# Self-Attention Enhanced Recurrent Neural Networks for Sentence Classification

Ankit Kumar
*South Asian University*
New Delhi, India
yaduvanshiankit@outlook.com

Reshma Rastogi (nee Khemchandani)
*Department of Computer Science*
*South Asian University*
New Delhi, India
reshma.khemchandani@sau.ac.in

***Abstract*—In this paper we propose self-attention enhanced Recurrent Neural Networks for the task of sentence classification. The proposed framework is based on Vanilla Recurrent Neural Network and Bi-directional Recurrent Neural Network architecture. These architectures have been implemented over two different recurrent cells namely Long Short-Term Memory and Gated Recurrent Unit. We have used the multi-head self-attention mechanism to improve the feature selection and thus preserve dependency over longer lengths in the recurrent neural network architectures. Further, to ensure better context development, we have used Mikolov's pre-trained word2vec word vectors in both the static and non-static mode. To check the efficacy of our proposed framework, we have made a comparison of our models with the state-of-the-art methods of Yoon Kim on seven benchmark datasets. The proposed framework achieves a state-of-the-art result on four of the seven datasets and a performance gain over the baseline model on five of the seven datasets. Furthermore, to check the effectivity of self-attention on the task of sentence classification, we compare our self-attention based framework with Bahdanau's attention based implementation from our previous work.**

***Index Terms*—Sentence Classification, Recurrent Neural Network, Bidirectional Recurrent Neural Network, Long Short-Term Memory, Gated Recurrent Unit, Self-Attention**

## I. Introduction

Sentence classification is a broad term which includes several important text classification tasks like language detection [1], intent detection [2], emotion detection [3], sentiment analysis [4] among others. Over the past few decades, there has been enormous advancement in the accessibility of text data which has encouraged and additionally entangled the task of sentence classification, bringing about the necessity of more organized and adaptable approaches to manage this issue. This, in turn, has resulted in the growth of application of sentence classification tasks in various field such as social media trends [5], financial market [6], medical applications [7], entertainment, fashion and news industry [8] among others.

A large chunk of data that we experience is textual. Text data requires taking into account both the semantic and syntactic information for classification. With the approach of deep learning, Natural Language Processing (NLP) has accomplished new dimensions. It empowers our machines to analyze, comprehend and determine meaning out of texts. Recurrent Neural Network (RNN), a deep neural network architecture, is showing up as a promising contrasting option to withstand this challenge.

A recurrent neural network is a class of artificial neural network that has been utilized for a number of applications like intent detection [2], speech recognition [9], image captioning [10], language translation [11], and numerous others. Theoretically, RNNs showcase dynamic temporal behavior for a time series. However, as explained by Hochreiter [12] and Bengio, et al., [13] vanilla recurrent neural networks are susceptible to vanishing or exploding gradients. To overcome this limitation, Hoschreiter et al., [14] in 1997 introduced Long Short-Term Memory (LSTM). LSTM uses three gate mechanism to solve the gradient problem. A more generalized version of LSTM namely, Gated Recurrent Unit (GRU), was proposed by Cho et al., [15] in 2014. These have been utilized to improve the state-of-the-art for various complex problems including language translation, speech modeling, handwriting recognition, among others. Chung et al., [16] has shown that both LSTMs and GRUs deliver comparable outcomes on most tasks with the principle distinction being that LSTMs are computationally less efficient than GRUs since LSTM uses three gates, namely forget, input and output while GRUs use only two gates, reset and update.

Bi-directional Recurrent Neural Network (BRNN) proposed by Schuster et al., [17] in 1997, is an architectural variant of RNN that improves the context development in the network. It does so by including two independent RNNs aligned in opposite order. Thus, at each time-step, BRNN has the context from both the past as well as the future, resulting in richer context architecturally.

Text classification, being the contextual task, requires the architecture to develop as much strong context as possible. But, we shouldn't rely only on the architecture to frame the context for classification. Therefore, we use word embedding, which is known to preserve the syntactic and semantic information of the text, as input to RNNs instead of the cliche one-hot vector which suffers from the curse of high dimensionality with the increasing size of the input vocabulary. In 2016, Mikolov et al., [18] trained a word2vec model on Google News data consisting of 100 billion words and made it publically available. We intend to use Mikolov's word2vec model to generate word vectors of the desired size.

The rest of this paper is structured as follows: Section 2 highlights some of the work being carried out in the field of text classification using RNN and the calculation of attention

978-1-5386-9276-9/18/$31.00 

mechanism and self-attention mechanism. Section 3 details our model. while section 4 incorporates the details of our implementations, datasets, results and the observations made based on the basis of the outcomes of our experiments. We conclude this paper with our final remarks in section 5.

## II. Background

### A. Recurrent Neural Networks for Text Classification

Recurrent neural network is a sequential network in which output at each step is the function of its current input and the outputs of the previous inputs. With the recent advancement in the field of text classification using RNN, recurrent networks are now being used for a variety of tasks in text classification. Pollastri et al., [19] used RNNs for predicting protein secondary structure in 2002. Arevian [20] used RNN to classify real-life text data in 2007. Irsoy et al., [21] in 2014, used RNN for opinion mining. Tang et. al., [22] in 2015, used the gated recurrent network for sentiment classification. Lai et al., [23] used RNN in combination with Convolution Neural Network (CNN) to classify text in 2015. Melsin et al., [24] used RNNs for the task of slot filling in 2015. Lee et al., [25] also used the recurrent neural network in combination with Convolution Neural Network to classify short texts in 2016. Liu et al., [2] in 2016 used RNN for the task of joint intent detection and slot filling. RNNs are being used for diverse tasks in text classification. It seems only sane to explore more with RNN for the task of sequential classification.

### B. Attention

Recurrent neural networks, even though being a sequential network, do not perform well on data where the length of the sequence is large. In practice, vanilla RNNs work well with a sequence length of 3-5 only. This is mainly due o the fact that RNNs suffer from vanishing or exploding gradient problem. LSTM and GRU improve the RNN architecture to an extent where the architecture is able to learn sequences of length 10-15. But, most of the sentences we came across in our datasets have a length greater than 18. Thus, we needed a stronger approach which could make RNNs deal with sequences of longer lengths. Bahdanau et al., [26] in 2014, proposed attention mechanism in the landmark language translation paper. Attention mechanism is a strong method that allows the network to learn sequences of longer lengths. It forms an additional layer and learns from the sums of previous outputs to decide which features are important to retain by the network. Thus, the attention mechanism ensures that RNNs can learn over the sequences at longer lengths.

To calculate the yield for the current cell, attention mechanism utilizes the weighted blend of all the input states, not only the last state. Attention weights enable the network to choose the amount of each past input ought to be considered, keeping in mind the end goal, to get yield for the next cell. In this way, it helps in choosing which state's yield is more imperative for the output of current cell. Numerically, for a hidden unit $h_t$, at each time step, Bahdanau's attention model

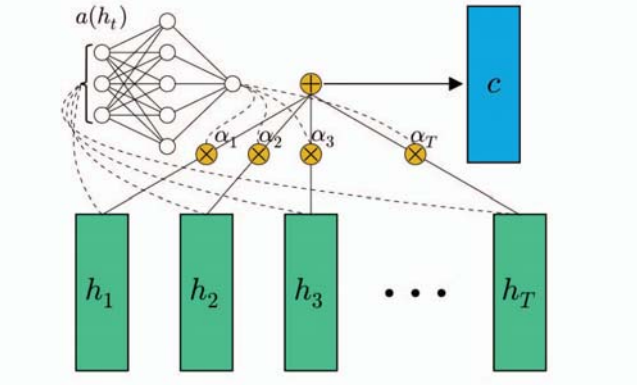


Fig. 1. An illustration of Attention Mechanism.

calculates a context vector $c_t$ as the weighted mean of the state sequence as follow

$$c_t = \sum_{j=1}^{T} \alpha_{tj} h_j, \tag{1}$$

where $T$ is the number of time steps, $\alpha_{tj}$ is the calculated weight at time step $t$ for $h_j$. This context vector is then used to process a new state sequence $s$ where $s_t$ relies upon past state sequence $s_{t-1}$, current context $c_t$ and model's yield at time $t-1$. The weight $\alpha_{tj}$ is then computed as

$$e_{tj} = a\left(s_{t-1}, h_j\right), \tag{2}$$

$$\alpha_{tj} = \frac{\exp\left(score\left(e_{tj}, h_s\right)\right)}{\sum_{k=1}^{T} \exp\left(e_{tk}, h_s\right)}. \tag{3}$$

where $a$ is a learned function that computes a scalar significance value for $h_j$ given the estimate of $h_j$ and the previous state $s_{t-1}$.

### C. Self-Attention

Bahdanau's attention model is useful when the network has some extra information. The task of text classification is given a sentence only. Hence, Bahdanau's attention model seems to not work to its full capacity for the task of sentence classification. Thus, Lin et al., [27] in 2017 proposed Self-Attention mechanism for the tasks like text classification. Self-attention sublayers utilize $h$ attention heads. To frame the sublayer yield, parameterized linear transformation is applied to the concatenation of results from each head.

Each attention head computes a new sequence $z = (z_1, z_2, ..., z_n)$ by operating on the input sequence $x = (x_1, x_2, ..., x_n)$.

Each $z_i$, the output, is calculated as the weighted sum of linearly transformed inputs:

$$z_i = \sum_{k=1}^{n} \alpha_{ik} \left(x_k W^V\right), \tag{4}$$

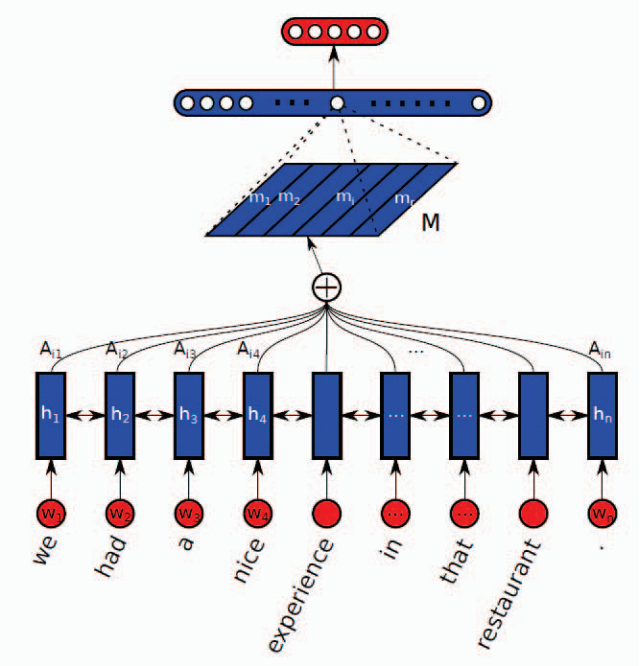


Fig. 2. An illustration of Self-Attention Mechanism.

Each weight coefcient, $\alpha_{ik}$, is computed using a softmax function:

$$\alpha_{ik} = \frac{\exp e_{ik}}{\sum_{j=1}^{n} \exp e_{ij}}. \tag{5}$$

Further, $e_{ik}$ is calculated using a function that compares two inputs:

$$e_{ik} = \frac{\left(x_i W^Q\right)\left(x_k W^J\right)^T}{\sqrt{d_z}} \tag{6}$$

$W^V, W^Q, W^J \epsilon R^{d_x \times d_z}$ are matrix parameters. These matrices are unique for each attention head and layer.

## III. Model Details

Our framework consists of four models, two each on vanilla Recurrent Neural Network and the Bi-directional Recurrent Neural Network architecture. These models use Long Short-Term Memory and the Gated Recurrent Unit cells which are described below.

### A. Long Short-Term Memory

In traditional RNN, the sum of inputs is calculated using $tanh$ whereas an LSTM cell, depicted in Fig 3, computes on the sum of inputs with a $k^{th}$ cell memory $c_t^k$ at time $t$ and produces the output as

$$h_t^k = o_t \tanh\left(c_t^k\right), \tag{7}$$

where $o_t$ is the output gate that controls the amount of memory content exhibited in LSTM cell. This output gate for $k^{th}$ cell is computed as

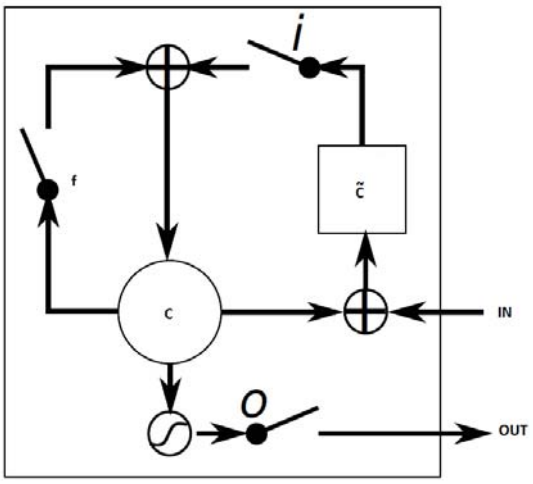


Fig. 3. An LSTM cell with forget ($f$), input ($i$) and output ($o$) gates, $c$ as memory cell and $\bar{c}$ as new memory cell content.

$$o_t^k = \sigma\left(W_o x_t + U_o h_{t-1} + V_o c_t^k\right)^k, \tag{8}$$

where $\sigma$ is a sigmoid function, $W_o$ is the input to hidden layer weight, $U_o$ is the hidden layer to hidden layer weight and $V_o$ is the diagonal matrix. At each time step the memory cell $c_t^k$ is updated using following equation

$$c_t^k = f_t^k c_{t-1}^k + i_t^k \bar{c}_t^k, \tag{9}$$

where $\bar{c}_t^k$ is given as follows

$$\bar{c}_t^k = \tanh\left(W_c x_t + U_c h_{t-1}\right)^k, \tag{10}$$

Forget gate $f_t^k$ controls the amount of existing memory to be forgotten while input gate $i_t^k$ controls the amount of new content to be added. These are evaluated as follows

$$f_t^k = \sigma\left(W_f x_t + U_f h_{t-1} + V_f c_{t-1}^k\right)^k, \tag{11}$$

$$i_t^k = \sigma\left(W_i x_t + U_i h_{t-1} + V_i c_{t-1}^k\right)^k. \tag{12}$$

Note that $V_o$, $V_f$ and $V_i$ are diagonal matrices.

### B. Gated Recurrent Unit

In GRU, as depicted in Fig 4, activation $h_t^k$ for $k^{th}$ recurrent unit at time $t$ is the linear interpolation between previous activation $h_{t-1}^k$ and future candidate activation $\bar{h}_t^k$. It is given as

$$h_t^k = \left(1 - z_t^r\right) h_{t-1}^r + z_t^r \bar{h}_t^k, \tag{13}$$

The update gate $z_t^k$ decides the content to be updated by the unit and can be calculated as

$$z_t^k = \sigma\left(W_z x_t + U_z h_{t-1}\right)^k, \tag{14}$$

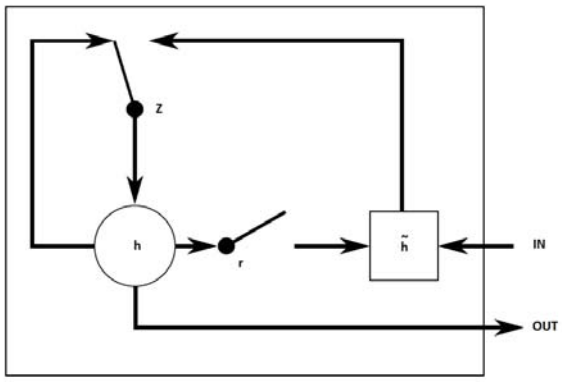


Fig. 4. A GRU cell with reset ($r$) and update ($z$) gates, and $h$ as the activation and and $\tilde{h}$ as the candidate activation.

Reset gate allows GRU to forget the preceding computations done on input and reads the new input as if it is the first word of the sequence. It is calculated as:

$$r_t^k = \sigma \left( W_r x_t + U_r h_{t-1} \right)^k . \tag{15}$$

Here $\sigma$ is the sigmoid function, $W_r$ and $W_z$ are the weights from input to hidden layer and, $U_r$ and $U_z$ are the weights from one hidden unit to its next hidden unit in layer.

### C. Self-Attentional Long Short-Term Memory

SALSTM uses the vanilla Recurrent Neural Network with Long Short-Term Memory cells instead of the original $tanh$ function used in vanilla RNN. At the top of the vanilla network, we have applied the self-attention mechanism which provides a set of summation of weight vectors for the network's hidden state.

### D. Self-Attentional Gated Recurrent Unit

SAGRU uses the vanilla Recurrent Neural Network with Cho et al.'s [15] Gated Recurrent Unit cells instead of the $tanh$ function used in vanilla RNN. At the top of the vanilla network, we have applied the self-attention mechanism which provides a set of summation of weight vectors for the network's hidden state.

### E. Self-Attentional Bi-Long Short-Term Memory

SABLSTM uses the Bi-directional Recurrent Neural Network with Long Short-Term Memory cells. In this bidirectional network, the output of the two independent RNN's is concatenated together. At the top of the BRNN, we have applied the self-attention mechanism which provides a set of summation of weight vectors for the network's hidden states moving in both directions.

### F. Self-Attentional Bi-Gated Recurrent Unit

SABGRU uses the Bi-directional Recurrent Neural Network with Cho et al.,'s [15] Gated Recurrent Unit cells. Again, the output of the two independent RNN's is concatenated together in the BRNN. At the top of the BRNN, we have applied the self-attention mechanism which provides a set of summation of weight vectors for the network's hidden states moving in both directions.

## IV. Experiments

### A. Datasets

We have tested our model on seven benchmark datasets namely MR, CR, TREC, MPQA, Subj, SST-1 and SST-2. A summary of these datasets is given in Table I.

TABLE I
SUMMARY OF ALL DATASETS. HERE $\mathbf{L_{av}}$: AVERAGE LENGTH OF SENTENCES, $\mathbf{Classes}$: NUMBER OF CATEGORIES IN DATASET, $\mathbf{Size}$: NUMBER OF SENTENCES IN DATASET, $|\mathbf{V}|$: SIZE OF VOCABULARY OF DATASET, $|\mathbf{V_{pre}}|$: NUMBER OF WORDS OBTAINED FROM PRE-TRAINED MODEL. $\mathbf{Test}$: SIZE OF TEST DATA, 10CV: 10-FOLD CROSS VALIDATION (APPLIED WHENEVER TEST SET WAS NOT GIVEN).

| $\mathbf{Datasets}$ | $\mathbf{L_{av}}$ | $\mathbf{Classes}$ | $\mathbf{Size}$ | $\lvert\mathbf{V}\rvert$ | $\lvert\mathbf{V_{pre}}\rvert$ | $\mathbf{Test}$ |
|---|---|---|---|---|---|---|
| MR | 20 | 2 | 10662 | 19585 | 16303 | 10CV |
| CR | 19 | 2 | 3775 | 4508 | 4254 | 10CV |
| MPQA | 3 | 2 | 10606 | 6195 | 5964 | 10CV |
| Subj | 23 | 2 | 10000 | 22304 | 17799 | 10CV |
| SST-2 | 19 | 2 | 9613 | 16132 | 14737 | 1821 |
| TREC | 10 | 6 | 5952 | 8590 | 7364 | 500 |
| SST-1 | 18 | 5 | 18553 | 17790 | 16179 | 2210 |

- **MR:** MR dataset consists of movie reviews with one sentence per review. Each review is either a positive or a negative review [28].
- **CR:** It consists of customer reviews of various products labeled positive or negative [29].
- **TREC:** A question dataset where each question is categoried into numeric, location, person, abbrevation, description and entity [30].
- **MPQA:** It consists of opinions categorized into negative or positive opinions [31].
- **Subj:** A subjective dataset where sentences are labeled as being subjective or objective [32].
- **SST-1:** Stanford Sentiment Treebank dataset. It is an extension of MR but with separate training and testing set. The sentences are labeled very negative, negative, positive, very positive and neutral. They were re-labeled by Socher et al., [33].
- **SST-2:** SST-2 dataset is similar to SST-1 dataset but with binary labels i.e. positive and negative reviews [33].

### B. Implementation Detail

The data on which we train our models is firstly pre-processed to remove the stop words from each sentence. The remaining word form our vocabulary. Once we have the vocabulary, we load Mikolov's [18] pre-trained word2vec model to obtain the 300-dimensional word vectors for every word in our vocabulary. We randomly initialize the vocabulary words not found in the Mikolov's model. To ensure proper training, we train the obtained word vectors on their respective datasets. Therefore, these non-static word vectors are more

TABLE II
RESULT OF OUR SELF-ATTENTIONAL NON-STATIC RNN MODELS AGAINST THE BASELINE CNN MODELS OF YOON KIM. (BOLD ITEM REPRESENTS IMPROVED RESULT OVER BASELINE MODEL WHILE BOLD WITH ITALIC REPRESENTS THE NEW STATE-OF-THE-ART RESULT.)

| Model | MR | CR | MPQA | Subj | SST-2 | TREC | SST-1 |
|---|---|---|---|---|---|---|---|
| CNN-rand | 76.1 | 79.8 | 83.4 | 89.6 | 82.7 | 91.2 | 45.0 |
| CNN-static | 81.0 | 84.7 | 89.6 | 93.0 | 86.8 | 92.8 | 45.5 |
| CNN-non-static | 81.5 | 84.3 | 89.5 | 93.4 | 87.2 | 93.6 | ***48.0*** |
| CNN-multichannel | 81.1 | 85.0 | 89.4 | 93.2 | **88.1** | 92.2 | 47.4 |
| SALSTM | **81.53** | 84.83 | **89.68** | 92.94 | 87.80 | **95.18** | 47.81 |
| SAGRU | ***82.47*** | 84.46 | ***89.91*** | **93.58** | 88.06 | **95.39** | 47.96 |
| SABLSTM | **82.10** | 84.35 | 89.49 | 93.40 | 87.73 | 94.91 | 47.88 |
| SABGRU | **81.98** | ***85.98*** | **89.64** | **93.51** | 87.64 | ***96.31*** | 47.65 |
| Embedded-LSTM | 80.96 | 84.03 | 89.34 | 91.10 | 84.17 | 92.35 | 46.65 |
| Embedded-GRU | 81.43 | 82.76 | 89.23 | 93.10 | 83.42 | 92.11 | 46.54 |
| Embedded-BLSTM | 81.19 | 83.02 | 89.25 | 92.40 | 83.80 | 91.68 | 46.33 |
| Embedded-BGRU | 81.24 | 84.88 | 89.32 | 92.80 | 83.91 | 91.83 | 45.97 |

TABLE III
RESULTS WITH BAHDANAU'S ATTENTIONAL MODEL [35].

| Model | MR | CR | MPQA | Subj | SST-2 | TREC | SST-1 |
|---|---|---|---|---|---|---|---|
| ALSTM | 81.16 | 84.08 | **89.69** | 92.4 | 84.07 | 94.66 | 46.61 |
| AGRU | **82.18** | **84.62** | 89.62 | **93.5** | 84.01 | 94.79 | 46.74 |
| ABLSTM | **81.89** | 83.29 | 89.43 | 92.90 | 84.02 | 94.13 | 45.88 |
| ABGRU | 81.34 | **85.68** | 89.51 | 93.30 | 83.42 | 94.42 | 46.24 |

TABLE IV
RESULT OF OUR MODELS WITH STATIC WORD VECTORS. (BOLD ITEM REPRESENTS IMPROVED RESULT OVER BASELINE MODEL)

| Model | MR | CR | MPQA | Subj | SST-2 | TREC | SST-1 |
|---|---|---|---|---|---|---|---|
| SALSTM-Static | 80.96 | 83.41 | 89.11 | 92.00 | 83.62 | 92.05 | 46.36 |
| SAGRU-Static | **81.93** | 84.37 | 89.26 | 93.16 | 83.58 | 91.94 | 46.38 |
| SABLSTM-Static | 81.39 | 82.86 | 88.98 | 92.21 | 83.55 | 91.40 | 45.72 |
| SABGRU-Static | 81.13 | **85.02** | 89.04 | 92.82 | 83.27 | 91.76 | 46.14 |

task-specific than the general word vectors obtained from the pre-trained model.

The final input sequence fed to our networks is, therefore, given as:

$$x_{1:n} = x_1 \oplus x_2 \oplus ... \oplus x_n \tag{16}$$

where $\oplus$ is the concatenation operator and $x_i$ represents the $i^{th}$ word in the sequence.

Recurrent neural networks can be quite unstable while training which sometimes makes it extremely difficult to train. Therefore, to stablize the training, we initialize the kernel and recurrent weights orthogonally since orthogonal matrix has absolute value 1. Thus, ensuring that the weights in network are stable irrespective of the number of times repeated multiplication is performed in each iteration. Further, we apply $L_2$ norm to both these weights and $L_1$ norm on activity regularizer. In addition, we apply batch normalization to each layer. Batch normalization parameters are initialized to transform the input to zero mean or unit variance distribution yet amid the training, the network can discover that some other distribution may be better. Thus, batch normalization enables each layer to learn a little more by itself. Further, for regularization in the network, dropout is used on the hidden layers. The final fully connected dense layer is implemented with sigmoid activation.

A random set of test data has been used for the datasets with no explicit test set. Also, ten-fold cross-validation has been used to dimish the impact of randomness. Training is done using Adam [34] optimization algorithm - an extension of stochastic gradient descent, performed over shuffled mini batch.

RNNs are known to be a sequence based network. To test their mettle, we implement the non-attentional versions of the architecture with tuned word vectors and same configuration of the network. We term these Embedded-LSTM and Embedded-GRU on unidirectional RNN and Embedded-BLSTM and Embedded-BGRU on bidirectional architecture.

Further, to test the mettle of self-attention model against Bahdanau's attention model we compare our self-attention based results in this work with the results obtained on Bahdanau's attention by four models ALSTM, AGRU, ABLSTM and ABGRU in our previous work [35].

Furthermore, to test the effect of non-static word embedding and the generality of Mikolov's pre-trained word2vec model, we implement the self-attention networks with static word vectors as well.

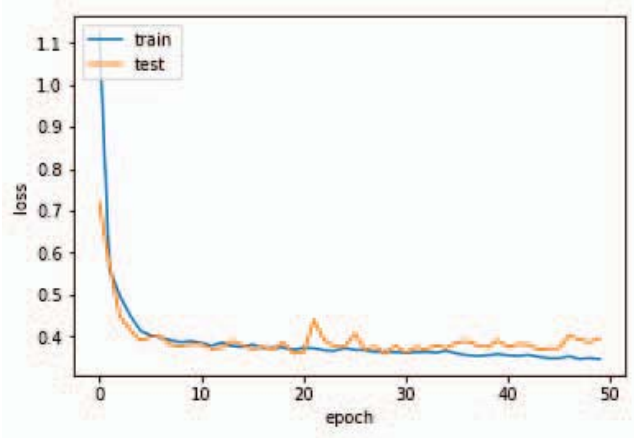


Fig. 5. Training and testing loss with default (uniform normal) recurrent weights on MPQA dataset.

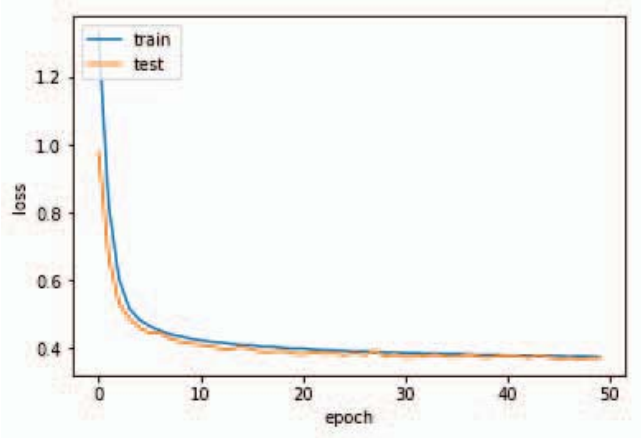


Fig. 6. Training and testing loss with orthogonally initialized recurrent weights.

### C. Results

The results for our models against the baseline model of Yoon Kim [36] are listed in Table II. Yoon Kim implemented four models based on Convolution Neural Network. The first model **CNN-rand** implements the basic CNN architecture with randomly initialized word vectors. The second model, **CNN-static** uses pre-trained static word vectors for classification using CNN architecture. Third model, **CNN-non-static** implements CNN with fine-tuned word vectors. The vectors are trained on all datasets to make them more task specific. The final model of Yoon Kim, **CNN-multichannel** implements two sets of word vectors for training. One set is trained while the other is kept static.

Our results emphasize that every model has performed well on all the seven datasets with self-attention models outperforming the non-attentional models consistently. Also, to our surprise self-attention models, SAGRU and SALSTM have performed better than bidirectional self-attention models SABGRU and SABLSTM. Our models have produced better results as compared to the baseline models on CR, MPQA, MR, TREC and Subj datasets while the results on SST-1 and SST-2 datasets are comparable. Four of the results, 82.47% on MR by SAGRU, 85.98% on CR by SABGRU, 89.91% on MPQA by SAGRU and 96.31% by SABGRU on TREC are better than the previous state-of-the-art results achieved by Yoon Kim's models.

This result shows that the basic RNN architectures, i.e. RNN and BRNN are able to learn the pattern in the sequence and are producing strong results thus proving the sturdiness of recurrent neural networks in the task of sequence learning.

Table III lists the results obtained with Bahdanau's attention model as shown in our previous work [35]. The results, even though stronger than some of the baseline models of Yoon Kim, are still consistently lower against the results obtained by self-attention models. Thus, showing the superiority of self-attention model over Bahdanau's attention model for simple sequential tasks like sentence classification.

The result of implementation with static word vectors in Table IV shows that the static implementations are almost as good as non-static. Even though the results are consistently low, but the difference is very small. Also, training the static models takes less time than to train the non-static models. So, there is a trade-off between training time and accuracy.

Figure 5 and 6 show the training and testing loss for training the model on MPQA dataset with default random uniform weights and orthogonal weights respectively. We can clearly see that the training with orthogonal weights is more stable as compared to the Keras default random uniform weight initialization. Thus, showing the importance of proper weight initialization for recurrent networks. As the number of epoch increases, we observed that the loss became saturated for orthogonally initialized networks.

### D. Observations

We make following observations from our experiment:

- Attentional models perform better than non-attentional models.
- Self-attention model performs better than Bahdanau's attention model at the task of classification.
- GRU cell based networks perform better than Long Short-Term Memory cell networks at the task of sentence classification.
- The basic architectures of Recurrent Neural Network learn the features effectively and produce good results which proves the mettle of RNNs at sequential tasks.
- Word vectors trained for specific task improve the performance of the task over the training done on static vectors. However, the difference is not huge and thus shows that Mikolov's pre-trained word2vec vectors can be universally accepted vectors.
- Orthogonal weight initialization, $L_1$, and $L_2$ norm on weights, and batch normalization together improve the stability of the RNN network.

## V. Conclusion

In this paper, we have done a series of experiments with vanilla and Bi-directional Recurrent Neural Network using Long Short-Term Memory and Gated Recurrent Unit cells. We implemented the architectures with Self-Attention, attention and without attention using static and non-static word vectors obtained from Mikolov's pre-trained word2vec model. Our results confirm that RNN, being a sequential network, is good at learning dependencies in the text both with LSTM and GRU cells. Attention models support RNN strongly by making the network learn dependencies in text over longer lengths. Self-attention, in particular, improves the RNN for sentence classification. Our results confirm this. Also, a small difference in the results with static and non-static vector suggests that Mikolov's pre-trained word2vec vectors are universal in nature.

## References


[1] J. C. Chang and C.C. Lin, "Recurrent-neural-network for language detection on Twitter code-switching corpus." arXiv preprint arXiv:1412.4314 (2014).

[2] L. Bing, and I Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling." arXiv preprint arXiv:1609.01454 (2016).

[3] P. Li, J. Li, F. Sun, and P. Wang, "Short Text Emotion Analysis Based on Recurrent Neural Network." In Proceedings of the 6th International Conference on Information Engineering, p. 6. ACM, 2017.

[4] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification." In Proceedings of the 2015 conference on empirical methods in natural language processing, pp. 1422-1432. 2015.

[5] P. Jotikabukkana, V. Sornlertlamvanich, O. Manabu, and C. Haruechaiyasak, "Effectiveness of social media text classification by utilizing the online news category." In Advanced Informatics: Concepts, Theory and Applications (ICAICTA), 2015 2nd International Conference on, pp. 1-5. IEEE, 2015.

[6] R. P. Schumaker, and H. Chen, "Textual analysis of stock market prediction using breaking financial news: The AZFin text system." ACM Transactions on Information Systems (TOIS) 27, no. 2 (2009): 12.

[7] M. Hughes, I. Li, S. Kotoulas, and T. Suzumura, "Medical text classification using convolutional neural networks." Stud Health Technol Inform 235 (2017): 246-50.

[8] R. M. Pirinen, "The construction of womens positions in sport: A textual analysis of articles on female athletes in Finnish womens magazines." Sociology of Sport Journal 14, no. 3 (1997): 290-301.

[9] A. Graves, A. R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks." In Acoustics, speech and signal processing (icassp), 2013 ieee international conference on, pp. 6645-6649. IEEE, 2013.

[10] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator." In Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on, pp. 3156-3164. IEEE, 2015.

[11] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks." In Advances in neural information processing systems, pp. 3104-3112. 2014.

[12] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions." International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 6, no. 02 (1998): 107-116.

[13] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks." In International Conference on Machine Learning, pp. 1310-1318. 2013.

[14] S. Hochreiter, and J. Schmidhuber, "Long short-term memory." Neural computation 9, no. 8 (1997): 1735-1780.

[15] K. Cho, B. V. Merrinboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation." arXiv preprint arXiv:1406.1078 (2014).

[16] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling." arXiv preprint arXiv:1412.3555 (2014).

[17] M. Schuster, and K. K. Paliwal, "Bidirectional recurrent neural networks." IEEE Transactions on Signal Processing 45, no. 11 (1997): 2673-2681.

[18] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality." In Advances in neural information processing systems, pp. 3111-3119. 2013.

[19] G. Pollastri, D. Przybylski, B. Rost, and P. Baldi, "Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles." Proteins: Structure, Function, and Bioinformatics 47, no. 2 (2002): 228-235.

[20] G. Arevian, "Recurrent neural networks for robust real-world text classification." In Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, pp. 326-329. IEEE Computer Society, 2007.

[21] O. Irsoy, and C. Cardie, "Opinion mining with deep recurrent neural networks." In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 720-728. 2014.

[22] Y. Tang, and J. Liu. "Gated Recurrent Units for Airline Sentiment Analysis of Twitter Data."

[23] J. Wang, L.C. Yu, K. R. Lai, and X. Zhang, "Dimensional sentiment analysis using a regional CNN-LSTM model." In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), vol. 2, pp. 225-230. 2016.

[24] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. H. Tur, X. He, "Using recurrent neural networks for slot filling in spoken language understanding." IEEE/ACM Transactions on Audio, Speech, and Language Processing 23, no. 3 (2015): 530-539.

[25] J. Y. Lee, and F. Dernoncourt, "Sequential short-text classification with recurrent and convolutional neural networks." arXiv preprint arXiv:1603.03827 (2016).

[26] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).

[27] Z. Lin, M. Feng, C. N. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding." arXiv preprint arXiv:1703.03130 (2017).

[28] B. Pang, and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales." In Proceedings of the 43rd annual meeting on association for computational linguistics, pp. 115-124. Association for Computational Linguistics, 2005.

[29] M. Hu, and B. Liu, "Mining and summarizing customer reviews." In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 168-177. ACM, 2004.

[30] X. Li, and D. Roth, "Learning question classifiers: the role of semantic information." Natural Language Engineering 12, no. 3 (2006): 229-249.

[31] J. Wiebe, T. Wilson, and C. Cardie, "Annotating expressions of opinions and emotions in language." Language resources and evaluation 39, no. 2-3 (2005): 165-210.

[32] B. Pang, and L. Lee, "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts." In Proceedings of the 42nd annual meeting on Association for Computational Linguistics, p. 271. Association for Computational Linguistics, 2004.

[33] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank." In Proceedings of the 2013 conference on empirical methods in natural language processing, pp. 1631-1642. 2013.

[34] D. P. Kingma, and J. Ba, "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[35] A. Kumar, and R. Rastogi. "Attentional Recurrent Neural Networks for Sentence Classification." In Innovations in Infrastructure, pp. 549-559. Springer, Singapore, 2019.

[36] Y. Kim, "Convolutional neural networks for sentence classification." arXiv preprint arXiv:1408.5882 (2014).