**IE643: Deep Learning - Theory and Practice** **July-Nov 2022**

End-term Project Report : Making the Best Use of Review Summary for Sentiment Analysis

*Team Name: DeepMind2.0* *Team Members: 200100154, 200100173*

**Abstract**

This report contains the details on our project which deals with the problem of achieving a better sentiment classification by using additional information like summary that comes in addition with the review. It has been found that indeed the sentimental signal distribution of a review and that of its corresponding summary are in fact complementary to each other. Hence, we initially describe a review-centric attention model with Bi-LSTM and show empirically it's effectiveness with respect to other baseline models. We have also experimented with a completely different model like Bi-GRU. We have played around with the number of hidden layers to see it's effect on the final accuracy.

# 1 Introduction

Sentiment analysis is a fundamental task in natural language processing, which predicts the subjectivity and polarity of a given text. In practical scenarios, sentiment is usually extracted automatically from user reviews and this finds wide applications in E-commerce and movie reviews. So far, people have trained several models that only take review information as input for classifying the sentiment.

In many review websites such as Amazon, the user is allowed to give a summary in addition to the review, where summaries contain more general information about the review. Hence, summaries can be used as additional training signals. It is thus an interesting research question on how to make the best use of both review and summary information for better sentiment classification in such cases.

Certainly, the goal of the project is to achieve a highly accurate sentiment classification which will ultimately empower businesses who can now monitor their brand and product sentiment in customer feedback and accordingly understand customer needs. This could also be extremely useful in intent analysis which involves analyzing the user's intention behind a message and identifying whether it relates an opinion, news, marketing, complaint, suggestion, appreciation or query.

We provide a survey of existing literature in Section 2. Our proposal for the project is described in Section 4. We give details on experiments in Section 6. A description of future work is given in Section 8. We conclude with a short summary and pointers to forthcoming work in Section 10.

# 2 Literature Survey

Though our project is primarily based on experimenting with the review-centric attention model proposed by Yang et al.,2020 [1]. However, we have researched about various other models that people have so far tried out to understand the reason of success behind the respective architectures.

We first tried to look into the paper by Pang et al., 2002 [2] who proposed several machine learning techniques to determine whether a review is positive or negative by using movie reviews as data. However, the

three machine learning methods they employed (Naive Bayes, maximum entropy classification, and support vector machines) do not perform as well on sentiment classification as on traditional topic-based categorization.

We then looked into the paper by Zhang et al., 2015 [3] who proposed character-level convolutional networks (ConvNets) for text classification. This architecture takes the entire review as input and does not make use of summary information in any way.

Different from previous work, we additionally consider user-generated or automatically-generated summaries as input. Our work is related to existing work on joint summarization and sentiment classification. Ma et al., 2018 [4] who proposed a multi-view attention model for joint summarization and sentiment classification. Wang et al., 2018 [5] improved the model of Ma et al., 2018 [4] by using additional attention on the generated text. However, they both were interested in generating better summaries from the review.

Instead, we make a broader discussion on how to make the best use of both review and summary for sentiment classification.

# 3 Methods and Approaches

## 3.1 Problem Formulation

- Input $\rightarrow (X^w, X^s)$ where $X^w = x_1^w, x_2^w, ..., x_n^w$ is a review and $X^s = x_1^s, x_2^s, ..., x_n^s$ is the corresponding summary
- Task $\rightarrow$ Predict sentiment label, y $\in$ [1,5] where 1 $\rightarrow$ most negative sentiment & 5 $\rightarrow$ most positive sentiment
- n $\rightarrow$ #words in review & m $\rightarrow$ #words in summary

## 3.2 Questions answered empirically through this research

- What are the roles of and the correlation between a review and its summary for predicting the user rating?
- How to better leverage information from both the review and the summary for effective sentiment classification?

## 3.3 Architecture

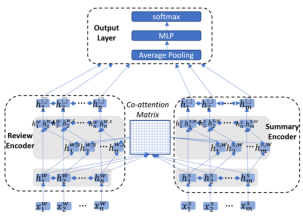



Figure 1: Co-attention joint encoder architecture

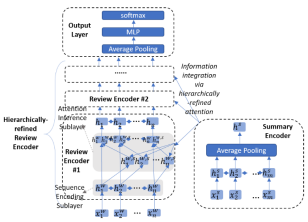



Figure 2: Review-centric joint encoder architecture

### 3.3.1 Sequence Encoding

Bi-LSTM was used as the sequence encoder for all experiments. The input is a sequence of word representations $x = x_1, x_2, ..., x_m = emb(x_1), ..., emb(x_m)$, where emb denotes a word embedding lookup table. Word representations are fed into a standard Bi-LSTM. We adopt a standard LSTM formulation, where a sequence of hidden states $h_t$ are calculated from the sequence of $x_t$ (t ∈ [1, ..., m]).

A forward left-to-right LSTM layer and a backward right-to-left LSTM yield a sequence of forward hidden states and a sequence of backward hidden states. The two hidden states are concatenated to form a final representation:

$$\mathbf{h}_i = [\overrightarrow{\mathbf{h}_i}; \overleftarrow{\mathbf{h}_i}]$$
$$\mathbf{H} = \{\mathbf{h}_1, ..., \mathbf{h}_m\}$$

This encoder structure serves as a basis for all the models. In particular, for the review-only and summary-only baselines, we use a single encoder as described above.

### 3.3.2 Baselines

- Separate Encoding - Two Bi-LSTMs are adopted to separately encode reviews and summaries
  - Average-pooling baseline - the hidden state matrices are concatenated and then average-pooled to form a final representation for later prediction
  - Self-attention baseline - the hidden state matrices are separately processed using self-attention mechanism. Subsequently the two matrices are concatenated and average-pooled to produce the final representation for later prediction
- Symmetric Joint Encoding - On top of the sequence encoder, they separately adopted several mechanisms
  - Pooling
  - Self-attention
  - Hard-attention
  - Co-attention

As for co-attention baselines, we use two Bi-LSTMs to separately encode review and summary. The two hidden state matrices then interact with each other. The formulations are written as :

$$\mathbf{A} = \mathbf{H}^w \mathbf{W}^w (\mathbf{H}^s \mathbf{W}^s)^\intercal$$
$$\mathbf{H}^w_{co\text{-}att} = \mathbf{H}^w + \text{softmax}(\frac{\mathbf{A}}{\sqrt{d}})\mathbf{H}^s$$
$$\mathbf{H}^s_{co\text{-}att} = \mathbf{H}^s + \text{softmax}(\frac{\mathbf{A}^\intercal}{\sqrt{d}})\mathbf{H}^w$$

where $H_w$ and $H_s$ are the hidden states of reviews and summaries, respectively. $\mathbf{A} \in \mathbb{R}^{n \times m}$ is the co-attention matrix. $n$ and $m$ are the lengths of the review text and the summary text, respectively. $d$ represents the hidden size of the Bi-LSTM. $H^w_{co-att}$ and $H^s_{co-att}$ are the co-attention representations of a review

and its corresponding summary, respectively. They are then fed into subsequent layers for making predictions.

#### 3.3.3 Review-centric Joint Encoding

- **Changed only the review encoder** over the baseline
- The review encoder has a set of **stacked layers**, each consisting of a sequence encoding sublayer and an attention inference sublayer
- The **sequence encoding sublayers** takes the same BiLSTM structure as the summary encoder, but with different model parameters
- The **attention inference sublayer** integrates summary information into the review representation

$$\begin{aligned}\boldsymbol{\alpha} &= \text{softmax}(\frac{\mathbf{H}^w\mathbf{W}_i^Q(\mathbf{h}^s\mathbf{W}_i^K)^\intercal}{\sqrt{d_h/k}}) \\ \mathbf{head}_i &= \hat{\mathbf{A}}(\hat{\mathbf{H}}^s)^\intercal\mathbf{W}_i^V \\ \mathbf{H}^{w,s} &= \text{concat}(\mathbf{head}_1, \ldots, \mathbf{head}_k)\end{aligned}$$

Following Vaswani et al. (2017), we adopt a residual connection around each attention inference layer:

$$\mathbf{H} = \text{LayerNorm}(\mathbf{H}^w + \mathbf{H}^{w,s})$$

#### 3.3.4 Output Layer

Global average pooling is applied on $\mathbf{H}$, followed by a classifier layer:

$$\begin{aligned}\mathbf{h}^{avg} &= \text{avg-pooling}(\mathbf{h}_1, ..., \mathbf{h}_n) \\ \mathbf{p} &= \text{softmax}(\mathbf{W}\mathbf{h}^{avg} + \mathbf{b}) \\ \hat{y} &= \text{argmax}\,\mathbf{p}\end{aligned}$$

where $\hat{y}$ is the predicted sentiment label; $\mathbf{W}$ and $\mathbf{b}$ are parameters to be learned during **training**.

Models were trained by using the following **Loss Function** :

$$L = -\sum_{t=1}^{|T|}\log(\mathbf{p}^{[y_t]})$$

where $p^{[yt]}$ denotes the value of the label in $\mathbf{p}$ that corresponds to $y_t$.

# 4 Work Done

## 4.1 Work done before mid-term project review

Prior to the mid-term project review, most of our efforts went into thoroughly reading the paper [1] from which we drew our major inspiration, as well as two related papers. We also spent good amount of time to understand all aspects of the problem statement as well as the various methods used to tackle it. LSTMs and GRUs were previously unknown to us, and we understood not only these basic models but also advanced models such as Bi-LSTMs and Bi-GRUs. We theoretically understood the importance of the co-attention layer as well as global average pooling. We learned the basics of the PyTorch framework to make it easier for us to use it ahead. We also started our work by first procuring the usable dataset for Amazon reviews on Toys and Games which was available online and then used it to train the Bi-LSTM based review-centric attention model whose code was made available by Yang et al. on GitHub. We initially tried to train it locally on our PC setup which had Cuda 11.3 Environment. During the training phase, the data pre-processing part was completed successfully. However, we were able to train the model up to 2 epochs only post which the training did not show any progress since our PC setup was running out of GPU memory. We finally decided to leverage the GPU offered by Google Colab. There were a lot of ideas for modifications that we felt could lead to even better sentiment classification however we also discussed and spent time on understanding and selecting the ones that were really worth trying. The **main modification** we planned to experiment with was to change the Bi-LSTM block of the model to a Bi-GRU block. Some other modifications we felt were worth trying out were :

- Train on a smaller subset of the training data and then consider increasing the subset.
- Change the number of layers present in their proposed model from 2 to 3.
- Use dropout instead of the global average pooling
- Replace PG-Net with something more state-of-the-art for summary generation
- Add the summary generated by the model to the actual summary given by the user.

## 4.2 Work done after mid-term project review

After the mid-term project review, we progressed with the construction of the Bi-GRU model that we had proposed during the mid-term project review in PyTorch and consolidated the dataset. We simultaneously successfully completed the training of the review-centric Bi-LSTM model. Hyper-parameters were also tuned appropriately and were kept at those values for which the model would not take a lot of time to train. Then we were also able to train the Bi-GRU model. This roughly took us about a week. We then tackled the problem of changing the number of layers within the Bi-LSTM model from 2 to 3. We also decided to experiment with the number of layers within the Bi-GRU as well and hence we trained a Bi-GRU model with 3 hidden layers. Finally as a part of our end-term project review, we also prepared a video wherein we go over our code along with another video wherein we demonstrate the different experiments that we have conducted.

# 5 Data set Details

The SNAP Amazon Review Dataset (McAuley and Leskovec, 2013) [6] consists of around 34 million Amazon reviews in different domains, such as books, games, sports and movies. Each review mainly consists

of a product ID, a piece of user information, a plain text review, a user-written summary and an overall sentiment rating which ranges from 1 to 5.

For training purposes however, we have restricted ourselves to only 1 dataset i.e. Toys & Games. Within this domain, there are about 168,000 products with the average length of the reviews being around 99.9 and the average length of the corresponding summaries being around 4.4.

For each dataset, the first 1000 samples are taken as the development set, the next 1000 samples as the test set, and the rest as the training set.

Before using the dataset for training, few data preprocessing techniques were adopted and they are as follows:

- Tokenizing the dataset
- Processing word embedding files for the given dataset
- Saving the tokenized dataset, word to index, and index to vector pickle files

# 6 Experiments

Since we were using Google Colab, our training was heavily dependent on the GPU that was given to us during that particular runtime.

After having successfully trained the model as described in the paper using a Bi-LSTM, the first experiment that was done was changing the Bi-LSTM block of the model to a Bi-GRU (Bidirectional Gated Recurrent Unit) in PyTorch. The Bi-LSTM consists of a cell state, which can be thought of as a conveyor belt running in both directions through all the Bi-LSTM cells and a hidden state. The information to be removed and added to the cell state is regulated by structures known as gates. They consist of a sigmoid neural network layer and a pointwise multiplication function, and they are a way of optionally letting information through. A Bi-LSTM cell has an input (to decide what to update), forget (to decide what information to retain) and output gate (to decide what to output).

A Bi-GRU combines the forget and the input gates into a single update gate, and merges the cell state and hidden state into a reset gate. The operations among these are also modified accordingly. As the number of gates are reduced, the resulting model is far less complex. It is also approximately as good as an Bi-LSTM despite the reduction in complexity and is generally used when training needs to be done quickly and with decent accuracy, or if there are infrastructural constraints. We hypothesized that this could speed up training without compromising much on accuracy. When implementing the Bi-GRU model, all other model parameters and layers were kept the same as before. To be more precise, we also used GloVe : 300-dimensional embeddings as pretrained word vectors. The Bi-GRU hidden size was set to 256. We used Adam to optimize the model, with a learning rate of $3e{-}4$, decay rate of 1 and a decay interval of 200. The number of epochs for which the model was trained is 10 with a batch size of 128.

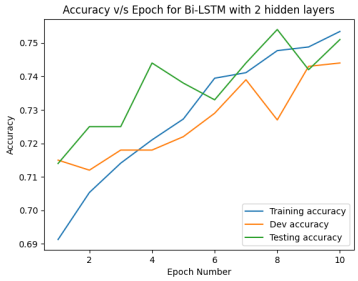
The dropout rate was kept at 0.5 and number of attention heads was kept at 1. We initially trained the Bi-GRU with 2 hidden layers. However, we also trained another Bi-GRU model with 3 hidden layers. We finally trained another Bi-LSTM model with 3 hidden layers with the same parameters.

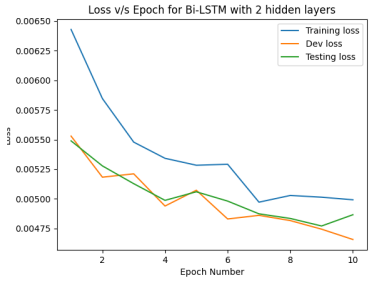
# 7 Results

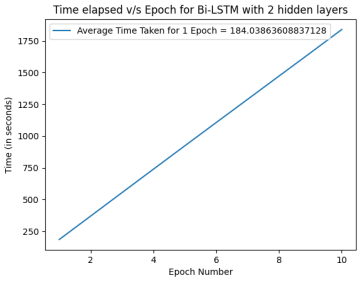
All the models were trained for a total of 10 epochs on 1200 batches with every batch having a size of 128.

After training the Bi-LSTM with 2 hidden layers, we were able to obtain a test accuracy close to 75.1, test loss of around 0.004834 and average test time for every epoch was around 184 seconds.

Here are the corresponding plots obtained for accuracy, loss and test time :

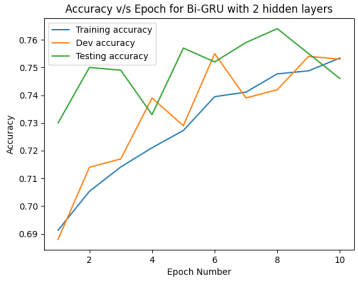


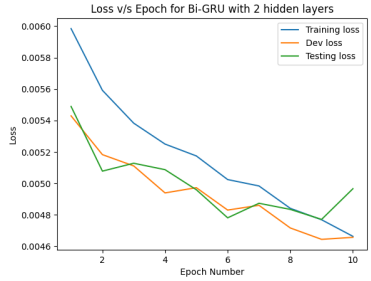



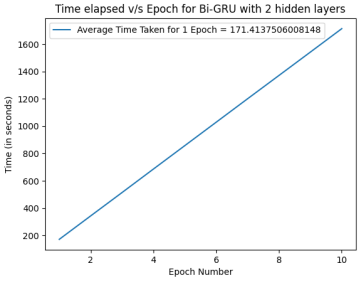




After training the Bi-GRU with 2 hidden layers, we were able to obtain a test accuracy close to 74.6, test loss of around 0.004966 and average test time for every epoch was around 171 seconds.

Here are the corresponding plots obtained for accuracy, loss and test time :

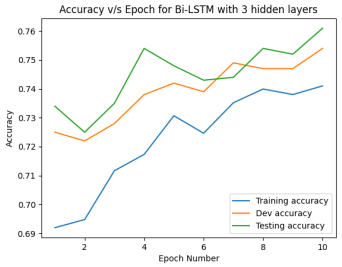


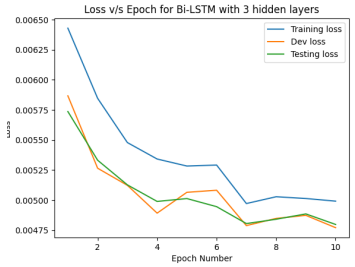



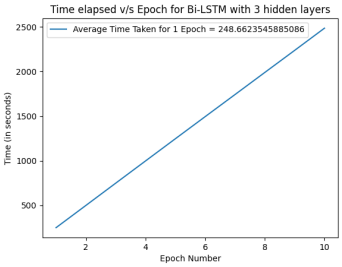




After training the Bi-LSTM with 3 hidden layers, we were able to obtain a test accuracy close to 76.4, test loss of around 0.00479 and average test time for every epoch was around 249 seconds.

Here are the corresponding plots obtained for accuracy, loss and test time :

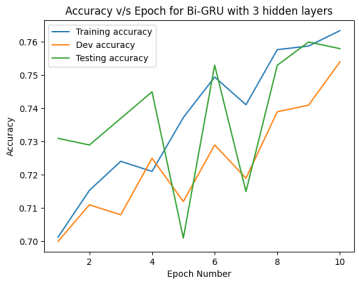


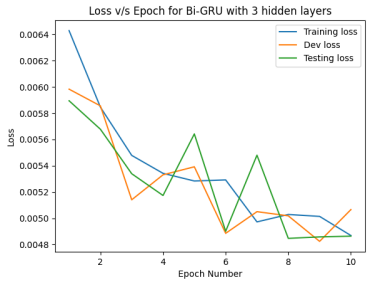



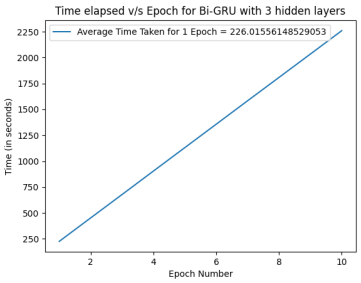




After training the Bi-GRU with 3 hidden layers, we were able to obtain a test accuracy close to 75.8, test loss of around 0.004824 and average test time for every epoch was around 226 seconds.

Here are the corresponding plots obtained for accuracy, loss and test time :




# 8 Future Work

Things that can be explored to get more reliable and robust performances are:

- Current state of the art methods for various NLP problems, especially text classification, leverage attention based Transformer-based architectures like XL-Net. We can try a similar approach in an attempt to make a better use of review summary for effective sentiment classification.
- The summary generation model used in all of the models we trained was PG-Net. One can explore the possibility of replacing it with better and more state of the art summary generation model. This can definitely be a good research topic since meaningful summaries generally result in the correct sentiment getting classified.
- The accuracy of the model in correctly classifying sarcastic sentiments can also be researched upon.
- The sentiment distribution within the training set and it's corresponding impact on the model's testing performance can be studied and explored.

# 9 Conclusion

In this project, we researched and implemented solutions to the problem statement of better sentiment classification through the use of review summary. People so far had devised several models that would use only the review information as input to classify the sentiment. However, it has been empirically shown that meaningful summaries can help in sentiment classification as well and hence can be used as an additional training signal. Hence, we are interested in understanding the roles of and the correlation between the review and it's corresponding summary so that we can come up with an architecture which best leverages both review as well as summary information for effective sentiment classification. Yang et al., 2020 [1] had proposed a review-centric attention model with Bi-LSTM at it's core. Upon training their model, for a total of 10 epochs on 1200 batches with every batch having a size of 128, on SNAP Amazon Review Dataset (Toys & Games) we were able to obtain a test accuracy close to 75.1 %. The modification we proposed was substituting the Bi-LSTM block with Bi-GRU, which we hypothesized would reduce the parameters and the complexity of the network, therefore reducing training time without compromising on accuracy. Our expectations were met, as the Bi-GRU based model achieved almost the same accuracy on the test set while also taking around 8 % less time to reach the same level of loss as the Bi-LSTM network had after 10 epochs. Therefore, Bi-GRUs offer a reliable method of achieving the same classification results with a less complex network architecture and training time. We also attempted training the network with different number of hidden layers. We observed that we were able to improve upon the accuracy by

around 1.4 % but this also resulted in 35 % increase in computational time. Future work that can be carried out in order to improve the sentiment classification are the use of Transformer based architectures like XLNet and the replacement of the existing summary generation model with a state of the art model that can generate highly accurate summaries based on review information.

# 10 References


[1] : **Yang et al, 2020** https://arxiv.org/pdf/1911.02711.pdf

[2] : **Pang et al., 2002** https://aclanthology.org/W02-1011.pdf

[3] : **Zhang et al., 2015** https://arxiv.org/abs/1509.01626

[4] : **Ma et al., 2018** https://arxiv.org/abs/1805.01089

[5] : **Wang et al., 2018** https://proceedings.mlr.press/v95/wang18b.html

[6] : **SNAP Amazon Review Dataset** https://snap.stanford.edu/data/web-Amazon.html