Abstractive Summarization on Amazon Review Using Attentive Bi-LSTM with RRA

Xinyue Ke, Lu Pan, Chang Liu
Department of Computer Science
University College London
ucabxke;uczllpa;uczlcl8@ucl.ac.uk

Abstract

With the development of internet, online shopping has entered peoples lives. After finishing shopping online, people could write the reviews of products on the e-commerce platform. The problem coming with the character of online shopping is that the enormous textual data explosion takes place on the e-commerce platform. Consequently, It could be difficult for customers to capture the valid information and then make purchase decisions. In addition, It also increases the difficulty for merchants to manage their business and products. In this paper, we develop a encoder-decoder architecture to do the abstractive summarization on the reviews from the Amazon. The architecture is based on the standard LSTM model. In order to improve the effectiveness of the abstractive summarization, we introduce the Recurrent Residual Attention from Chengs work. We utilize the bi-directional LSTM in the encoder, while the standard LSTM is used in decoder. By developing this model, we expect that it could be easier for customers to make their own purchasing decisions.

1 Introduction

In recent years, the use of e-commerce increases enormously since the online shopping has become the essential part of most people in life. More and more people and merchants begin to buy and sell products or services via internet. Almost all e-commerce platforms are designed to encourage customers to write reviews about products. Consequently, the enormous textual data explosion takes place on e-commerce platforms. Reviews of products on e-commerce platforms could be extremely useful as customers could be easy to know the user experience from other guests. However, customers would spend plenty of time on reviews to select valid information. Inspecting reviews from various and extensive products may cause inefficiency. Additionally, there exists reviews with a long length and only a few sentences inside contain the valid information. A simple and effective summarization of reviews could help merchants to improve their service and customers to have a more enjoyable shopping experience.

In this paper, we aim to generate efficiently abstractive summarization of multi-products reviews from the Amazon by developing an encoder-decoder architecture. With the development of the deep neural network, standard Recurrent Neural Network (RNN) is capable of memorization by adding the recurrent connection on the feed forward neural network. However, the studies in 2013 from Bengio have revealed that standard RNN suffers severely from the problem of short-memory which is caused by gradient vanishing and exploding. The issue become very prominent, especially when standard RNN is used to learn very long dependencies.

In order to solve the problem of vanishing gradient occurred in standard RNN, Hochreiter and Schmidhuber in 1997 proposed the Long Short Term Memory Network (LSTM) which is the most successful technique to deal with the vanishing gradient problem. In contrast to the standard RNN memory cell, LSTM memory cell contains three gates (forget gates, input gates and output gates) which could remove, store and add information from the memory of the network. Compared to the standard RNN, this character of LSTM make it to be capable of long-term memory.

In order to solve the difficulties in training a deeper neural networks, He et al. in 2015 published a paper in which they introduced the residual learning framework to achieve the training of ultra-deep convolutional networks. The success of their work is in the field of computer version. In 2017, Cheng reformulated their work to make residual learning with attention fit in the field of

sequence learning. In this way, the recurrent neural network could learn the ultra-long range dependencies across time steps in sequence learning. Different to the residual learning in the field of computer version, Cheng combined the attention mechanism and residual learning in the context of sequence learning. Attention mechanism could make the encoder to encode information which is fully relevant to the task.

The following is a brief introduction of the combination of attention mechanism and residual learning by Cheng. Suppose that the recurrent model is denoted by $\mathcal{M}$ and the residual function is denote by $\mathcal{F}$. Then the whole recurrent neural network by adding the combination of residual with attention is as following:

$$\mathcal{M}\left(\mathbf{x}_t, \mathbf{h}_{t-1}\right) + \mathcal{F}\left(\mathbf{h}_{t-2}, \mathbf{h}_{t-3}, \ldots, \mathbf{h}_{t-K-1}; \mathbf{W}_a\right),$$

where $\mathbf{W}_a$ is the attention weights.

Word embeddings have been widely used in the task of deep learning based Natural Language Processing (NLP), since they have shown the effectiveness in summarizing context similarity and capturing semantic information (Young et al. 2018). In recent years, models which are used to create word embeddings closely resembles the neural networks. Word embedding is a method to create the word representation that makes attempt to assign the similar representation to words with similar meaning (Young et al. 2018). It is essentially for word embedding to follow a distributional hypothesis which describes that the words with similar meaning are more likely to appear in similar context. The algorithm of word embedding we use in our experiment is GloVe which is developed by Pennington et al. in 2014. We give the detail of GloVe in the below section of method.

In this paper, we utilize the LSTM as base Recurrent Neural Network to include the recurrent residual attention (RRA). The encoder constructs by the Bi-directional LSTM with RRA while the decoder uses the standard LSTM with RRA. Also local attention mechanism was involved in the decoder phrase.

2 Related Work

2.1 Models

2.1.1 Feed Forward Neural Network

The feed forward neural network is a traditional neural network (Schmidhuber, 2015). This neural network is the simplest artificial neural network, which contains three layers: an input layer, a hidden layer and an output layer. In some cases, there could be multiple hidden layers. In the hidden layers, each node is called neuron which is the basic unit of a neural network. In this network, the data is fed to the input layer, passes through the hidden layer and then ends in the output layer. In other words, there is only one direction for information to move in this network. However, the traditional neural network is not able to use previous information to effect the later ones. Thus, moving on now to RNN.

2.1.2 Standard Recurrent Neural Network

There have been several investigations into the standard recurrent Neural Network (RNN) (Rumelhart et al., 1986; Elman, 1990; Werbos, 1988; Burget, 2010). In contrast to the feed forward neural network, the standard recurrent neural network contains the looping mechanism which could pass the prior information forward. In other words, recurrent connections are added to the standard RNN. These recurrent connections are able to let the networks hidden unit to capture the previous information. Then the previous information could be used to make the prediction for the next hidden state output. Therefore, RNN is capable of memory.

The standard recurrent neural network also contains three layers: an input layer, a hidden layer and an output layer. In time t, the input to the network is $x(t)$, $h(t)$ is the hidden state of the network and output from the network is $o(t)$. The information is contained in the hidden state, which is the representation of previous input. We break up the sentence that need to be summarized into single word. Each time we feed one word to the standard RNN. Thus, the input vector $x(t)$ at time t is formed by concatenating the current word and the output $h(t-1)$ from the last hidden layer. For a standard RNN, the internal of the hidden units could be constructed as following:

$$h_t = \tanh\left(W_h\left[h_{t-1}, x_t\right] + b_h\right)$$
$$o_t = \sigma\left(W_o h_t + b_o\right),$$

where $\sigma()$ is the sigmoid activation function. The limitation of the standard RNN is that it suffers from short-term memory due to the problem of vanishing gradient (Bengio et al. 2013). Therefore, variants of standard RNN are designed to solve this problem.

2.1.3 Long Short Term Memory networks

In 1997, Hochreiter and Schmidhuber were the first to introduce the Long Short Term Memory networks. LSTM, is a variant of standard RNN, which is designed to overcome the problem of vanishing gradient. It could have the ability to learn the long-term dependencies.

In contrast to the networks of standard RNN, LSTM has the same form of a chain of repeating memory cells of neural network. However, compared to the memory cell of standard RNN, LSTM has more complicated memory cell. There are three gates in the memory cell of LSTM: a forgate gate, an input gate and an output gate. Suppose that in time t, x_t is the current input data, h_t is the hidden state and c_t is the cell (internal) state.

Turning now to explain the work process of LSTM memory cell.The forget gate is the first step of a LSTM memory cell. During it, we need to decide what information we want to throw away from the previous hidden state c_{t-1}. The output f_t is obtained from the forget gate layer. We multiply the previous cell state c_{t-1} by f_t, forgetting the things we decided to forget earlier.

The next step is that in the input gate layer we decide what new information we should store in the previous cell state c_{t-1}. There are two parts in this step. A tanh layer creates a vector of new candidate values and then a sigmoid layer decides which new candidate value we should update to the previous cell state c_{t-1}. The selected candidate values are then added to the previous cell state c_{t-1} to form the current cell state c_t.

Output gate is the last step in a LSTM memory cell. During it, the previous hidden state h_{t-1} and the current input x_t are sent to a sigmoid function. This sigmoid layer could decide that what parts of the new cell state c_t we should output. At the same time, we put the new cell state c_t through a tanh layer and then multiply it by the output of the sigmoid layer, so that we get the new hidden state h_t.

2.2 Residual Learning

In 2015, He et al. published a paper in which they first pointed to the challenge (the degradation problem) from training a deeper neural network. Problems of vanishing and exploding gradients are notorious obstacle for networks in the way of convergence from beginning. These two problems could be addressed by normalized initialization and intermediate normalization layers. However, a degradation problem comes after solving gradient vanishing and exploding. According to introduction from He et al., with the increasing of the network depth, degradation problems could lead to the saturation of accuracy and then the rapid degradation. This problem is not related to overfitting, thus dropout cannot salvage it. In order to address degradation problems, He et al. proposed the deep residual learning framework to ease the training of deeper networks. In other words, they add residual connections to networks. The term residual connection refers to connecting the output of previous layers to the output of new layers. The deep neural network with adding residual connection across layers has significantly improve the performance in computer version (He et al. 2015). In 2017, Cheng inspired by He et al., he learn long and complex dependencies from sequential data by incorporating residual connection across multiple steps. Besides the use of residual learning, Cheng fitted the case of sequence learning by casting attention mechanism to recurrent residual connection. This method could also learn the semantic meaning from sequential data by considering the information that are several steps apart. The following part is a brief description of related work on attention mechanism.

2.3 Attention Mechanism

Neural networks with attention have achieved considerable success in several domains, such as machine translation (Bahdanau, Cho, and Bengio 2014). In 2014, Bahdanau et al. published a paper in which they described a translation neural model using encoder-decoder architecture and introduced an attention mechanism.

Attention is used to solve the following problem in the traditional encoder-decoder architecture (Young et al. 2018). When the encoder encode information, it compress all information into a fixed-length vector. Thus, the traditional encoder-decoder could not capture the most relevant information of the input sentence, especially when the input sentence is very long or information-rich. The neural network with attention is capable of paying attention to relevant part of input when the input is processed in the model. In other words, attention in neural networks is designed to assign weights to different information in the input sentence instead of compressing all input sequences

equally.

The structure of attentive model is following. There are n arguments $y_1, , y_n$ in the attentive model and a context information c. The model returns a vector z which contains some weights. The vector z is the summary of $y_i, i = 1, ...n$, based on how y_i is related to context c. In other words, weights in vector z demonstrates the relevance between each y_i and c. The sum of weights in vector z should be 1.

2.4 summary

Cheng create the Recurrent residual attention (RRA) by reformulating an attention over residual connection in recurrent network.

Cheng used standard LSTM as his baseline and LSTM with RRA as his model. His model outperforms the standard LSTM network on three benchmark tasks: the adding problem, pixel-by-pixel and MINST. Inspired by Chengs work, we will apply the bi-directional LSTM with RRA on the Amazon reviews so that we could get an efficiently abstractive summarization.

3 Method

3.1 Word embedding

The famous word embedding algorithm–GloVe was developed by Pennington et al. in 2014. This is also the algorithm of word embedding we use in our experiment. The reason for us to use GloVe algorithm is that the in order to get the efficient summarization of reviews we expect the embedding algorithm could capture semantic and syntactic relationships between words. GloVe could achieve this point.

The section below describes the main points of GloVe in detail (Pennington et al, 2014). The aim of Glove algorithm is to learn the word vector w. To see how the algorithm achieve this aim, we introduce the following contents.

Word-word Co-occurrence Matrix: GloVe firstly construct a clear word-word co-occurrence matrix using statistics across the global text corpus. This matrix shows how frequently words co-occur with one another in a given corpus. Before looking at the detail of the construction of word-word co-occurrence matrix (Pennington et al, 2014), let introduce some notations. Let X denote the word-word co-occurrence counts matrix. The entries of this matrix are denoted by X_{ij} which tabulate the number of times word j appears in the context of word i. Then the number of times any word appears in the context of word i is calculated as $\sum_k X_{ik}$

Co-occurrence Probabilities Matrix and ratio of these probabilities: The probability that word j appear in the context of word i is calculated as following.

$$P_{ij} = P(j|i) = \frac{X_{ij}}{X_i}$$

To get the co-occurrence probabilities matrix, we combine these probability to a matrix. i and j are words that we are interested in. Let denote the various probe words by k. From the co-occurrence probabilities matrix, we could calculate the ratio of co-occurrence probabilities with various probe words,k, as P_{ik}/P_{jk} The study of ratio of co-occurrence probabilities with various probe words k could examine the relationship between word i and j.

The above construction is related to the derivative of GloVe.

Soft constraints for each word pair:

$$w_i^T \tilde{w}_k + b_i + \tilde{b}_k = \log(X_{ik})$$

w_i is the word vector for the main word i, $\tilde{w}_k$ is the separate context word vector for the various probe words, k, b_i is the bias for word i and $\tilde{b}_k$ is the bias for words k. The above equation is in the ideal situation. Now turning to the cost function in real situation.

Cost function: To be realistic, we expect the right side of soft constraints approaches the left side of it. Then by introducing a weight $f(X_{ik})$, we get the cost function as following:

$$J = \sum_{i,k=1}^{V} f(X_{ik}) \left(w_i^T \tilde{w}_k + b_i + \tilde{b}_k - \log X_{ik} \right)^2,$$

V is the size of the vocabulary.

The weighting function $f(x)$ could help us to prevent learning from extremely common pairs. Pennington et al. proposed a weighting function as following.

$$f(x) = \begin{cases} (x/x_{\max})^\alpha & \text{if } x < x_{\max} \\ 1 & \text{otherwise} \end{cases}$$

3.2 Model

In this experiment, the sequence to sequence model would be studied to draw abstractive summarization on the amazon review. The model contains two parts, an encoder and a decoder. Because of the problem of vanishing gradients, the

standard RNN suffers from the short-memory. In other words, the standard RNN does not learn the long-range dependencies across time steps. Thus, LSTM is designed to mitigate the problem of short-memory. In contrast to the standard RNN, LSTM are capable of learning long-term dependencies by using gates. Furthermore, In 2017, Cheng published a paper in which he introduced a LSTM with recurrent residual attention (RRA). This model solves the problem of short-memory by casting the attention mechanism to residual connection over timesteps in recurrent network. Cheng has proved that this model could learn ultra-long range dependencies across time steps in sequence learning as compared to standard LSTM. We utilize the LSTM as base Recurrent Neural Network to include the recurrent residual attention (RRA). The encoder constructs by the Bi-LSTM with RRA while the decoder uses LSTM with RRA. Also local attention mechanism was involved in the decoder phrase. The whole model is shown in the Figure 1.

3.2.1 Bidirectional LSTM

Firstly words are transferred into its vector representations. If a word cannot be found in the vocabulary, it would be regarded as an unknown term (unk) and the vector of unk would be called instead.Then the vectors become the input of the encode layers, the Bi-LSTM with RRA.

For the Bidirectional LSTM, it means the neurons of LSTM are split into two directions, forward and backward direction. The output are generated after hidden states, and when the operation in two directions finished, output would be concatenate together. The words in sentences were transferred into LSTM one by one. LSTMs are composed of a cell which denotes memory of LSTM unit and three gates: an input gate, an output gate and a forget gate. The formula for LSTM can be written as:

$$f_t = \sigma_g\left(W_f x_t + U_f h_{t-1} + b_f\right)$$
$$i_t = \sigma_g\left(W_i x_t + U_i h_{t-1} + b_i\right)$$
$$o_t = \sigma_g\left(W_o x_t + U_o h_{t-1} + b_o\right)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c\left(W_c x_t + U_c h_{t-1} + b_c\right)$$
$$h_t = o_t \circ \sigma_h\left(c_t\right)$$

where f is the forget gate, o is the output gate and i is the information gate. The calculation of hidden states would be different as recurrent residual attention is introduced into this model.

3.2.2 Recurrent Residual Attention (RRA)

When calculating the hidden states, recurrent residual attention is used. RRA is composed by two parts: residual connections and attention mechanism.

For every hidden states, the model would compute the weighted sum of the previous K hidden states where weighted based on some trainable parameters in the attention weight matrix, the weighted sum is denoted as RRA. And RRA would be added into the model when calculating the hidden states as:

$$h_t = o_t \circ \sigma_h\left(c_t + RRA\right)$$

Using this is to record connections among hidden states, then to establish long term dependencies.

3.2.3 Attention

Considered that the global attention has deficiency that it needs to attend to all words, which is expensive and even impractical when dealing with long sequences, a local attentional mechanism was applied, which focuses on a small subset of the source positions. Figure 2 shows the structure of attention mechanism.

Firstly, for each target word an aligned position p_t was calculated by this model at time t. Then, c_t, the context vector (the further use of the context vector would be discussed later), is generated using a weighted average over encode hidden states in the window [ptD, pt+D], while D is empirically selected. If the window is beyond the boundaries of sentences, outside parts would be excluded out. Here the predictive alignment was used to calculate p_t, here h_t denotes the former decode state.

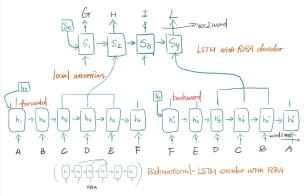
$$p_t = S \cdot \operatorname{sigmoid}\left(\boldsymbol{v}_p^{\top} \tanh\left(\boldsymbol{W_p} \boldsymbol{h}_t\right)\right)$$

Here, $\boldsymbol{W}_p$ and $\boldsymbol{v}_p$ are parameters that the model would learn. S is the encode sentence length. According to the properties of sigmoid, $p_t \in [0, S]$, a gaussian distribution centered at p_t is generated to calculate the alignment weights, i.e. using a truncated Gaussian to adjust alignment weights:

$$\boldsymbol{a}_t(s) = \operatorname{align}\left(\boldsymbol{h}_t, \overline{\boldsymbol{h}}_s\right) \exp\left(-\frac{(s-p_t)^2}{2\sigma^2}\right),$$

where $\overline{h}_s$ is the encoder hidden states in the window,and align function can be defined as:

Figure 1: Encoder-decoder architecture



$$\text{align}\left(\boldsymbol{h}_t, \overline{\boldsymbol{h}}_s\right) = \frac{\exp\left(\text{score}\left(\boldsymbol{h}_t, \overline{\boldsymbol{h}}_s\right)\right)}{\sum_{s'} \exp\left(\text{score}\left(\boldsymbol{h}_t, \overline{\boldsymbol{h}}_{s'}\right)\right)},$$

where the score function used here is a dot product, and standard deviation set is $\sigma = \frac{D}{2}$

Therefore, $\boldsymbol{h}_t$ and the context vector $\boldsymbol{c}_t$, has been obtained, a concatenation to combine information to generate an attentional hidden state is as follows: $\tilde{\boldsymbol{h}}_t = \tanh\left(\boldsymbol{W}_{\boldsymbol{c}}\left[\boldsymbol{c}_t; \boldsymbol{h}_t\right]\right)$

Then the predictive distribution can be calculated as: $p\left(y_t | y_{<t}, x\right) = \text{softmax}\left(\boldsymbol{W}_s \tilde{\boldsymbol{h}}_t\right)$

Here, $\boldsymbol{W}_s$ is a parameter that the model would learn, the size of it would become $2hidden_size \times vocab_length$, the most probable vocabulary would be chosen and output. All sentences would terminate with a end-of-sentence token.

In the experiment, first encoded hidden state is used as the initial decoder state. The decoder part use the LSTM with RRA, similarly as encoder part.

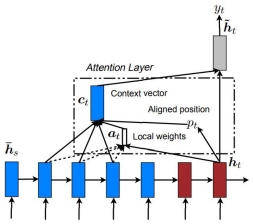
3.2.4 Vec2word

After obtaining output, the nearest neighbour method was used to convert vectors to words. The word vector in the dictionary which has highest similarity to word vector given was returned. This similarity is calculated by the cosine similarity. Similar to the word to vector stage, if the vector representation is unknown, and no corresponding word is known, then it returns the word representation of a known vector representation which is most similar to the vector given as argument.

4 Results Discussion

To evaluate the results, automatic metrics and human evaluation could be used. For automatic systems, ROUGE-N and BLEU were wildly used to evaluate Natural Language Generation systems. And for human evaluation, we choose both visual inspection of output and human scores based on several rules. After generating results, it is found

Figure 2: Blue blocks denote the encoder layers, $\overline{\boldsymbol{h}}_{\boldsymbol{S}}$ denotes the encoder outputs, red blocks denote the decoder LSTM and the gray block denotes the next cell following the second red one. $\tilde{\boldsymbol{h}}_t$ would become h_{t+1} then transfer to the next cell.



that using ROUGH-N and BLUE is not appropriate. ROUGE measures similarity between test summary and label summary. But this cannot represent the quality of the summarization. Thus we mainly used human evaluation methods to measure the quality of summaries generated.

In the following part, we would visualize the predicted summary and collect human rankings. Through visually inspecting the results, we found a interesting phenomenon. For example, These predicted summaries might contain repeated phrase. Another one is the length of these predicted summaries are too short to contain all the useful information. In order to do the qualitative analysis, we picked three predicted summaries which are paired with the original summary written by the customers. Three qualitative examples are listed as following:

Then we let three people analyze the predicted summary and give a ranking for the following criterions. The first criterion is words based ranking, whether the model could capture the right sentiment and whether the model could determine the keyward of the text which is semantic similarity. The second criterion is grammar based ranking, readability and grammaticality. Readability is the difficulty that a reader understands a text and grammaticality measure the grammatical correctness of the predicted summaries. The last criterion is informativeness and narutalness. Informativeness is the ability of the model to provide all the useful information from the source text and naturalness is whether the generated summaries have a similar expression as produced by a native speaker.

4.1 Qualitative analysis: result visualisations

4.1.1 Good result exhibition

Review 1: better than other brands or supermarkets imitations. it was worth the wait because they are unk great tasting cookies.
Predicted Summary: yummy cookies
Actual Summary: wonderful cookies
Reason: This model caught the right keyword cookies from the text and the right sentiment where it summarised great tasting to yummy. The readability is high and the grammar is correct. It abstract all the useful information and prase generated by the model just like native speakers would use.
Review 2: these are the best chocolate fudge cookies commercially available. and the price is way below what you at the grocers.
Predicted Summary: inexpensive favorite
Actual Summary: chocolate fudge
Reason: Even if the loss of this sentence is quite high, due to the inappropriate reference summary. But the quality of the prediction is quite high as it sums up two major key words in the sentences: inexpensive and favorite. This reminds us in the future work it is necessary to drop out some noise example in the data preprocessing phase, which needs a lot work due to it can only be operated manually.

4.1.2 Unsatisfied result exhibition

Review 1: green mountain unk are consistently a great value. breakfast blend is n't too strong or too weak, just right for my whole family.
Problem: repeated phrase
Predicted Summary: great set set set
Actual Summary:great unk and price
Reason: If the decoder produces the same word or phrase repeatedly then it could mean that the context vectors fed to the decoder at these time steps are very similar. And the decoder has over-reliance on the decoder input. Since we do not require the current context vector to be orthogonal to all previous context vectors, the model tend to generate words repeatedly.
Review 2: the coffee is good, however the unk is not packaged well, the coffee machine barely streams coffee, more of a fast drip. the whole idea is it 's fast, right?
Problem: missing information
Predicted summary: best flavor coffee live
Actual summary: good coffee bad packaging
Reason: Some useful information is missing in this example. This model understand the coffee is good but it did not extract the information of bad packaging which might caused by unk. When we map words to vectors, we used limited vocabularies, it only contains the common vocabularies which could dramatically reduce the number of parameters. This is a trade-off between time limitation and model accuracy. If the we expand the training set, increase the number of vocabularies in lexicon and train through large number of epochs, the phenomenon of missing information would be alleviated.
Review 3: I bought this TV during the bank holiday sale, and all I can say is it was a bargain! The TV is easy to set up, and comes with a variety of apps. I mainly use the TV to watch movies, TV shows and for gaming (unk), which all look incredible. I'd highly recommend this TV!
Problem: grammar issue
Predicted summary: TV shows easy to recommend TV apps
Actual summary: A great TV for a great price.
Reason: Grammaticality of this model is not guaranteed which might caused by the adding position of residual connection. We could see that in the review, several verbs follow in the main thing–TV. In addition, the semantic dependencies between words that are far apart. The residual connection with attention was added in the wrong place over timesteps in the recurrent networks.

4.1.3 Quantitative analysis: Human Rankings

We select 200 sentences to give scores between 0 to 10 according to the aspects inferred before. They are semantic similarity, readability, grammaticality, informativeness, naturalness, quality respectively. The table of average scores was exhibited below:

semantic similarity	readability	grammaticality
5.7	4.0	1.3
informativeness	naturalness	quality
4.8	6.2	5.3

This results suggested that the grammar of predictions is quite bad. The reason has been mentioned before.

5 Experiment

We obtained a dataset consisting of reviews on Amazon from SNAP. In the raw dataset, there are 34,686,770 reviews for 2,441,053 products which

were written by 6,643,669 customers.
We choose among the 24 topic from the dataset and use Electronics, CDs and Home and Kitchen to train the model. We randomly sampled 20,000 pairs of text and summary from each topic as our training set. We preprocessed the dataset by restricting the length of texts to the range from $2 \times D + 1$ up to 80 and limiting the length of summaries up to 8 which is also a trade-off between generalization and time limitation. After preprocessing, the final training set contained 16,076 samples of text and summaries. In the testing part, we added one more topic of reviews from Toys and Games because we expect that our model could generalise to not seen topics of reviews summarization.
We implemented this model on Google Colaboratory using Tensorflow. For GloVe, the smallest one with words represented by vectors of dim 50 (glove.6B.50d.txt) was picked. During the experiment, limited vocabularies were used instead of normal size of vocabularies, which only includes words appears in the dataset and several special words representing markers 'eos', 'sos'," etc. The reason for doing this is that the network would output the probability distribution over words based on the vocabularies. Thus a limited edition would require less parameters when calculating probabilities. Then some samples, considered inappropriate, were removed from dataset to avoid crashes. And the local attention mechanism was involved in this model, local attention would select hidden states in the encoder in the range from $P_t - D$ to $P_t + D$. We introduced before that D is a empirical choice and p_t is a position determined. The range of $P_t - D$ to $P_t + D$ is the window and P_t is the center. As D was set as hyperparameter 10, the size of window should be $2D - 1$ and was required to be smaller or equal to the number of hidden states. Thus we removed input texts that length is less than $2D + 1$. Considered the length of one text would become the number of timesteps for the LSTM. To have a less intensive training, we would remove all data with size beyond a given threshold. Besides, those samples with summaries that are too long would be picked out as the training could be processed better if target outputs are converted into a fixed length.
The hyperparameters we set is that hidden size is 500 learning rate is 0.003. Training iterations (epoch) is 5. In RRA, there is K = 5 previous hidden states to be added to the current state. For attention, D is set as 10. Thus windows size for local attention will be $2 \times D + 1 = 21$.
In this discrete classification task, in order to examine the probability error, we used cross entropy as our loss function Specifically, it is sparse softmax cross entropy with logits. For optimization, we used Adam optimizer, which computes adaptive learning rates for every parameter. The normal initializer and zero initializer were used to initialize parameters required to train in this model.

6 Conclusion and Future Work

Considered exploring reviews in internet, in this paper we tried to develop a sequence to sequence encoder-decoder architecture model to draw abstractive summarization from reviews. The encoder part used a bi-directional LSTM with RRA and the decoder part used a LSTM with RRA. The local attention mechanism was applied between the hidden states of encoder and decoder input. The prepossessing using GloVe generated vocabularies for this model. To check the quality of predicted reviews, we used human evaluation, both visual inspection and human rankings, to measure whether there are valid summaries produced in this task.

Some predictions are appropriate, as they have close meaning with reference summaries, and caught the right keyword with satisfied readability and correct grammar, which means our model can predict some reviews. However, some results have drawbacks, e.g. repeated phrase, missing information and grammar issue.

After the analysis of reasons leading to this deficiency, there are several improvement we could do in the future. Firstly, if there is a more advanced machine that can avoid the crash problem and short the computation time, we could use whole vocabularies rather than the limit version. And more epoch could be applied and more training data could be put in this model. Moreover, observing that the reference summaries itself have some noise, a more sophisticated data prepossessing should be implemented in order to exclude those labels that cannot have a valid summary of reviews itself. Moreover, different hyperparameters could be explored to see whether there exist a more suitable choice. And also, an advanced model could be investigated to address these problems.

References

Wang C. RRA: Recurrent residual attention for sequence learning[J]. *arXiv preprint arXiv:1709.03714, 2017.*

Singhal S, Bhattacharya A. *Abstractive Text Summarization[J].*

Luhn, H. P. 1958. The automatic creation of literature abstracts. *IBM Journal of Research and Development, vol. 2, no. 2.*

Turney P D. Learning to extract keyphrases from text[J]. *arXiv preprint cs/0212013, 2002.*

Frank E, Paynter G W, Witten I H, et al. Domain-specific keyphrase extraction[C]//16th International joint conference on artificial intelligence (IJCAI 99). *Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999, 2: 668-673.*

Mihalcea R, Tarau P. Textrank: Bringing order into text[C]//*Proceedings of the 2004 conference on empirical methods in natural language processing. 2004.*

Schmidhuber J. Deep learning in neural networks: An overview[J]. *Neural networks, 2015, 61: 85-117.*

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

Rumelhart D E, Hinton G E, Williams R J. Learning representations by back-propagating errors[J]. *Cognitive modeling, 1988, 5(3): 1.*

Werbos P J. Generalization of backpropagation with application to a recurrent gas market model[J]. *Neural networks, 1988, 1(4): 339-356.*

Mikolov T, Karafit M, Burget L, et al. Recurrent neural network based language model[C] *Eleventh annual conference of the international speech communication association. 2010.*

Elman J L. Finding structure in time[J]. *Cognitive science, 1990, 14(2): 179-211.*

Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks[C] *International conference on machine learning. 2013: 1310-1318.*

Hochreiter S, Schmidhuber J. Long short-term memory[J]. *Neural computation, 1997, 9(8): 1735-1780.*

Young T, Hazarika D, Poria S, et al. Recent trends in deep learning based natural language processing[J]. *ieee Computational intelligenCe magazine, 2018, 13(3): 55-75.*

Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. *arXiv preprint arXiv:1409.1556, 2014.*

He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C] *Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.*

Glorot X, Bordes A, Bengio Y. Domain adaptation for large-scale sentiment classification: A deep learning approach[C] *Proceedings of the 28th international conference on machine learning (ICML-11). 2011: 513-520.*

Hermann K M, Blunsom P. The role of syntax in vector space models of compositional semantics[C]. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2013, 1: 894-904.*

Pennington J, Socher R, Manning C. Glove: Global vectors for word representation[C].*Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014: 1532-1543.*

Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality[C]. *Advances in neural information processing systems. 2013: 3111-3119.*