of-the-art model.

5 Conclusion and future work

In this paper, a new fused attention mechanism-based model (FA-Net) is developed in order to extract comprehensive representation of sequential and local features. In order to overcome the challenge of spelling variation and code mixing, code-mixed hybrid embedding (CMHE) is constructed using a customized corpus. After this, CMHE is used to initialize FA-Net with relevant weights instead of random initialization. In this work, the bidirectional long-short term memory model is employed to create a transitional sentence representation, which is then used to generate attention weights in a sequential manner. The attention weights enable to extract the most crucial information contained in the sentence. Moreover, merging the bidirectional long-short term memory with the convolutional network equips the model to gather comprehensive information, such as past, forthcoming, and local context, from any location in a sequence. It is capable of retrieving the most significant and comprehensive information present in a sentence. Experimental results demonstrate that the proposed model outperformed current state-of-the-art approaches on two benchmark datasets for offensive text classification. Our model will map Hi-En code mixed text to dense, low-dimensional space and have knowledge of emotion, insulting, hate-related text; therefore, it can substantially contribute to emotion classification, sentiment analysis, toxicity analysis from the perspective of deep semantics. In addition, future study will concentrate on the following points.

- Addition of user meta information, user pattern, and behaviour in posting offensive content can be incorporated as a feature into the model.
- So far, the problem is interpreted as supervised task, but unsupervised and semi-supervised learning can be explored to leverage large amount of Hi-En code-mixed social media content.
- Integration of multimodal features such as image and text can be explored to enhance contextual information.

Data availability The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

Abadi M, Agarwal A, Barham P, et al. (2016) TensorFlow: large-scale machine learning on heterogeneous systems. Software available from tensorflow.org. https://www.tensorflow.org/

Abuqaddom I, Mahafzah BA, Faris H (2021) Oriented stochastic loss descent algorithm to train very deep multi-layer neural networks without vanishing gradients. Knowl Based Syst 230:107391. https://doi.org/10.1016/j.knosys.2021.107391

Badjatiya P, Gupta S, Gupta M, Varma V (2017) Deep learning for hate speech detection in tweets. 26th International World Wide Web Conference 2017, WWW 2017 Companion, 759–760. https://doi.org/10.1145/3041021.3054223

Bahdanau D, Cho KH, Bengio Y (2015) Neural machine translation by jointly learning to align and translate. In: 3rd International Conference on learning representations, ICLR 2015

Bali K, Sharma J, Choudhury M, Vyas Y (2015) "i am borrowing ya mixing ?"an analysis of english-hindi code mixing in facebook, 116–126. https://doi.org/10.3115/v1/w14-3914

Bhattacharya S, Singh S, Kumar R, *et al.* (2020) Developing a multilingual annotated corpus of misogyny and aggression. In: Proceedings of the second workshop on trolling, aggression and cyberbullying, pp. 158–168. European Language Resources Association (ELRA), Marseille, France. https://aclanthology.org/2020.trac-1.25

Bhat IA, Mujadia V, Tammewar A, Bhat RA, Shrivastava M (2015) Iiit-h system submission for fire2014 shared task on transliterated search. https://doi.org/10.1145/2824864.2824872

Bohra A, Vijay D, Singh V, Akhtar SS, Shrivastava M (2018) A dataset of Hindi-English code-mixed social media text for hate speech detection, 36–41. https://doi.org/10.18653/v1/W18-1105

Bojanowski P, Grave E, Joulin A, Mikolov T (2017) Enriching word vectors with subword information. Trans Assoc Comput Linguis 5:135–146. https://doi.org/10.1162/tacl_a_00051

Bulao J (2022) How much data is created every day in 2022? https://techjury.net/blog/how-much-data-is-created-every-day

Chollet F et al (2015) Keras. https://github.com/fchollet/keras

Datta A, Si S, Chakraborty U, Naskar SK (2020) Spyder: Aggression detection on multilingual tweets. In: Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying, language resources and evaluation Conference (LREC 2020, pp. 87–92. https://www.smartinsights.com/social-media-

Devlin J, Chang M-W, Lee K, Toutanova K (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference 1, 4171–4186

Dharma EM, Gaol FL, Leslie H, Warnars HS, Soewito B (2022) The accuracy comparison among word2vec, glove, and fasttext towards convolution neural network (cnn) text classification. J Theor Appl Inf Technol 100(2):31

Du C, Wang Y, Wang C, Shi C, Xiao B (2020) Selective feature connection mechanism: concatenating multi-layer cnn features with a feature selector. Pattern Recogn Lett 129:108–114. https://doi.org/10.1016/j.patrec.2019.11.015

Jefferson-Henrique (2019) CodeGetOldTweets3 0.0.11. https://pypi.org/project/GetOldTweets3/

James Ker-Lindsay (2022) Hinglish. https://en.wikipedia.org/wiki/Hinglish

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9:1735–1780. https://doi.org/10.1162/NECO.1997.9.8.1735

Joshi A, Prabhu A, Shrivastava M, Varma V (2016) Towards sub-word level compositions for sentiment analysis of Hindi-English code mixed text. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 2482–2491. The COLING 2016 Organizing Committee, Osaka, Japan. https://aclanthology.org/C16-1234

Kapil P, Ekbal A (2020) A deep neural network based multi-task learning approach to hate speech detection. Knowl Based Syst 210:106458. https://doi.org/10.1016/j.knosys.2020.106458

Kim Y (2014) Convolutional neural networks for sentence classification. EMNLP 2014 - 2014 Conference on empirical methods in natural language processing, Proceedings of the Conference, 1746–1751. https://doi.org/10.3115/v1/d14-1181

Koufakou A, Basile V, Patti V (2020) FlorUniTo@TRAC-2: Retrofitting word embeddings on an abusive lexicon for aggressive language detection. In: Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying, pp. 106–112. European Language Resources Association (ELRA), Marseille, France. https://aclanthology.org/2020.trac-1.17

Kumari K, Singh JP, Dwivedi YK, Rana NP (2021) Bilingual cyber-aggression detection on social media using LSTM autoencoder. Soft Comput 25(14):8999–9012

Majumder A, Paul A, Banerjee A (2022) Deep learning-based approach using word and character embedding for named entity recognition from hindi-english tweets, 237–243. https://doi.org/10.1007/978-981-16-7305-4_23

Malte A, Ratadiya P (2019) Multilingual cyber abuse detection using advanced transformer architecture. IEEE Region 10 Annual International Conference, Proceedings/TENCON 2019-Octob, 784–789. https://doi.org/10.1109/TENCON.2019.8929493

Mathur P, Shah R, Sawhney R, Mahata D (2019) Detecting offensive tweets in hindi-english code-switched language, 18–26. https://doi.org/10.18653/v1/w18-3504

Ma Q, Yu L, Tian S, Chen E, Ng WWY (2019) Global-local mutual attention model for text classification. IEEE/ACM Trans Audio Speech Lang Process 27:2127–2139. https://doi.org/10.1109/TASLP.2019.2942160

Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. Adv Neural Inf Process Syst 26:3111–3119

Monto MA, McRee N, Deryck FS (2018) Nonsuicidal self-injury among a representative sample of us adolescents, 2015. Am Journal Public Health 108:1042–1048. https://doi.org/10.2105/AJPH.2018.304470

Novaković JD, Veljović A, Ilić SS, Željko Papić, Milica T (2017) Evaluation of classification models in machine learning. Theor Appl Math Comput Sci 7:39–46

One Speaker, Two Languages (1995) Cross-disciplinary perspectives on code-switching. Cambridge University Press

Pasricha J (2016) Cyber violence against women in India - a research report. https://feminisminindia.com/2016/11/15/cyber-violence-against-women-india-report/

Patchin JW, Hinduja S (2018) Deterring teen bullying: assessing the impact of perceived punishment from police, schools, and parents. Youth Violence Juvenile Justice 16:190–207. https://doi.org/10.1177/1541204016681057

Paul S, Saha S, Singh JP (2022) Covid-19 and cyberbullying: deep ensemble model to identify cyberbullying from code-switched languages during the pandemic. Multimedia Tools and Applications, 1–17. https://doi.org/10.1007/S11042-021-11601-9/TABLES/8

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830

Pires T, Schlinger E, Garrette D (2019) How multilingual is multilingual BERT? In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 4996–5001. Association for Computational Linguistics, Florence, Italy. https://doi.org/10.18653/v1/P19-1493

Rehurek R, Sojka P (2011) GENSIM. https://radimrehurek.com/gensim/models/word2vec.html

Samghabadi NS, Mave D, Kar S, Solorio T (2018) Ritual-uh at TRAC 2018 shared task: Aggression identification. CoRR **abs/1807.11712** 1807.11712

Santosh TYSS, Aravind KVS (2019) Hate speech detection in hindi-english code-mixed social media text. ACM Int Conf Proc Ser. https://doi.org/10.1145/3297001.3297048

Sasidhar TT, B P, P SK (2020) Emotion detection in hinglish(hindi+english) code-mixed social media text. Procedia Computer Science 171, 1346–1352. https://doi.org/10.1016/j.procs.2020.04.144. Third International Conference on Computing and Network Communications (CoCoNet'19)

Sharma S, Srinivas PYKL, Balabantaray RC (2015) Text normalization of code mix and sentiment analysis. 2015 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2015, 1468–1473. https://doi.org/10.1109/ICACCI.2015.7275819

Shetty A (2008) India ranks third on global cyber bullying list. https://www.firstpost.com/tech/news-analysis/india-ranks-third-on-global-cyber-bullying-list-3602419.html

Singh V, Varshney A, Akhtar SS, Vijay D, Shrivastava M (2018) Aggression detection on social media text using deep neural networks. EMNLP 2018, 43. https://doi.org/10.18653/v1/w18-5106

Singh KN, Devi SD, Devi HM, Mahanta AK (2022) A novel approach for dimension reduction using word embedding: an enhanced text classification approach. Int J Inf Manag Data Insights 2:100061. https://doi.org/10.1016/J.JJIMEI.2022.100061

Si S, Datta A, Banerjee S, Naskar SK (2019) Aggression detection on multilingual social media text. 10th International Conference on computing, communication and networking technologies, ICCCNT 2019, 1–5. https://doi.org/10.1109/ICCCNT45670.2019.8944868

Too EC, Yujian L, Njuki S, Yingchun L (2019) A comparative study of fine-tuning deep learning models for plant disease identification. Computers and Electronics in Agriculture 161, 272–279. https://doi.org/10.1016/j.compag.2018.03.032. BigData and DSS in Agriculture

Zhang X, LeCun Y (2015) Text Understanding from Scratch. arXiv. https://doi.org/10.48550/ARXIV.1502.01710

Zhang Y, Wallace BC (2017) A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. In: Proceedings of the Eighth International joint conference on natural language processing (Volume 1: Long Papers), pp. 253–263