Text Ads Generation Using Deep Neural Network

Stanford CS224N Custom Project

Weijian Ma
Department of Computer Science
Stanford University
Bing Ads @ Microsoft
krisma@stanford.edu

Abstract

Automatic text summarization has been a well-researched NLP topic in recent years. It is possible to build machine learning models that are capable of distilling crucial information from a larger piece of text and condensing it to a smaller one. Text summarization using deep neural networks has become an effective approach and there are many use cases for that technique. One possible use case is text ads generation in online search advertising. Many advertisers are publishing text ads with ad titles that are not effective. Less effective ad titles will result in a lower chance of user conversion (clicks), which is harmful to the advertisers, and also a waste of hosting resources to the ads marketplace. It is meaningful to build a model that could rewrite the adtitle by summarizing the ad content. In this paper, we will study and leverage several state-of-the-art text summarization models, compare their performance and limitations, and finally, propose our own solution that could outperform the existing ones.

1 Key Information to include

- Mentor: Rui Yan
- External Collaborators (if you have any): N/A
- Sharing project: N/A

2 Introduction

Sponsored advertising is an industry in which the most advanced predictive modeling techniques are applied because improvement in predictive capability can directly increase revenue. Using state-of-the-art machine learning techniques, the advertisers are able to attract more users and better advertise their brand. It is also beneficial to ads marketplaces like Microsoft and Google because it will help them conserve computing power and generate more revenue.

As a suggested application area of text summarization, text ads generation is not a trivial problem. The challenges we are facing can be summarized into three categories. The first one is lack of pre-training context. In recent years, many text summarization systems include encoders pre-trained on massive text corpus with self-supervised objectives. However, the text corpus in the publicly available pre-training data, like CNN/Daily Mail, is very different from text ads in both vocabulary and syntax, and thus a pre-trained model does not work well in generating high quality ad titles.

The second challenge is that there is no such thing as "golden-standard" text ads in the marketplace. In general text summarization tasks, we can rely on human-written summaries, which have important features like accuracy, fluency, and relevance. In ads marketplaces, there is not a quantitative way to measure the quality of an ad title. Our task to improve the ad title quality by summarizing ad description could be difficult if the data used to train the model are not high-quality. Lack of reliable training data is always a problem to supervised learning.

The third challenge is that an ad title generated by machine can contain misleading information if the decoder is free to search for words that are not in the original ad descriptions. For example, "free shipping" is a common term in ad titles to attract buyers, and the model trained on massive ad corpus can generate "free shipping" in ad titles while in reality the advertiser is not willing or able to do that.

In general, a text ad includes an ad title and an ad description. In the following example, the blue font at the beginning is "ad title", and the grey font below the link is "ad description".

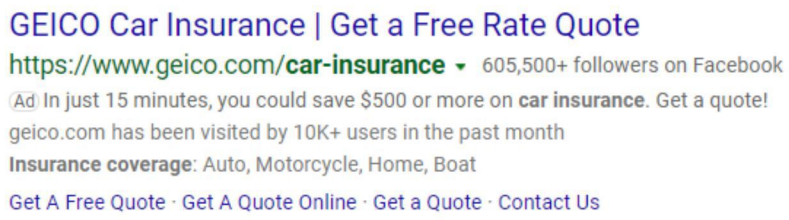

Figure 1: A simple text ad consists of ad title "GEICO Car Insurance ..." and ad description "In just 15 minutes, you could save ..."

Our task is to build a deep learning model which takes in ad descriptions and generate ad titles correspondingly. Trained on massive ads data from Bing Ads, a RNN-based encoder-decoder model could make fairly good summarization of the ad descriptions.

3 Related Work

There are many techniques published in recent years that achieved success in text summarization tasks. In general, there are two categories: extractive and abstractive. Extractive methods choose tokens or sentences from the source and directly copy to the target [1]. Depending on the design of such systems, they could also do a ranking or rearranging of the selected tokens to make sure that the target text is grammatically correct. On the other hand, abstractive models [2] consume source texts and generate target texts. They are capable of generating words that are not in the source texts. In this paper, we will focus on the recent research on sequence-to-sequence (seq2seq) models for abstractive text summarization.

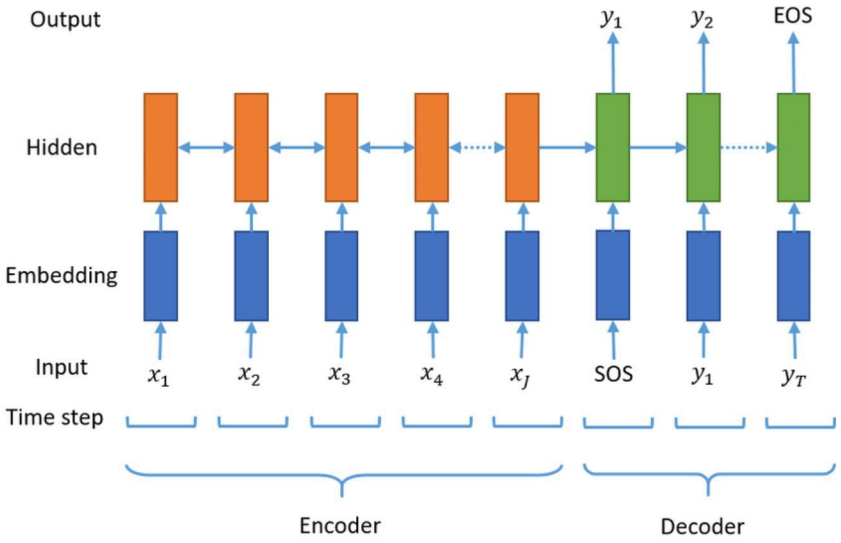

Figure 2: Sequence-to-sequence model structure (Seq2seq)

The attention mechanism has been widely used in natural language processing (NLP) [3] tasks like machine translation and image caption. Attention takes in two sentences and computes the correlation between each token in the two sentences. In case the two sentences are the same one, it is called "self-attention". The key of Attention mechanism is the vector of alignment scores. For each output hidden state to be generated in the decoding stage, we calculate the alignment scores between the previous output hidden state and every input hidden state to locate the tokens that the model should focus on. Eventually we generate the context vector by multiplying the encoder hidden states and their respective alignment scores. Then we concatenate the context vector with the previous decoder output, and we feed the result, together with the decoder hidden state, to decoder RNN to generate the next decoder output.

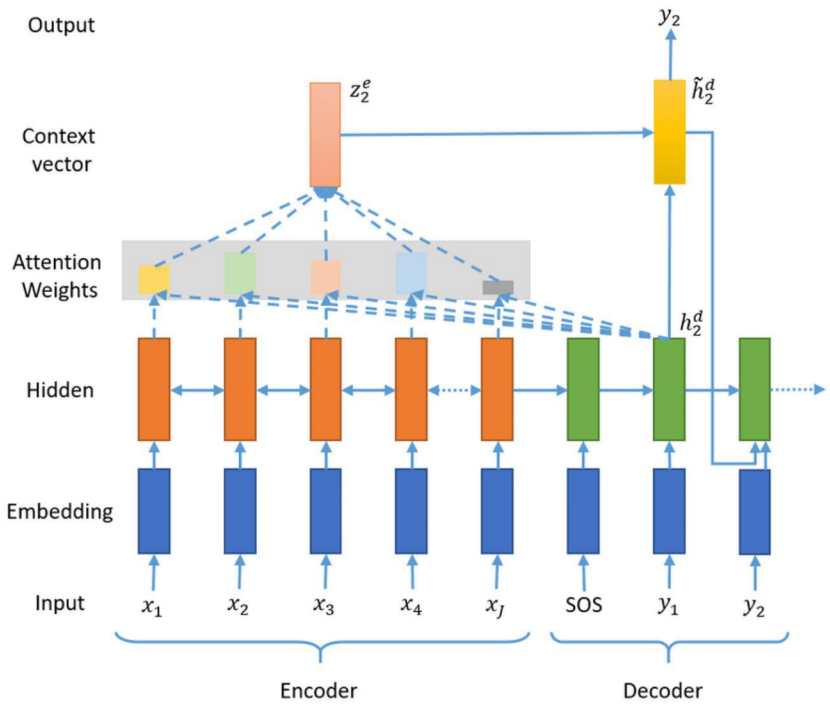

Figure 3: Sequence-to-sequence model with Attention

4 Approach

4.1 Neural Machine Translation with RNN

Although machine translation is a different topic from text summarization, they both can be solved using sequence-to-sequence models. A well-designed NMT model could be used to generate summarized text with minor changes. We start with studying the attention based encoder of Bahdanau et al. (2014)[4]. Bahdanau introduced an RNN-based encoder-decoder neural machine translation (NMT) system which allows the model to automatically search for relevant parts in the source sentence when generating each part of the target sentence. A basic encoder-decoder system consists of an encoder, which reads the input sentence, and a decoder, which generates the output sentence. A simple RNN-based encoder takes in sentence x:

$$h_t = f(x_t, h_{t-1}) \tag{1}$$

and outputs vector c:

$$c = q(h_1, \ldots, h_{T_x}) \tag{2}$$

where $h_t \in \mathbb{R}^n$ represents the hidden state at time t and c represents the output vector of the encoder.

A RNN-based decoder is trained to predict the probability distribution of y_t at each time t given all previously predicted token $\{y_1, \ldots, y_{t-1}\}$ and c:

$$p(y) = \prod_{t=1}^{T} p(y_t|\{y_1 \ldots, y_{t-1}\}, c), \tag{3}$$

and in a basic RNN setting, we use some non-linear functions g to output y_t:

$$p(y_t|\{y_1, \ldots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c), \tag{4}$$

where s_t is the hidden state at time t from RNN. In the next step we will investigate an advanced approach based on this model. It includes LSTM to serve as the g function in (4) and applies a global attention mechanism at each decoding step.

4.2 Abstractive Text Summarization with Attention

We study the attention-based NMT system[5], modify the published code implemented by Pencheng Yin [6] to make it compatible with text summarization tasks, and train the model with ad description as source sentence and ad title as target sentence.

As the baseline of our research, a vanilla attention-base NMT is directly used as text summarizer. The technique and code are from [6] with minor change to conform our company infrastructure. The training and testing data are from Bing Ads marketplace and randomly sampled. This approach introduces a global attention mechanism:

$$a_t(s) = \text{align}(h_t, \overline{h}_s) = \frac{\exp(\text{score}(h_t, \overline{h}_s))}{\sum_{s'} \exp(\text{score}(h_t, \overline{h}_{s'}))}, \tag{5}$$

where h_t is the current hidden state for target sentence and $\overline{h}_{s'}$ is each of the hidden state for source sentence. The author[5] lists three implementation for the score function, which is considered as a "content-based" function. In our approach, we used the "general" one:

$$\text{score}(h_t, \overline{h}_s) = {h_t}^T W_a \overline{h}_s \tag{6}$$

Eventually the hidden state h_t is used to compute the attention vector $\hat{h}_t$:

$$\tilde{h}_t = \tanh(W_c[c_t; h_t]) \tag{7}$$

and then the y_t is calculated using

$$p(y_t|y < t, x) = \text{softmax}(W_s \tilde{h}_t) \tag{8}$$

Our objective is to maximize the log-likelihood of observed sequences,

$$logP_\theta(\hat{y}|x) = \sum_{t=1}^{T} logP_\theta(\hat{y}|\hat{y}_{<t}, x), \tag{9}$$

which is equivalent to minimizing the cross entropy (XENT) loss, $loss_{XENT} = -logP_\theta(\hat{y}|x)$.

In this approach, we notice that the ad titles generated are not good enough. For example, an ad piece from an insurance company is:

Ad Description	Save up to 78% on Auto Insurance Quotes. Bundle & Save with rates from $16.08!
Ad Title (Ground truth)	Senior Car Insurance(Quotes) - Cheap Senior Auto Insurance
Ad Title (Pred)	Auto Auto Insurance - Save Up to Quotes

and the predicted ad title has duplicate terms "Auto" and the second part "Save Up to Quotes" make little sense.

4.3 The pointer-generator network

The pointing/copying mechanism [7] proposed by Vinyals et al. provides a new solution to text summarization. In text ads, ad titles tend to include the same words from ad descriptions. The pointing mechanism generates output tokens by first calculating the attention weights of input tokens and then copy the important ones directly to the output. One popular alternative of this mechanism is Pointer Softmax [8]. At the decoding step t, a short-list softmax $P_{vocab,t}$ calculated by Equation (8) is used to predict output tokens in the vocabulary. The location softmax gives locations of tokens that will be copied directly from the source sentence x to the target y_t based on the attention weight a_e^t.

$$p_{gen,t} = \sigma(W_{s,z}Z_t^e + W_{s,h}h_t^d + b_s), \tag{10}$$

As we can see in equation (9), the probability of copying a token from vocab at time t, P_gen, t is calculated using the context vector z_t^e and the hidden state h_t^d. The final probability of producing the target token y_t is calculated by the concatenation of vectors $P_{gen,t}P_{vocab,t}$ and $(1 - P_{gen,t})a_e^t$.

5 Experiments

5.1 Data

The dataset is a sampled subset of Bing Text Ads Corpus. It is a collection of all text ads published on Bing Ads marketplace. There are ads in different languages targeting different markets, but in order for our model to perform better, we only sample ads in English. In theory, the mechanism should work in general as long as the training set and the test set are in the same language. In Bing Ads marketplace, ad title is limited to 30 characters while ad description is limited to 90 characters.

In the first stage we sample 50 thousand text ads. Afterward, we trained the model on a much larger dataset which consists of 200 thousand text ads. We can see that increasing the size of the training dataset results in better performance in the holdout.

After training and validating with Bing Ads data, we also crawl publicly available Google Ads data. All example ads in this paper are crawled from google.com and because Google Ads is in the same format as Bing Ads (ad title and ad description), our model is capable of inferencing high quality ad titles. Since many advertisers publish to both marketplaces (Microsoft and Google), we remove the sample if it appears in training data to avoid leaking information.

5.2 Evaluation method

We measure perplexity as a sanity check for our experiments. To compare the performance of different approaches we use common evaluation metrics like ROUGE score and BLEU score.

5.3 Experimental details

5.3.1 Abstractive Text Summarization with Attention

In the baseline model we sampled 50 thousand English text ads for training and 5 thousand for testing. The ads are randomly sampled so even tail ads (the ads rarely shown to users) could be included. We use the vanilla summarizer implemented based on Yin's code[6] and train the model on a V100 GPU with lr = 5e-5 and it takes 30 minutes to train 9 thousand iterations. We record the performance in row 1 in the results table below.

5.3.2 The pointer-generator network

In the next step we use Shi et el.'s implementation of RNN-based encoder-decoder model plus Pointer network [9] to better summarize the ads. We modify the pipeline to accommodate the Bing Ads dataset and our GPU infrastructure. The model is trained with 50 thousand text ads on a V100 GPU with lr = 1e-4 and it took 2 hours to finish the training. We can see that with the help of pointer generation, we successfully copy the crucial token directly to the output ad titles, and improve our model in both precision and recall. The performance is recorded in row 2.

5.4 Results

The baseline experiment with randomly sampled ads shows unsatisfactory results, as we can see in the first row. The BLEU is very low, which means that the generated ad title is not good enough. We will work further to improve the system performance from both model and dataset perspectives.

Results			
System	Ppl	BLEU	ROUGE1
Baseline: LSTM + Attention	8.18	1.44	6.34
Baseline + Pointer Network - (1)	7.55	1.92	10.63
(1) trained on massive data	7.43	2.12	11.87

6 Analysis

One big problem in generating the ad title is that the real ad title may contain information not shown in ad description. For example, the title may contain the brand name like "Nike" and in description it will not repeat itself. In this way, it is very hard for our model to figure out the brand is "Nike" based on the ad description unless the model has huge amount of parameters and simply memorizes the ad content.

Ad Description	Earth's biggest selection of books, electronics, apparel & more at low prices.
Ad Title (Ground truth)	Amazon.comA(r) Official Site - Fast Free Delivery with Prime

In the example above, only human beings with extensive knowledge can realize that the "biggest selection of books" refers to amazon.com and our model is not capable of doing that. Also, the "Fast Free Delivery with Prime" is novel information that does not appear in the ad description.

We learn from the failed inference that in case the ad title has many words in common with the ad description, we better use extractive text summarization, or Pointer network which seeks to directly copy words from source to target. In case the ad title does not share the same words, we can benefit from generating words that are not presented in the ad description, which is what abstractive text summarization can better handle.

7 Conclusion

From the experiments we can see that even though we attempt multiple approaches to improve the model performance, simply condensing information from an ad description usually can not provide an ad title of better quality than the original one. One reason could be that there is some valuable information we are missing. For example, we can crawl the landing page (the web page loaded when the user clicks the ad) and parse the HTML components for the model to train on. In this way, the model is capable of better understanding what product or service the advertiser is providing.

Another reason that text summarization fails to generate good ad titles is because we do not have a pre-training process, so the model lacks an overall understanding of text ads. It is possible for us to pre-train on the all text ads so the model knows what an ad title usually looks like. There are limitations on pre-training the summarization model on more general data sources like wikipedia and New York Times because the language and writing style of such corpses are quite different from that of sponsored ads. The model could not learn much from them.

References

[1] Mehdi Allahyari, Seyed Amin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut. Text summarization techniques: A brief survey. *CoRR*, abs/1707.02268, 2017.

[2] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *CoRR*, abs/1509.00685, 2015.

[3] Dichao Hu. An introductory survey on attention mechanisms in NLP problems. *CoRR*, abs/1811.05544, 2018.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

[5] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. *CoRR*, abs/1508.04025, 2015.

[6] PengCheng Yin. A basic pytorch implementation of attentional neural machine translation. In *GitHub*. `https://github.com/pcyin/pytorch_nmt`, 2017.

[7] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[8] Çaglar Gülçehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. Pointing the unknown words. *CoRR*, abs/1603.08148, 2016.

[9] Tian Shi, Yaser Keneshloo, Naren Ramakrishnan, and Chandan K. Reddy. Neural abstractive text summarization with sequence-to-sequence models. *CoRR*, abs/1812.02303, 2018.