A Hybrid Deep Learning Architecture for Sentiment Analysis

Md Shad Akhtar, Ayush Kumar, Asif Ekbal, Pushpak Bhattacharyya
Department of Computer Science & Engineering
Indian Institute of Technology Patna, India
{shad.pcs15,ayush.cs12,asif,pb}@iitp.ac.in

Abstract

In this paper, we propose a novel hybrid deep learning architecture which is highly efficient for sentiment analysis in resource-poor languages. We learn sentiment embedded vectors from the Convolutional Neural Network (CNN). These are augmented to a set of optimized features selected through a multi-objective optimization (MOO) framework. The sentiment augmented optimized vector obtained at the end is used for the training of SVM for sentiment classification.We evaluate our proposed approach for coarse-grained (i.e. sentence level) as well as fine-grained (i.e. aspect level) sentiment analysis on four Hindi datasets covering varying domains. In order to show that our proposed method is generic in nature, we also evaluate it on two benchmark English datasets. Evaluation shows that performance of the proposed method are consistent across all the datasets and often outperform the state-of-art systems. To the best of our knowledge, this is the very first attempt where such a deep learning model is used for sentiment analysis in less-resourced languages such as Hindi.

1 Introduction

Sentiment Analysis (Pang and Lee, 2008) in natural language processing (NLP) deals with the problem of identifying the polarity in a user generated content. With growing social media platforms such as Twitter, Facebook etc., copious amount of data is being generated continuously. According to Domo's Data Never Sleep 2.0[1], the global internet population is about 2.4 billion users. Online platforms such as Twitter alone generate over 300,000 tweets per minute[2]. At the same time more than 26K user reviews are posted on Yelp, an online user review portal. This tremendous amount of semi-structured data poses a great challenge in its efficient processing for any specific purpose. Sentiment analysis for web generated content e.g. tweets and online reviews, is a cumbersome problem mainly due to its unstructured and noisy nature (e.g. *gr8*, *g8* etc. for *great*) and spelling and grammatical mistakes. Considering the challenges as mentioned above, authors have proposed their sentiment analyzers for Twitter data and/or online reviews (Kim and Hovy, 2004; Mohammad et al., 2013a; Gupta et al., 2015). However, most of the works have been done on the resource-rich languages such as English.

India is a multi-lingual country with great linguistic and cultural diversities. There are 22 officially spoken languages. However, there have not been enough research works that address sentiment analysis involving Indian languages, *except* few such as (Balamurali et al., 2012; Bakliwal et al., 2012; Kumar et al., 2015). However, these existing works do not address the fine-grained sentiment analysis at the aspect level. The prime reason behind this is the scarcity of benchmark datasets and other resources/tools in Indian languages. In our work, we focus on sentiment analysis in Hindi, the official language of India and the fourth most spoken language all over in the world. We make use of benchmark datasets released as part of a shared task on sentiment analysis in Indian languages (SAIL) for Twitter (Patra et al., 2015). Recently, we (Akhtar et al., 2016) have created a dataset for aspect based sentiment analysis

[1]https://www.domo.com/learn/data-never-sleeps-2

[2]http://aci.info/2014/07/12/the-data-explosion-in-2014-minute-by-minute-infographic/

(ABSA) (Pontiki et al., 2014) in Hindi. For sentence-level sentiment analysis we annotate these same set of reviews. Here, we evaluate our proposed approach for both coarse-grained (sentence based) and fine-grained (aspect based) sentiment analysis.

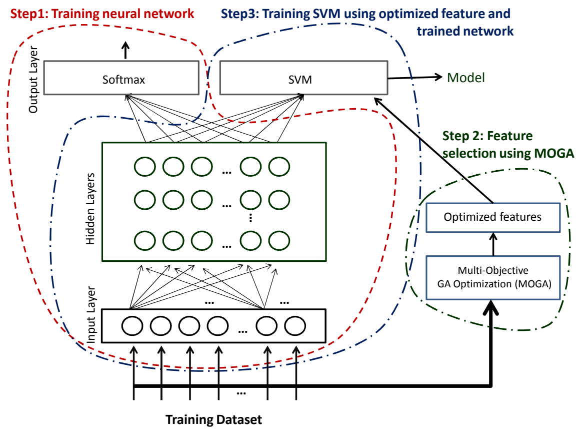
Our proposed method is based on deep learning, which has shown its premise in various NLP problems including sentiment analysis. Authors worldwide have proposed many variants of its architecture (Kim, 2014; dos Santos and Gatti, 2014), which have shown success for solving problems in varying domains. Most of these works employ traditional technique of using softmax as an activation function on top of a typical convolutional neural network (CNN). However, in our work we learn *sentiment embedded vectors* using CNN pipeline and perform final classification using a strong classifier, Support Vector Machine (SVM) (Vapnik, 1995). Replacing softmax layer with some stronger classifier might be useful as shown in very few research, such as computer vision (Tang, 2013) and NLP (Poria et al., 2015).

In this work, we do not use the traditional pipeline of CNN (c.f. Section 2.1) for sentiment classification. Rather, we learn sentiment features through CNN, which we call as '*sentiment-embedded vector*'. Parallely, a multi-objective optimization (MOO) based framework using Genetic Algorithm (GA) (Deb et al., 2002) is employed to derive optimized features for the respective optimization functions. In the final step, we augment the sentiment-embedded vector with the optimized feature set to form '*sentiment augmented optimized vector*'. This vector is used as the feature for sentiment classification using a non-linear SVM. In order to study the impact of external optimized handcrafted features, we build different models of the baseline systems. The existing works which make use of external features in CNN architecture simply append features at the input layer. This method has mainly three drawbacks: **(i)** The information in external features appended at the input layer are not properly reflected in the output due to the processes of convolution and pooling layers. **(ii)** The set of features is not optimized i.e, optimal subset of features is not extracted, instead complete feature set is appended to the word representations at the input layer. **(iii)** Softmax is a weak classifier and has limitation over SVM. We propose to tackle all these problems using our approach, the results of which are encouraging and consistent across datasets of varying domains and languages. Such hybrid model using CNN, SVM and MOGA (c.f. Section 2.3) that performs sentiment classification using sentiment augmented optimized vector is novel, impactful as well as very effective for resource-constrained languages.

We summarize the main contributions of the proposed approach as follows: **i)** a hybrid modified architecture of CNN, that learns sentiment embedded vector instead of traditional pipelined-classification; **ii)** application of MOO for the systematic selection of optimized feature set, to generate sentiment augmented optimized vector; **iii)** replacement of softmax layer to produce more robust hybrid deep learning network by using non-liner SVM based classification at the final step; and **iv)** generic approach, applicable to different languages and domains. We evaluate the approach on the datasets of varying domains, i.e. Twitter (generic as well as sarcastic) and online product reviews (sentence-level and aspect-level), across two different languages *viz.* Hindi and English for sentence-level as well as aspect-level sentiment analysis. Experiments show that the proposed hybrid deep learning architecture is highly efficient for sentiment analysis in multiple domains for Hindi. To the best of our knowledge, this is the very first attempt of using such a hybrid deep learning model for sentiment analysis, especially in less-resource languages. For English, we use the benchmark dataset of SemEval-2015 shared task on sentiment analysis in Twitter (Rosenthal et al., 2015) and SemEval-2014 shared task on aspect based sentiment analysis (Pontiki et al., 2014).

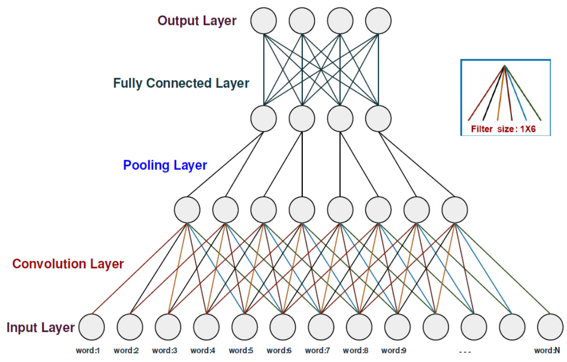
2 Methodology

Logistic regression (LR) (or Softmax regression for multi-class classification) and SVM are two algorithms that often produce comparable results. However, SVM has an edge over LR if the data is not linearly separable, i.e. SVM with non-linear kernel performs better than LR (Pochet and Suykens, 2006). Also, LR focuses on maximizing the likelihood and is prone to over-fitting. However, SVM finds a linear hyperplane by projecting input data into higher dimension and generalizes well. We incorporate this idea in our proposed research by replacing the softmax regression with SVM at the output layer of CNN. The motivation for using CNN architecture are two-fold: (i) The system can learn hidden semantics from a



Step1: Training neural network
Step3: Training SVM using optimized feature and trained network
Output Layer
Softmax
SVM
Model
Step 2: Feature selection using MOGA
Hidden Layers
Optimized features
Multi-Objective GA Optimization (MOGA)
Input Layer
Training Dataset

(a) Proposed methodology



(b) A typical architecture of CNN

Figure 1: (a) Proposed methodology. (b) A typical architecture of CNN.

large unlabeled corpus, and (ii) limited coverage of lexical resources (Hindi SentiWordNet). The proposed approach, *CNN-SVM*$_{W+X}$, operates in three steps (Figure: 1a; red, green & blue dotted lines show the processes of Step 1, 2 and 3, respectively.):

1. Learning sentiment embedded vector using CNN architecture;
2. Generation of sentiment augmented optimized vector using a multi-objective GA (MOGA) based optimization technique; and
3. Training of SVM with non-linear kernel utilizing the network trained in first step and optimized features of Step 2.

In Step 1, we define the network and initialize its weights using Xavier initialization (Glorot and Bengio, 2010). We then train a CNN using a stochastic gradient descent back-propagation algorithm. Parallely, in Step 2, MOO based feature selection technique is employed to identify the most relevant set of features within the framework of SVM. Once the training of CNN is over, i.e. optimal parameters of the network are found, in Step 3 we concatenate the output of top hidden layer and optimized feature set reported by MOGA and feed it to SVM.

CNN performs reasonably well in capturing the relevant lexical and syntactic features on its own. Thus, the first step of the proposed approach ensures that it extracts such features from the training data automatically. The SVM in the proposed approach makes use of the features extracted from CNN along with the optimized features (from MOGA) to define a hyperplane which is more robust as compared to what defined by either CNN or SVM with optimized features alone. The pseudo code of the proposed approach is sketched in Algorithm 1. Statements 1-6 deal with the first step i.e. training of the deep learning network to learn sentiment embedded vectors while statement 7 finds out the optimized feature set. The last step is carried out by statements 8-14.

2.1 Convolutional neural network (CNN)

CNN is a special kind of multi-layer neural network which consists of one or more convolutional and pooling layers, followed by one or more fully-connected layers. The convolutional and pooling layers implicitly extract relevant feature representation from input data, and fed it to the fully connected layers for classification. The size and weights of the convolution filters determine the features to be extracted from the input data. Same convolution filter is floated over the complete input data in order to extract similar features at different spatial locations. Max pool layer is then applied to select the most significant features from the CNN features. Subsequently, after iterating several convolutional and max pooling layers, it is fed to a fully connected layer for classification. In general we use softmax as an activation

Algorithm 1 (*Pred*, *Acc*) = *CNN-SVM*$_{(W+X)}$ *(Train, Dev, Test, Test-Gold, θ)*

```
Require: Train, Dev, Test, Test-Gold - Datasets; θ - Termination criteria.
Ensure: Pred - Predicted output; Acc - Accuracy achieved.
 1: $Net \leftarrow$ BuildNetwork()
 2: InitializeNetwork($Net$)
 3: for $error >= \theta$ do
 4:     $error \leftarrow$ TrainNetwrok($Net, Train, Dev$)
 5: end for
 6: /* Training complete */
 7: $Feature_{opt} \leftarrow$ MOGA($Train, Dev$)
 8: $H_{Train} \leftarrow$ GetTopHiddenLayer($Net, Train$)
 9: $Train_{combined} \leftarrow H_{Train} + Feature_{opt}$
10: $Model_{SVM} \leftarrow$ SVM$_{\text{Train}}$($Train_{combined}$)
11: $H_{Test} \leftarrow$ GetTopHiddenLayer($Net, Test$)
12: $Test_{combined} \leftarrow H_{Test} + Feature_{opt}$
13: $Pred \leftarrow$ SVM$_{\text{Test}}$($Model_{SVM}, Test_{combined}$)
14: $Acc \leftarrow$ Evaluation(Test-Gold, Pred)
15: return (Pred, Acc)
```

function in the fully connected layer. A typical CNN architecture is shown in Figure 1b. Feature map represents the size of the filter while each edge corresponds to a weight of the filter.

2.2 Word representation

A neural network requires word embedding (or, sentence embedding) as an input to the network, i.e. a vector representation of each word or sentence. We use word2vec tool (Mikolov et al., 2013) which efficiently captures the semantic properties of words in the corpus. We train with a corpus of 6.7 million sentences, which were collected from Wikipedia and Twitter sources. This trained model is used for translating a word into its respective vector representation. We set the vector dimension of a word to 200. Each sentence is padded with zero vectors in order to make its length uniform throughout the dataset. Hence, the vector dimension ($Vector_{dim}$) of each sentence (i.e. number of neurons at input layer) counts to 200×max-sentence-length.

2.3 Multi-Objective genetic algorithm (MOGA) based feature selection

We develop a feature selection technique based on multi-objective optimization (MOO) (Deb, 2001). The problem of feature selection can be modeled as follows: Given a set of features F and $M = \langle m_1, m_2, .., m_M \rangle$ objective functions, find a subset F^* of F such that M objectives are optimized simultaneously. For instance, maximization of all objective functions can be mathematically stated as:

$$Objective_M(F^*) = \underset{M, S\epsilon F}{argmax}\{Objective_M(S)\}$$

We use a binary version of genetic algorithm (GA) for determining the best fitting feature set. The basic operations of GA are 'crossover', 'mutation' and 'selection'. First, we randomly initialize N chromosomes of length n, each representing a solution in the population. The length of each chromosome (n) corresponds to the number of features available, i.e. each bit position encodes exactly one feature. The value of 1 in a bit position denotes that the respective feature is used for classifier's training, otherwise the feature is not used. A representation of a chromosome is presented in Figure 2. Selection, crossover

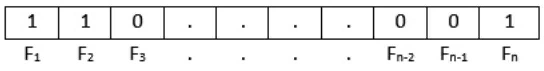


Figure 2: Representation of chromosome in GA based optimization.

and mutation operations are then performed on the chromosomes.

1. **Selection:** At first we select top N solutions w.r.t fitness value. For fitness computation, we construct a SVM based classifier with the selected features, and iterate this process 5 times for 5-fold

cross-validation experiments. In multi-objective optimization we perform non-dominating sorting for the selection. Two solutions, A & B, are non-dominated to each other if solution A is not bad than solution B in at least one objective function and vice-verse. In contrast, a solution A dominated by B if for all objective functions A is less optimal than solution B. A set of solutions, that are non-dominating to each other but dominates every other solutions in the population forms a non-dominating *front-0*. Similarly, non-dominating *front-1* consists of remaining solutions that are non-dominating to each other but dominate other solutions. Hence, *front-0* solutions dominates *front-1* solutions which in turns dominates *front-2* solutions and so on. Set of solution in *front-0* forms pareto-optimal surface (Rank 1). A pictorial representation of non-dominating solutions are depicted in Fig. 3. We use binary tournament selection, as in non-dominated sorting GA (NSGA)-II (Deb et al., 2000). We use elitism operation, where non-dominated solutions among parent and child generations are propagated to the next generation. MOO provides a set of non-dominated solutions (Deb et al., 2002) on the final Pareto optimal front. Although each of these solutions is equally important from the algorithmic point of view, but user may often require to produce only a single solution. In our case we select the particular solution that yields maximum accuracy.

2. **Crossover:** In crossover, for any two solutions a random split position is chosen. Two new solutions are generated by swapping the information of the chromosomes with each other at the split point.

3. **Mutation:** Similarly, mutation operator is applied to each entry of chromosome, where an entry is randomly replaced by 0 or 1 based on mutation probability.

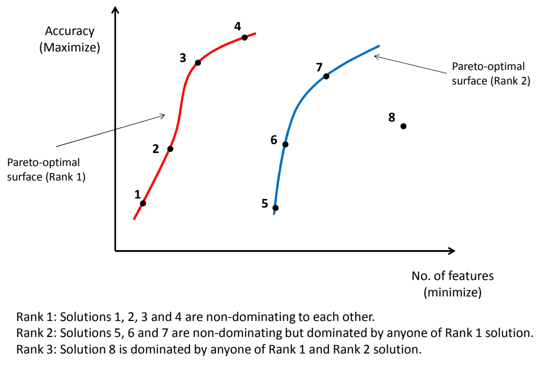

Accuracy (Maximize)
Pareto-optimal surface (Rank 2)
Pareto-optimal surface (Rank 1)
1
2
3
4
5
6
7
8
No. of features (minimize)
Rank 1: Solutions 1, 2, 3 and 4 are non-dominating to each other.
Rank 2: Solutions 5, 6 and 7 are non-dominating but dominated by anyone of Rank 1 solution.
Rank 3: Solution 8 is dominated by anyone of Rank 1 and Rank 2 solution.

Figure 3: Representation of dominated and non- dominated solutions.

In this work we optimize two objective functions: accuracy (maximize) and number of features (minimize). We set the parameters of MOO as follows: population size=60, number of generations=30, crossover probability=0.8, mutation probability=0.03.

3 Datasets, Experiments and Analysis

3.1 Datasets

For experiments we use the following four datasets for Hindi:

1. **Twitter-Hindi ($Twitter_H$):** We use benchmark dataset released by the organizers of 'SAIL: Sentiment Analysis in Indian Languages' task (Patra et al., 2015).

2. **Online reviews for aspect based sentiment analysis in Hindi ($Review_{A_H}$)[3]:** This dataset is developed by us (Akhtar et al., 2016) for aspect based sentiment analysis (ABSA). It comprises of 5,417 product and service reviews across 12 domains. Reviews are annotated with aspect terms along with

its polarities. We consider four classes, namely *positive*, *negative*, *neutral* and *conflict*. In this work we solve only one problem of ABSA i.e. aspect term sentiment problem.

3. **Online reviews for sentence based sentiment analysis in Hindi ($Reviews_{S_H}$)[3]:** There is no available benchmark dataset which deals with sentence-level sentiment analysis for online product reviews in Hindi. Therefore, we extract user reviews from $Review_{A_H}$ dataset and annotate these using four classes as mentioned above.

4. **Online movie reviews-Hindi ($Movie_H$)[3]:** We collect user reviews from various news and blog websites, and annotate using four classes.

Detailed statistics of the above datasets are presented in Table 1. For each dataset $Review_{A_H}$, $Reviews_{S_H}$ and $Movie_H$ we distribute 70%, 20% and 10% of the data as training, test and development, respectively. For generalization, we also evaluate the proposed method on two other benchmark datasets in English *viz.* SemEval 2015 shared task on sentiment analysis in twitter (Rosenthal et al., 2015) and SemEval-2014 shared task on ABSA (Pontiki et al., 2014).

Datasets		**Sentiment**				
		#Pos	**#Neg**	**#Neu**	**#Con**	**Total**
$Twitter_H$	Train	168	559	494	-	1221
	Test	166	251	50	-	467
$Review_{A_H}$	-	1986	569	1914	40	4509
$Reviews_{S_H}$	-	2290	712	2226	189	5417
$Movie_H$	-	823	530	598	201	2152

Table 1: Dataset statistics. Here, pos: positive, neg:negative, neu:neutral and con:conflict

3.2 Baseline, proposed model and its variants

In order to compare our proposed approach, we define the following baseline models:

- B_{SVM}: This is a SVM based model that incorporates all the available features.

- B_{CNN_W}: It is a simple CNN based model, trained and evaluated using word embedding as features (c.f. Section 2.2).

In addition, we also try to understand the behavior of the proposed model in presence or absence of extra handcrafted features. For a comparative study we define following two models based on CNN architecture:

- CNN-SVM$_W$: This represents our proposed model in the absence of optimized feature set. It is trained and evaluated only with the word embedding features. We extract feature vectors from the top hidden layer and feed it to SVM for training.

- $B_{CNN_{(W+X)}}$: This model is similar to baseline B_{CNN_W}. The only difference is the usage of optimized features as determined by MOO based feature selection technique (in addition to word embedding).

3.3 Feature set

Table 2 shows the set of features that we use for building different models and the optimized feature subset that we obtain through the feature selection technique.

[3]Resource available at http://iitp.ac.in/~ai-nlp-ml/resources.html

Category	Dataset	Feature	Description
Lexical and syntactic features	All	PoS, Word N-grams, Character N-grams	Part-of-Speech tag, Unigram, Bigram and Trigram
Twitter specific features	$Twitter_H, Twitter_E$	Hashtags	Number of hashtag(#) tokens in the tweet.
		Emoticons	Binary valued feature denotes the presence or absence of the positive and negative emoticons
		Punctuation	Number of occurrences of contiguous sequence of question marks, exclamation marks etc
		URL and Username	# of url and usernames present in the tweet.
		Average length	Average length of the tokens
Lexicon features	$Review_{A_H}$, $Reviews_{S_H}$, $Movie_H$	SentiWordNet for Indian Language (Das and Bandyopadhyay, 2010)	# of positive tokens, negative tokens and average score.
	$Review_{A_H}$, $Reviews_{S_H}$, $Movie_H$, $Review_{A_E}$	Semantic Orientation (Hatzivassiloglou and McKeown, 1997)	Sum of semantic orientation score of each token.
	$Twitter_E$, $Review_{A_E}$	Bing Liu lexicon (Ding et al., 2008)	# of positive tokens and negative tokens.
		MPQA lexicon(Wiebe and Mihalcea, 2006)	Number of positive tokens and negative tokens.
	$Twitter_E$	NRC lexicons (Mohammad et al., 2013b; Mohammad and Turney, 2013)	# of tokens with positive score, negative score and zero score, total emoticons score and total sentiment score.

Dataset	Optimized feature set*
$Twitter_H$	Emoticons, Punctuation, SentiWordNet
$Review_{A_H}$, $Reviews_{S_H}$	Semantic Orientation
$Movie_H$	Semantic Orientation, SentiWordNet
$Twitter_E$	HashTag, Emoticons, Punctuation, Bing Lui and NRC Lexicon
$Review_{A_E}$	Bing Lui and MPQA Lexicon

*We leave out lexical and syntactic features from the optimized set as these information will be captured by the CNN itself.

Table 2: Feature set and the optimized features

3.4 Experiments

For experiments we use DL4J[4], a java based package for deep learning implementation, and LibSVM library (Chang and Lin, 2011) for SVM. We use the development set to fine-tune the parameters of CNN. For SVM, we perform grid search to find the optimal parameter settings of RBF kernel. CNN classifier is trained for 50, 100 and 120 epochs with 300 feature maps of size $4 \times Vector_{dim}$. We use stochastic gradient descent and negative log-likelihood as the optimization algorithm and loss function, respectively. In addition, we use L2 regularization and dropout technique (Srivastava et al., 2014) to build a robust system. Results of the proposed method along with the baselines are presented in Table 3a. Our proposed method achieves 62.52% accuracy for $Twitter_H$ which convincingly outperforms SVM based baseline by 13 points, and reports approximately 2 points better accuracy as compared to B_{CNN_W}. Comparison to the participating systems of SAIL shared task shows that we are ahead of the best system as reported in (Se et al., 2015) by almost 7 points. For aspect-level Hindi review dataset, $Review_{A_H}$ the proposed approach reports an accuracy of 65.96% against 54.09% as reported in our previous attempt (Akhtar et al., 2016), which was based only on SVM. Similarly for sentence-level sentiment analysis on $Reviews_{S_H}$ and $Movie_H$ datasets, our proposed method performs better compared to the other baselines. Since evaluation on $Reviews_{S_H}$ and $Movie_H$ datasets are performed for the first time, we do not have any existing model for comparison. In Table 3b, we show the class-wise accuracies of *CNN-SVM*$_{(W+X)}$ for the Hindi datasets.

3.4.1 Effect of handcrafted features

Since the twitter-specific features are not very relevant for the product reviews dataset, we only use lexicon-based features for it. In comparison to the baseline B_{CNN_W}, augmenting external features in $B_{CNN_{(W+X)}}$ shows better accuracy. We also observe similar phenomenon for all the other settings. Addition of extra feature helps *CNN-SVM*$_{(W+X)}$ for $Twitter_H$ to achieve accuracy of 62.45% as compared

[4]http://deeplearning4j.org/

to 61.24% without it. Similarly, for $Review_{A_H}$ dataset *CNN-SVM*$_{(W+X)}$'s improvement is close to 5% by just using the sentiment lexicon features. Since word embeddings are good at capturing the semantic information, addition of lexicon based features assist it in finding the sentiment more accurately. It should be noted that augmentation of external features along with the features automatically extracted from CNN at the penultimate layer (sentiment augmented optimized vector) yields better result compared to the model where external features added to the word embeddings at the very input layer. This can be attributed to the fact that information in external features are lost through a series of convolution and max-pooling layers. While we add all features to B_{SVM} in the network, we observe that performance drops. This could be because the network itself captures lexical features on its own, and augmenting features further leads to over-fitting.

Method	**Accuracy**			
	$Twitter_H$	$Review_{A_H}$	$Reviews_{S_H}$	$Movie_H$
B_{SVM}	49.02	54.07	51.52	38.76
B_{CNN_W}	60.60	59.13	55.12	40.31
CNN-SVM$_W$	61.24	59.26	56.47	41.70
$B_{CNN_{(W+X)}}$	61.89	59.53	55.56	41.40
(Se et al., 2015)	55.60	-	-	-
Previous system: (Kumar et al., 2015), (Akhtar et al., 2016)	46.25	54.09	-	-
CNN-SVM$_{(W+X)}$	**62.52**	**65.96**	**57.34**	**44.88**

(a) Overall performance

Class	**Accuracy**			
	$Twitter_H$	$Review_{A_H}$	$Reviews_{S_H}$	$Movie_H$
Positive	24.69 (41/166)	67.43 (265/393)	65.77 (294/447)	87.71(150/170)
Negative	88.84 (223/251)	58.94 (89/151)	27.64 (47.170)	23.58 (25/106)
Neutral	56.0 (28/50)	70.46 (241/342)	65.71 (276/420)	21.68 (18/83)
Conflict	-	00.00 (0/16)	8.6 (4/46)	00.00 (0/70)
Total	**62.52 (292/467)**	**65.96 (595/902)**	**57.34 (621/1083)**	**44.88 (913/430)**

(b) Class-wise performance

Table 3: Results of baseline models and proposed method for $Twitter_H$, $Review_{A_H}$, $Reviews_{S_H}$ and $Movie_H$ datasets. Subscript *W+X* represents models with word embedding and optimized feature set while *W* represent models with only word embeddings. B_{SVM} and B_{CNN_W} are the two baseline systems and CNN-SVM$_{(W+X)}$ is the proposed method. CNN-SVM$_W$ represents proposed system without optimized feature set.

3.4.2 Evaluation on other benchmark datasets

In order to show the domain and language adaptability, we evaluate our proposed method i.e. *CNN-SVM*$_{(W+X)}$, on two benchmark datasets in English *viz.* $Twitter_E$ and $Review_{A_E}$. The $Twitter_E$ dataset belongs to SemEval-2015 shared task on sentiment analysis in Twitter (Rosenthal et al., 2015) and comprises of 8,210, 1,654 and 2,392 tweets for training, testing & development, respectively. Tweets in the dataset belong to different genres, i.e. **generic** as well as **sarcastic**. We evaluate the system for both genres in isolation. The second dataset belongs to SemEval-2014 shared task on ABSA (Pontiki et al., 2014). It contains approximately 3,800 user reviews from two domains, *viz.* laptop and restaurant. Table 4 depicts the results on $Twitter_E$ and $Review_{A_E}$ datasets for both the genres and domains, respectively. Results suggest that use of SVM on top of CNN, i.e. *CNN-SVM*$_W$ performs better than the typical CNN system, i.e. B_{CNN_W}. We observe the same phenomenon when optimized feature sets are concatenated with word embeddings in systems $B_{CNN_{(W+X)}}$ and *CNN-SVM*$_{(W+X)}$.

Method	Accuracy			
	$Twitter_E$		$Review_{A_E}$	
	Tweets	Sarcasm	Laptop	Restaurant
B_{SVM}	56.31	58.33	57.18	69.92
B_{CNN_W}	51.61	45.0	64.98	73.46
CNN-SVM$_W$	52.96	50.0	65.29	74.65
$B_{CNN_{(W+X)}}$	56.06	51.67	67.28	74.07
CNN-SVM$_{(W+X)}$	**58.62**	**61.67**	**68.04**	**77.16**

Table 4: Results of baseline models and proposed method for $Twitter_E$ and $Review_{A_E}$ datasets

#	Domain		Sentence	Actual	Predicted
1	$Reviews_{S_H}$	D	स्पीकर के आवाज़ की क्वालिटी अच्छी है , लेकिन हम कुछ और तेज़ आवाज़ चाहते थे ।	**Conflict**	**Positive**
		T_l	speekara ke AAvaaZ kee kvaaliTee Achchhee hai , lekin ham kuchh AOra teZ AAvaaZ chaahate the .		
		T_r	The sound quality of the speaker is good, but we expected better sound.		
	$Movie_H$	D	इन फिल्मों में कोई मजबूत कहानी नहीं थी मगर स्टंट के भरोसे फिल्म चल रही थी ।	**Conflict**	**Negative**
		T_l	In philmoN meN koEE majaboot kahaanee naheeN thee magara sTaNT ke bharose philm chal rahee thee .		
		T_r	There was no strong story in the film but has good stunts.		
2	$Twitter_H$	D	गणपति विसर्जन के मौके पर सुरक्षा कड़ी - URL	**Positive**	**Neutral**
		T_l	gaNNapati visarjan ke maoke para surakShaa kaDdee - URL		
		T_r	Tight security on the eve of Ganpati Visargan - URL.		
	$Reviews_{S_H}$	D	3जी प्लास्टिक का बना हुआ था जो हाथ से फिसलता था ।	**Negative**	**Neutral**
		T_l	3jee plaasTik kaa banaa huAA thaa jo haath se phisalataa thaa .		
		T_r	It was made up of 3G plastic which was slippery.		
3	$Reviews_{S_H}$	D	लेकिन इसमें लगे एएमडी रेडिओन एचडी7670एम ग्राफ़िक्स चिप का प्रदर्शन लाज़वाब है ।	**Positive**	**Neutral**
		T_l	lekin IsameN lage EEmaDee reDiOn EchaDee7670Em graaphiksa chip kaa pradarshan laaZvaab hai .		
		T_r	But the performance of the integrated AMD Radion HD7670M graphics chip is splendid.		

Table 5: Qualitative analysis: Examples of the error case. D, T_l and T_r represents devanagari, transliterated and translated forms

3.5 Analysis of results

Results suggest that our proposed architecture performs reasonably well for different domains and languages. The traditional CNN architectures are known to be capturing the lexical and structural features very well. We incorporate this idea into our work to learn sentiment embedded vector which helps in the attaining better results as evident from Table 3. The usage of SVM (rather than traditional softmax function) on sentiment embedded vector is able to generate the decision hyperplane more accurately by projecting the CNN features into higher dimension. Further, the (near) optimal set of features produced by MOO based feature selection technique (in addition to CNN features) assists SVM for more accurate prediction using sentiment augmented optimized vector.

We also perform Analysis of Variance (ANNOVA) (Anderson and Scolve, 1978) test which is a measure of statistical significance on the obtained results. We execute our approach 10 times with varying parameter settings. We observed that the variance in mean accuracy between proposed method and state-of-the-art methods is less than 5%. It signifies that improvements over the baselines are statistically significant.

We perform a detailed analysis (quantitative and qualitative) of the outputs to study the effectiveness as well as the shortcomings of the proposed approach.

We observe that 13% and 5.8% mis-classified test instances are correctly predicted by the *CNN-SVM*$_{(W+X)}$ model for Twitter and reviews domain, respectively. Motivations for using CNN architecture are two-fold: i) to learn hidden semantics from a large unlabeled corpus; and ii) handling limited coverage of lexical resources (e.g. Hindi SentiWordNet). In contrast to B_{SVM}, we observe that CNN architecture correctly captures instances such as "@imVkohli: धोनी के 'अतिआत्मविश्वास' के

कारण हारी टीम (@imVkohli: dhonee ke 'AtiAAtmavishvaasa' ke kaaraNN haaree Teema-@imVkohli: Team lost due to over-confidence of Dhoni.)", in which sentiment bearing words: अतिआत्मविश्वास (AtiAAtmavishvaas:Over-confidence) and हारी (haare:lost) were not found in SentiWordNet and in training set as well.

While analysing the outputs for the errors, we observe the following points:

1. Sentiment in a sentence can be either expressed by a explicit use of sentiment word e.g. अच्छा (Achchhaa:good) or by a word which carries implicit sentiment e.g. फीके (pheeke:light). For conflict sentences, explicit use of a positive or negative sentiment word drives the system to predict its output as 'positive' or 'negative'. In the first sentence of Table 5, presence of अच्छी (Achchhee:good) misguides the system to predict its sentiment as 'positive'.

2. Absence of an explicit sentiment marker in a sentence makes it harder for the system to correctly predict the sentiment. For the second sentence in Table 5, the system classifies as 'neutral' as no explicit trigger word is present.

3. The system mis-classifies some of the sentences which have explicit sentiment bearing words, but their corresponding word representations are missing due to their rare occurrences. For example, in the third sentence, word लाज़वाब (laaZvaab:splendid) has a positive sentiment. As its representation is missing from the word embedding output, system incorrectly predicts it as 'neutral'.

4 Conclusion

In this paper, we propose an efficient hybrid deep learning architecture for sentiment analysis in resource-poor languages. We learn sentiment embedded vector using CNN architecture and make a final prediction by replacing softmax function with a stronger classifier, i.e. SVM at the output layer of CNN. Training of SVM is further assisted by the optimized feature set computed by a multi-objective GA based feature selection technique to form sentiment augmented optimized vector. We build various models and evaluate our proposed method on the datasets of varying domains: Twitter (generic & sarcastic) and product/service reviews (aspect-level and sentence-level sentiment analysis). For all datasets we observed that our method consistently reports better accuracy than the various baselines and state-of-the-art systems. We observed that the usage of SVM and optimized feature set in the proposed approach helps it to achieve encouraging performance across the domains and languages compared to the state-of-the-art methods.

In this work, we include only one of the sub-problems of aspect based sentiment analysis i.e. aspect term sentiment classification. In future we would like to solve other sub-problems of ABSA such as aspect term extraction, aspect category detection and its sentiment classification. Aspect term extraction is an sequence labeling task while aspect category detection is a multi-lable classification tasks. We plan to explore recurrent neural networks (RNN) for aspect term extraction and extend our CNN based approach for multi-label classification. Also, since quality of word representation is an important factor in any neural network architecture, we plan to make use of techniques such as distance supervision for enhancing the quality of word representations.

References

Md Shad Akhtar, Asif Ekbal, and Pushpak Bhattacharyya. 2016. Aspect based sentiment analysis in hindi: Resource creation and evaluation. In *In Proceedings of the 10th edition of the Language Resources and Evaluation Conference (LREC)*.

T. W. Anderson and S.L. Scolve. 1978. *Introduction to the Statistical Analysis of Data*. Houghton Mifflin.

Akshat Bakliwal, Piyush Arora, and Vasudeva Varma. 2012. Hindi Subjective Lexicon: A Lexical Resource For Hindi Polarity Classification. *In Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC)*.

A. R. Balamurali, Aditya Joshi, and Pushpak Bhattacharyya. 2012. Cross-Lingual Sentiment Analysis for Indian Languages using Linked WordNets. In *COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Posters, 8-15 December 2012, Mumbai, India*, pages 73–82.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27.

Amitava Das and Sivaji Bandyopadhyay. 2010. SentiWordNet for Indian Languages. *In Asian Federation for Natural Language Processing, China*, pages 56–63.

Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and Tanaka Meyarivan. 2000. A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. In *Parallel problem solving from nature PPSN VI*, pages 849–858. Springer.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. 2002. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197.

Kalyanmoy Deb. 2001. *Multi-objective Optimization Using Evolutionary Algorithms*, volume 16. John Wiley & Sons.

Xiaowen Ding, Bing Liu, and Philip S. Yu. 2008. A Holistic Lexicon-based Approach to Opinion Mining. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM ’08.

Cícero Nogueira dos Santos and Maira Gatti. 2014. Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In *COLING*, pages 69–78.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS’10). Society for Artificial Intelligence and Statistics*.

Deepak Kumar Gupta, Kandula Srikanth Reddy, Asif Ekbal, et al. 2015. PSO-ASent: Feature Selection Using Particle Swarm Optimization for Aspect Based Sentiment Analysis. In *Natural Language Processing and Information Systems*, pages 220–233. Springer.

Vasileios Hatzivassiloglou and Kathleen R McKeown. 1997. Predicting the Semantic Orientation of Adjectives. In *Proceedings of the ACL/EACL*, pages 174–181.

Soo-Min Kim and Eduard Hovy. 2004. Determining The Sentiment of Opinions. In *In proceedings of the 20th International Conference on Computational Linguistics*, page 1367. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *arXiv preprint arXiv:1408.5882*.

Ayush Kumar, Sarah Kohail, Asif Ekbal, and Chris Biemann. 2015. IIT-TUDA: System for Sentiment Analysis in Indian Languages Using Lexical Acquisition. In *Mining Intelligence and Knowledge Exploration*, pages 684–693. Springer.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*.

Saif M. Mohammad and Peter D. Turney. 2013. Crowdsourcing a Word-Emotion Association Lexicon. 29(3):436–465.

Saif M Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013a. NRC-Canada: Building The State-of-The-Art in Sentiment Analysis of Tweets. *arXiv preprint arXiv:1308.6242*.

Saif M. Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013b. NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets. In *In Proceedings of the seventh international workshop on Semantic Evaluation Exercises (SemEval-2013)*, Atlanta, Georgia, USA, June.

Bo Pang and Lillian Lee. 2008. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.

Braja Gopal Patra, Dipankar Das, Amitava Das, and Rajendra Prasath. 2015. Shared Task on Sentiment Analysis in Indian Languages (SAIL) Tweets-An Overview. In *Mining Intelligence and Knowledge Exploration*, pages 650–655. Springer.

NLMM Pochet and JAK Suykens. 2006. Support Vector Machines Versus Logistic Regression: Improving Prospective Performance in Clinical Decision-making. *Ultrasound in Obstetrics & Gynecology*, 27(6):607–608.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 Task 4: Aspect Based Sentiment Analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, Dublin, Ireland, August.

Soujanya Poria, Erik Cambria, and Alexander Gelbukh. 2015. Deep Convolutional Neural Network Textual Features and Multiple Kernel Learning for Utterance-level Multimodal Sentiment Analysis. In *In Proceedings of EMNLP*, pages 2539–2544.

Sara Rosenthal, Preslav Nakov, Svetlana Kiritchenko, Saif M Mohammad, Alan Ritter, and Veselin Stoyanov. 2015. Semeval-2015 Task 10: Sentiment Analysis in Twitter. *In Proceedings of SemEval-2015.*

Shriya Se, R Vinayakumar, M Anand Kumar, and KP Soman. 2015. AMRITA-CEN@ SAIL2015: Sentiment Analysis in Indian Languages. In *Mining Intelligence and Knowledge Exploration*, pages 703–710. Springer.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Yichuan Tang. 2013. Deep Learning Using Linear Support Vector Machines. *arXiv preprint arXiv:1306.0239*.

V. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Synthesis Lectures on Human Language Technologies. Springer.

Janyce Wiebe and Rada Mihalcea. 2006. Word Sense and Subjectivity. In *In proceedings of the COLING/ACL*, pages 1065–1072.