News Headline Generation

Elaine Lin, ***Daphne Chiou,*** ***Satya Vasanth,*** ***Weijie Tang***

{elin, daphnechiou, satya.vasanth, twjeric }@cs.ucla.edu

Abstract

We built an encoder-decoder RNN model with an attention mechanism for generating headlines of news articles inspired by Lopyrev's[1] paper. In this paper we explored the use of a contextual embeddings for text summarization tasks. We experimented models with different word embedding choices: ELMo and GloVe. We found that ELMo word embeddings showed improvement in models with a larger input size and higher number of training iterations. We also found our models biased towards specific words and phrases due to our training data. Finally, we highlighted some important issues related to our training data that are necessary to resolve before continuing on with future work.

1. Introduction

Headline generation is within the field of text summarization in natural language processing. However, there are enough differences in the task of headline generation that differentiates itself. Not only does a headline need to highlight the most important theme of an article, its content cover must be broad in a concise language. With the explosive amount of information available on the Internet, we could benefit from automatic machine-generated headlines by reducing the information down to its relevant bits. In this project, we would like to explore automatic headline generation using the content of news articles.

Text summarization involves two main strategies. The first one is an extractive summarization approach that simply selects a subset of actual sentences from the original documents as a summary. This approach does not suit the goal of headline generation since it is unlikely that there will be a compact enough yet coherent sentence present in the article that captures the most important message. The second approach is a generative summarization approach that builds the semantic representation of a document and creates a summary with sentences which are not necessarily presented in the original document. We will employ this method and adopts an end-to-end encoder-decoder framework to train a machine for our headline generation task.

We base our model on the one proposed by Konstantin Lopyrev [1], which uses an encoder-decoder framework that utilizes an attention mechanism. The encoder and the decoder are each a recurrent neural network (RNN). The encoder encodes a source article into a sequence of latent vectors, and the decoder outputs a summary word by word based on the latent vectors. The attention mechanism allows the decoder to attend to different parts of the source.

2. Encoder-decoder RNN Model with Attention Mechanism

The recurrent neural network in [1] uses the encoder-decoder architecture described in [2]. It has 2 RNN parts, the encoder and the decoder.

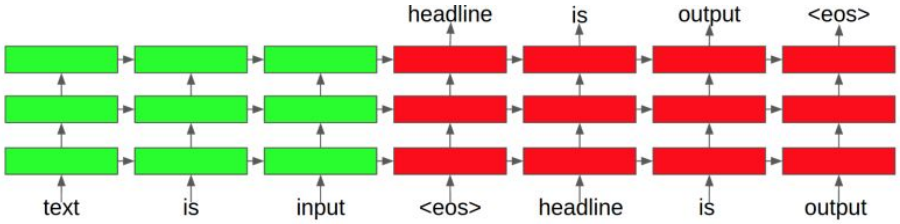

Figure 1: Encoder-decoder neural network architecture

The encoder takes the word embedding of the text (description) of a news article as inputs and feed them to a multi-layer neural network. The decoder takes the hidden layers as inputs, and an end-of-sequence symbol is fed in first, followed by the headline of the news article. Although we could use the actual headline words when training, we only have the predicted headline words when predicting. In order to overcome this disconnect, in practical, instead of using all the the actual headline words, we choose some generated headline words with a certain probability. In addition, the beam-search decoder is used to extend top k probability sequences when generating input words. The benefit of using encoder-decoder architecture is that it trains a single end-to-end model directly on the source and target sequences.

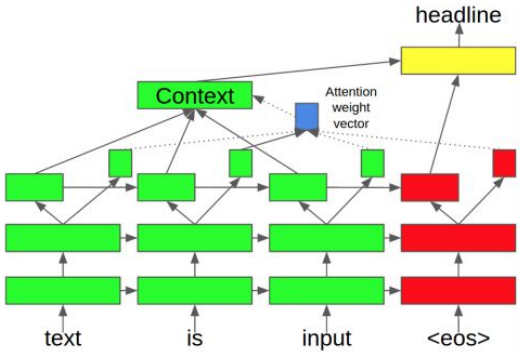

Figure 2: Simple attention

A potential limitation of encoder-decoder architecture is that it needs to compress all the source sequence into a fixed-length vector. The attention mechanism let the model to learn what to attend based on both the inputs and the current state by generating a context vector. [1] uses a simple context attention mechanism, in which the hidden units of the last layer of the decoder are splitted into 2 sets: a small one for computing the weights, and the other for the final softmax layer. The output weighted average is referred as the context.

3. ELMo & GloVe

For both the encoder and the decoder, an embedding layer is needed to transform the word into a distributed representation. We explore two of the most common ways of word vector representation: ELMo [3] and GloVe [4]. GloVe is an unsupervised learning algorithm to represent a word as a vector, which is trained on the aggregated global word-word co-occurrence statistics. Similarly, ELMo is a contextual word representation trained on bidirectional language model. It was first launched in 2018 and believed to have a better performance on many of the supervised NLP tasks owing to its ability to capture complex semantic and lexical features. In our experiments, both ELMo and GloVe embeddings are extracted from pretrained model and used as word representations when feeding the input text into the encoder. Note that a word can have several ELMo embeddings under different contexts. To overcome the problem, instead of adjusting the model architecture, we simply averaged over different embeddings in terms of each word to construct a embedding file with format similar to GloVe. This process, however, may result in embeddings that implicitly bias towards the training corpus.

4. Experiment

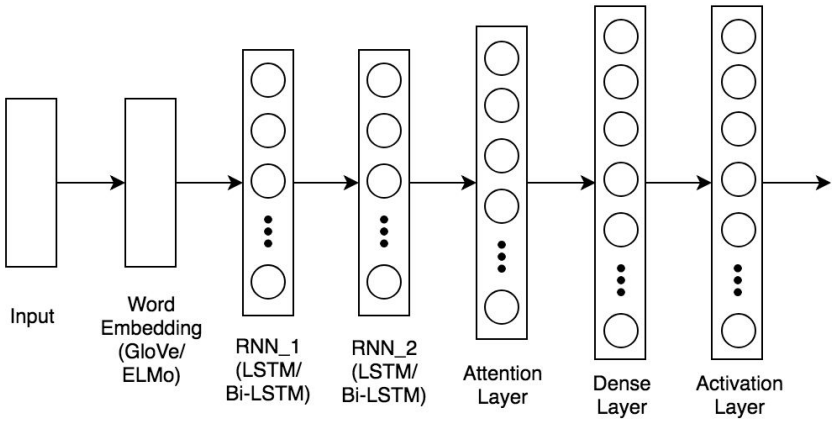

Figure 3. Model diagram

4.1 Dataset and preprocessing

Due to copyright issues, we used the all-the-news dataset by Kaggle [5], as we couldn't get the English Gigaword dataset. All-the-news dataset contains 143,000 articles from 15 American publications. For each article, we only used its title and content as the headline and the description of a news article, respectively. All the text is tokenized and lemmatized using the NLTK package.

We explored models trained with the first 50,000 articles and the first 5,000 articles. For the rest of the paper, we will address these two types of model as either 50k- or 5k-model. The size of the vocabulary from the 50,000 articles is 100,089 unique words; the size of the vocabulary from the 5,000 articles is 37512 unique words. Word distribution for both cases approximately follows Zipf's Law. The length of headlines in the examples center between 10 - 20 words in length. The length of description of the news

articles in the examples center between 150 words in length. Figure 4 shows these statistics using the 50k sample as an example. 5k sample follows these distributions very closely.

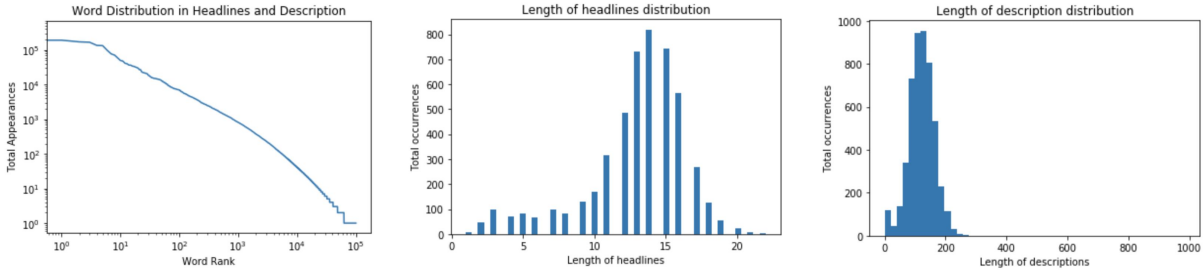

Figure 4. Statistics report for 50k articles from all-the-news dataset

4.2 Vocabulary embedding

Before feeding the text to the RNN, each word needs to be indexed and we need to get each word's vector representation. The first 5 sentences of each news article description are used by the model, and we use the pre-trained word vector representations, either from GloVe or ELMo, for each word in the corpus vocabulary. For any word in the corpus vocabulary that is not in the pre-trained vocabulary, the outside-of-vocabulary (OOV) is mapped to its closest match in GloVe/ELMo embedding matrix using the cosine distance of vectors. The embedding dimension was 100 in GloVe, and 300 in ELMo. Using the 50k training corpus, 39,123 tokens were found in GloVe and copied to the embedding matrix, and 40919 tokens found in ELMo and copied to the matrix, while the vocabulary size of the training corpus was 100,089. Using the 5k training corpus, 36,206 tokens were found in GloVe and copied to the embedding matrix, and 25,483 tokens found in ELMo and copied to the matrix, while the vocabulary size of the training corpus was 37,514.

4.3 Training

After the vocabulary embedding procedure, we used 90% of the training corpus as the training set and the rest of 10% as validation set. We performed padding and clipping for input of RNN. We constrain the maximum length of both the headlines and the descriptions of all articles to no more than 25 words. When the actual length is smaller than 25, we left pad the descriptions with the empty symbol and right pad the headlines with the empty symbol: [empty]. In addition, each headline and description is followed by an end-of-sequence symbol: [eos]. So the maximum length of the input sequence is 50 words in total: headline + description.

The first layer of the model is an embedding layer, which represents each word in the input as a vector of dimension either 100 or 300 in GloVe or ELMo, respectively. Then we add two RNN layers, each with the size of 512, which are used as the encoder-decoder architecture. After that, a context layer is added to implement the attention mechanism. Context layer reduces the input just to its headline part. For each word in this part, it concatenates the output of the previous layer (RNN) with a weighted average of the outputs of the description part. The first 40 outputs are used to compute the weights for the average. Finally, a dense layer and an activation layer are added to perform softmax on the output.

We experiment models using either 50k samples or 5k samples, and using either GloVe or ELMo for the vocabulary embedding matrix. We also experiment models trained with various training iterations.

We deploy the training process on the Google Cloud Platform. We train the models on a remote machine with 6 vCPUs and 1 NVIDIA Tesla K80 GPU.

4.4 Testing

We evaluate our models based on the BLEU (bilingual evaluation understudy) scores. It is an algorithm commonly used to evaluate the quality of machine translation and text summarization. BLEU score ranges between 0 to 1, and the value indicates the similarity between the generated headline to the reference text, with values closer to 1 representing more similar texts [6]. We will test our models on 78 news articles that are not in the training set, and calculate BLEU-1, BLEU-2, BLEU-3, and BLEU-4 for each generated headline. We will then report the average BLEU scores of each model.

4.5 Results and Discussion

The most time-consuming model we trained was the 50k-models trained with 500 iterations, and it took approximately 5 days to finish training. All 5k models took within a day to finish training.

We first explore the effect of training iterations on the model. Table 1 shows the results of 50k-models and 5k models trained with 200, 300, and 500 iterations, respectively. It is interesting to see that for 50k-model, 200 iterations outperformed the models with 300 and 500 iterations. However, 5k-models have higher BLEU scores as the training iterations increase. Overall, 5k-models seem to perform better than 50k-models. This is largely due to the fact that more than half of the vocabulary in 50k training corpus was outside of either GloVe or ELMo pretrained embeddings. As such, we had to use the cosine distance as an approximation for more than half of the corpus vocabulary. This possibly lowered the accuracy of our 50k-models by a significant amount.

Table 1. GloVe Models with Various Training Iterations

Training Sample Size	**50k**			**5k**		
Training Iterations	**200**	**300**	**500**	**200**	**300**	**500**
BLEU-1	0.051	0.025	0.014	0.035	0.035	0.052
BLEU-2	0.132	0.057	0.037	0.084	0.090	0.119
BLEU-3	0.186	0.080	0.055	0.116	0.127	0.167
BLEU-4	0.220	0.096	0.067	0.135	0.150	0.197

Next, we look at the effect of using ELMo instead of GloVe as an alternative for generating word vector representations. From the results, we can see that ELMo improved performance in 5k-200iter and 50k-500iter models.

Table 2. Models with ELMo Vocabulary Embedding

Training Sample Size	**5k**				**50k**			
Word Representation	**GloVe**		**ELMo**		**GloVe**		**ELMo**	
Training Iterations	200	500	200	500	200	500	200	500
BLEU-1	0.035	0.052	0.042	0.038	0.051	0.014	0.045	0.042
BLEU-2	0.084	0.119	0.099	0.100	0.132	0.037	0.106	0.117
BLEU-3	0.116	0.167	0.140	0.143	0.186	0.055	0.148	0.163
BLEU-4	0.135	0.197	0.166	0.170	0.220	0.067	0.174	0.195

Table 3 shows some examples generated by 50k model with GloVe vocabulary embedding and trained with 200 iterations. We observe two common issues in all models in our experiment. First, many headlines generated include the substrings "The New York Times" and "Breitbart", as seen in examples 1, 2, 4. This led us to discover a major issue with our samples - many provided headlines were appended with the name of the news publisher, and the New York Times and Breitbart News are two of the major news providers of the all-the-news dataset. Another issue we discovered is that our model tends to bias towards news involving Donald Trump or Barack Obama, as seen in examples 3, 4. We discovered that our samples are mostly news articles between 2016 - 2017, and therefore understandably news articles centering topics involving the presidential election took up a major portion of the sample.

Table 3. Examples of generated headlines by 50k-GloVe-200iter model

	Referenced Headline	Generated Headline (50k-GloVe, 200 iter.)
1	The Atlantic Politics Policy Daily Obama Surprises Biden With the Medal of Freedom	The San Francisco Your Monday Briefing The New York Times
2	On The Tonight Show Michelle Obama Cements a Legacy of Empathy	Kim Jong Un ISIS Your Morning Briefing The New York Times
3	Why Conservative Politicians May Be More Attractive Than Liberal Ones	Why *Trump* s about North Korea
4	Can Evangelicals Help Trump Thaw Relations With Russia	*Obama* Lives Matter *Trump* Is a Racist Breitbart

Nonetheless, we do observe some examples that indicate the potential of our learning machine. Table 4 shows a couple of cherry-picked results generated by 50k-ELMo-500iter model.

Table 4. Cherry-picked examples of generated headlines by 50k-ELMo-500iter model

	Referenced Headline	Generated Headline
1	Fidel Castro Cuba's revolutionary leader died aged 90	The Cuban Leader Fidel Castro's death
2	NBA national anthem singer kneel and reveals Black Lives Matter T-shirt	Black Lives Matter Protester in US
3	Mexican president contradicts Trump's account of border wall discussion	Mexico's President Donald Trump over Border Wall
4	The Trump Administration s Conflicts of Interest Scott Pruitt Edition	Trump Trump Cabinet Pick for a Breitbart
5	The Trump Administration s Conflicts of Interest A Crib Sheet	NYT Jon Stewart get suspended for

We look further into the quality of our models. Figure 5 shows an example of the BLEU Score Distribution for our 50k-ELMo-500iter model on testing against 78 examples. Although the figure shows the distribution for only one model, a common theme in all of our models in fact is that many generated headlines are completely irrelevant and thus have a BLEU score of 0. These poor-quality generated headlines represent the main factor that contribute to our low average BLEU scores across all of the models.

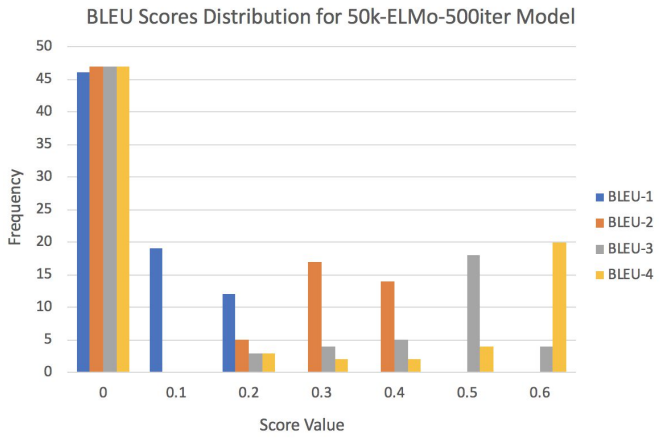

Figure 5. BLEU Score Distribution for 50k-ELMo-500iter Model

Finally, we investigate in the attention mechanism in play. Figure 6 shows two examples of the attention map generated from two samples. The X axis represents the first up to 25 words of article description, and the Y axis represents the generated headline by the model. From the left example, we see that "Trump and the and the Trump s" is generated by mainly paying attention to "impropriety" from the description. The example on the right has a more spread out attention weights for the last three words in the description, "a particle accelerator". Two characteristics are worth noting from these examples. The first is that over half of the description has very low attention weights. This makes the model prediction bias towards specific, short substrings in the description. The second is that a lot of the descriptions in our samples are not high-quality descriptions, such as 4b, whose description is more of a headline. Looking at

the training examples, we noticed that many examples whose descriptions were not indicative of the article, at least for the first 25 words. For example, from an article headlined “The Atlantic Daily Presidential Pressure”, we extracted the description of “This article is part of a feature we also send out via email a The Atlantic Daily a newsletter with story idea and image from The Atlantic written specially for subscriber”. These issues present as obstacles of the quality of our models.

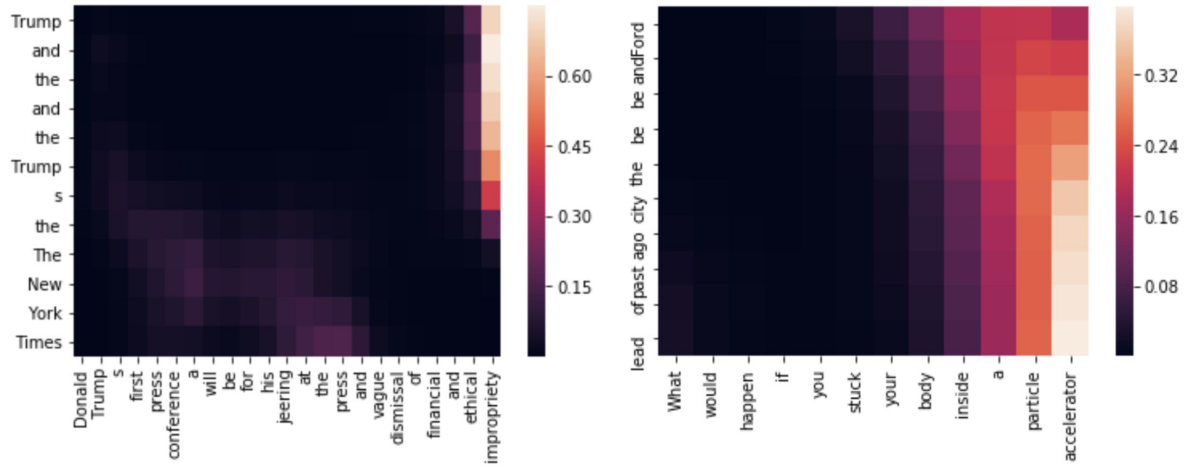

Figure 6. Attention weight maps generated from 50k-GloVe-200iter model

Last but not least, it is worth noting that Lopyrev’s model, the one we based off of, was trained on 5.5M news articles with 236M words. Our 50k training data size is less than 1% of that in comparison. Lopyrev’s model reported a 0.10 BLEU score at its best, whereas our best-performing model reported a 0.052 BLEU-1 score. Therefore, we believe that our model shows great potential if given more time for a much larger dataset.

5. Future Work

Instead of using LSTM layer, we looked at the effect of using Bi-LSTM layers. With the 50k dataset trained with 100 and 200 iterations, none of the models were able to generate headlines that have any overlapping n-gram with the referenced headlines causing all BLEU scores to be 0. We figure there must be a different way of using the architecture and this is something that can be further explored. Besides, we use averaged ELMo embeddings to avoid modifying the original model architecture, which may not represent the contextual information as precise as it would be without averaging. A possible solution to overcome the problem is to use ELMo embeddings as input and GloVe embeddings as the output of the model. Ideally, by training a mapping between input and output embeddings the model will be able to understand both representations. Simply put, we use ELMo embeddings for encoding the article description but use GloVe for the headlines. One more thing we observe is that the results are better on actual news articles rather than opinions and editorials. So picking an alternative dataset would also boost the model performance.

6. Conclusion

In this project, we implemented an encoder-decoder RNN model with an attention mechanism model for the task of automatic headline generation for news articles. We experimented models with different pre-trained word embedding: GloVe and ELMo. We compared our models with three variables: training corpus size, training iterations, and word embedding choices. We found that the 50k-GloVe-model trained with 200 iterations gave the best BLEU scores, although ELMo word embedding improved the performance of 50k-500iter and 5k-200iter models. Due to the dataset we used for training, the generated headlines were biased towards publishers and political leaders like Trump and Obama. We also observed that the attention layer put too much on the last few words of each news articles, which caused our model to become very sensitive to the last few words of the news content. The source code can be found here.

References

[1] Lopyrev, Konstantin. "Generating news headlines with recurrent neural networks." arXiv preprint arXiv:1512.01712(2015).

[2] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. CoRR, abs/1409.3215, 2014.

[3] Peters M E, Neumann M, Iyyer M, et al. Deep contextualized word representations[J]. arXiv preprint arXiv:1802.05365, 2018.

[4] 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.

[5] All the news dataset. Kaggle. https://www.kaggle.com/snapcrack/all-the-news/data

[6] Papineni, K.; Roukos, S.; Ward, T.; Zhu, W. J. BLEU: a method for automatic evaluation of machine translation. ACL-2002: 40th Annual Meeting of the Association for Computational Linguistics. Pp. 311-318.