Text Summarization for News

Rui Ma
Math Major Computer Science Minor
260572682

Yun-Fei Cheng
Computer Science
260784392

Xuer Liang
Computer Science
260506041

Abstract

With the present explosion of data circulating the digital space, which is mostly non-structured textual data, there is a need to develop automatic text summarization tools that allow people to get insights from them easily. Text summarization is the technique for generating a concise and precise summary of voluminous texts while focusing on the sections that convey useful information, and without losing the overall meaning. Machine learning algorithms can be trained to comprehend documents and identify the sections that convey important facts and information before producing the required summarized texts. In this paper, we aim to implement some of the state-of-the-arts of text summarization using both supervised and unsupervised methods.

1 Introduction

Currently, we enjoy quick access to enormous amounts of information. However, most of this information is redundant, insignificant, and may not convey the intended meaning. Therefore, using automatic text summarizers capable of extracting useful information that leaves out inessential and insignificant data is becoming vital.

Broadly, there are two approaches to summarizing texts in NLP: extraction and abstraction. In extraction-based summarization, a subset of words that represent the most important points is pulled from a piece of text and combined to make a summary. In abstraction-based summarization, advanced deep learning techniques are applied to paraphrase and shorten the original document, just like humans do. In this paper we used both of these two techniques to explore headline and summary generation. For headline generation, we explored the seq2seq model for extractive generation. For summary generation, we used the supervised models encoder-decoder with attention and pointer-network for abstractive summary generation and the unsupervised model GloVe+Text-Rank for extractive summary generation. Then we used both automatic and human evaluation methods to evaluate the generated texts.

2 Related Work

As early as the 1950's, text summarization was thought to be a trivial task. This paradigm quickly changed however, as it soon proved more challenging than we predicted.

Encoder-Decoder: An encoder neural network reads and encodes a source sentence into a fixed-length vector. A decoder then outputs a summarization from the encoded vector. The whole encoder–decoder system is jointly trained to maximize the probability of a correct summarization given a source sentence. A potential issue is that the vector cannot fully represent the information of the entire sequence, and the information carried by the previous content will be repetitively covered by the information entered later.

Attention: To solve the above problem, Ashish Vaswani et al. (10) proposed the Attention model. When this model generates an output, it also generates an attention range that indicates which parts of the input sequence to focus on in the next output, and then generates the next output based on the attention. In this way, the information carried by the input sequence can be fully utilized when generating each output.

Pointer-Generator Network: Unfortunately, the attention approach suffers from two problems. Firstly, the summaries sometimes reproduce factual details inaccurately, especially with rare or out-of-vocabulary words. Secondly, the summaries sometimes repeat themselves. The solution for the former problem is the pointer-generator network (3). This is a hybrid network that can choose to copy words from the source via pointing, while retaining the ability to generate words from the fixed vocabulary. To tackle the repetitive summaries problem, they use a technique called coverage. The idea is to use the attention distribution to keep track of what's been covered so far, and penalize the network for attending to same parts again.

3 Method

Our methodology is composed of 2 parts. The first is the implementation of various summarizer models, and the second involves automatic and human evaluation of said models. The first part can further be decomposed to supervised abstractive summarizer models, (i.e. seq2seq, encoder-decoder with attention, and pointer-network), and unsupervised extractive summarizer model GloVe+Text-Rank. We used seq2seq for headline generation and the rest for summary generation.

3.1 Dataset

This paper uses two news datasets found on Kaggle (1). One dataset, the Summary Dataset, consists of 4515 ex-

amples and contains author name, headline, url of article, gold standard summary and full article. The other dataset, the Headline Dataset, consists of 98401 examples and contains only the headline and article.

We used the summary dataset for summary generation. The dataset is split into two subsets with the first 500 examples as test set and the rest as training set. As to headline generation, we selected a subset of the headline dataset by the requirements of our model and the first 500 examples of the subset was held out as the test set.

Statistic of Data: The distribution of article length and summary length is shown in Figures 1 and 2. X-axis shows the number of words; Y-axis shows the count. The range of article length is 1 to 12202, and the range of summarization length between 44 and 62 words.

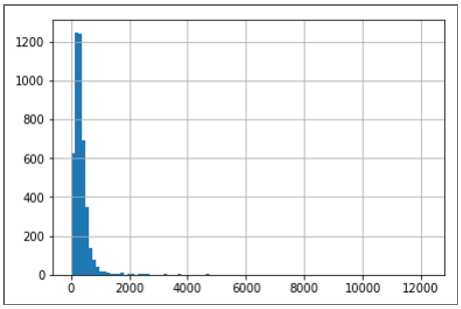

Figure 1: Distribution of article length

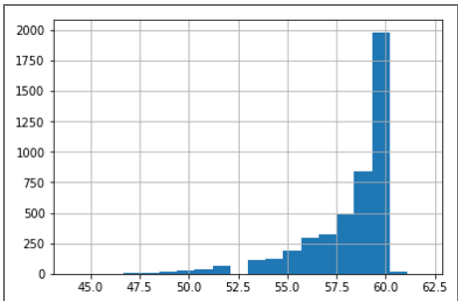

Figure 2: Distribution of summary length

3.2 Headline Generation

3.2.1 Supervised: seq2seq

A simple encoder-decoder model seq2seq (9) was trained on labeled data after basic preprocessing. Data preprocessing involved removal of empty articles, lowercasing, removal of stopwords and adding a ”start” and ”end” token before and after each article. We used this model for headline generation. In addition, only samples where the maximum length of article does not exceed 100 words and headline is under 15 words were chosen. 99.7% of the headlines have word count under 15, and 95.8% of articles have word count under 100.

The architecture of the model is in Figure 3.

We ran the model on 5 different sets of hyperparameters including number of epochs, optimizer and output length as shown in Table 1. Optimizers ”adam” and ”rmsprop” were tested not only because they have different training

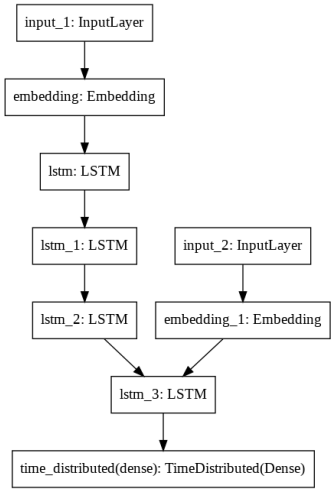

Figure 3: seq2seq model architecture

times, but because they often converge to different optima. The training results are summarized in Table 1 and visualized in Figure 4 and Figure. Step count is on X-axis and loss is on Y-axis. Validation test was run every 800 steps.

Hyperparameters	Train Loss	Val Loss
25epoch_rmsprop_15words	3.1332	3.3434
30epoch_rmsprop_15words	2.9967	3.2779
10epoch_rmsprop_15words	3.6624	3.6887
10epoch_adam_15words	3.4169	3.6544
10epoch_adan_20words	2.5113	2.7058

Table 1: Training results of simple seq2seq. Hyperparameters are the number of epochs, optimizer used, and max summary length.

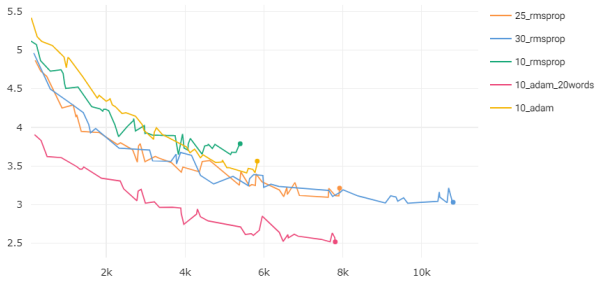

Figure 4: Batch Loss During Training

3.3 Summary Generation

3.3.1 Supervised: Encoder-Decoder with Attention

First, the encoder reads in the source text, producing a sequence of encoder hidden state. Second, on each step, the decoder receives the previous word of the summary as input and uses it to update the decoder hidden state which is used to calculate the attention distribution. Thirdly, the attention distribution is used to produce a weighted sum of the encoder hidden states, known as the context vector. Finally, the context vector and the decoder hidden states are used to calculate the distribution on a large fixed vocabulary. (10) The word with the largest probability is chosen as output (In this step, we always use beam search), and the

decoder moves on to the next step to generate abstractive summarization. Figure 5 shows the model architecture.

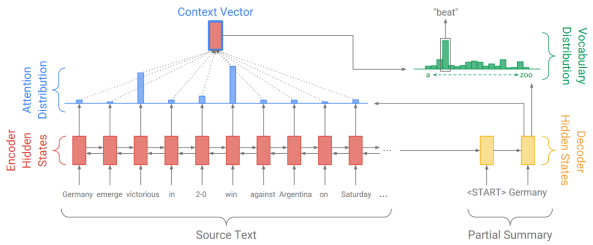

Figure 5: Baseline sequence-to-sequence model with attention. The model may attend to relevant words in the source text to generate novel words.

3.3.2 Supervised: Pointer Network

This model is built on top of the encoder-decoder with attention model. In this model, an additional step is added after the attention distribution and vocabulary distribution. This step calculates the generation probability, which is a scalar value between 0 and 1. The generation probability is used to weigh and combine the vocabulary distribution (3). Figure 6 shows the model architecture.

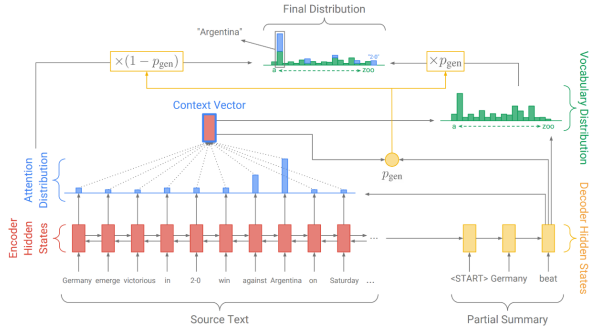

Figure 6: Pointer-generator model. For each decoder timestep, a generation probability P_{gen} is calculated. The vocabulary distribution and the attention distribution are weighed and summed to obtain the final distribution, from which we make our prediction. Note that out-of-vocabulary article words such as 2-0 are included in the final distribution.

3.3.3 Unsupervised: GloVe + Text-Rank

Unsupervised text extraction is useful for our dataset because the input size (average number of words in an article) is small. Extractive summarization is explored using Global Vectors (GloVe) embedding and text rank algorithm.

The text rank algorithm (7) is an unsupervised approach in that it removes the need to train data. The algorithm instead exploits the structure of text and chooses phrases that appear central or representative of other phrases. It is inspired by the famous page rank algorithm by google, where a notion of prestige of a website is used to calculate the rank of a website. In our implementation, instead of selecting key phrases, we chose to simplify by selecting top N key sentences as summary of the entire article so that no further processing is required to make the summary syntactically or semantically correct. Through empirical investigation, N = 3 produced reasonable summary of the articles, meaning the top 3 sentences were chosen as summary for articles. For articles with less than 3 sentences, N was set to 1 for a headline-esque summary.

The data was prepared first by simple preprocessing steps such as lowering the case of letters and removing stopwords. Then, pre-trained GloVe model was used for word embedding. (8) The advantage of GloVe over word2vec is that it incorporates global statistics (word co-occurrence). Using the word embeddings, a sentence embedding was created for each sentence. Then a sentence similiarity matrix was constructed by computing the cosine similarity. This was then passed to the page rank algorithm to compute the respective importance scores for each sentence. Finally, the top 3 sentences were outputted as the summary.

4 Evaluation

In this paper we used both automatic and human evaluation to judge the quality of the generated headlines and summaries. For automatic evaluation, we used 4 models to evaluate the similarity between the generated summary and the gold standard summary. For human evaluation, a member of the team scored a subset of generated summaries on a scale of 1 to 10 following a set of criteria.

4.1 ROUGE

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It includes measures to count the number of overlapping units such as n-gram, word sequences, and word pairs between the two summaries. We used the py-rouge module for this evaluation which includes two different measures: ROUGE-N and ROUGE-L. ROUGE-N measures unigram, bigram and trigram. ROUGE-L measures longest matching sequence of words using Longest Common Subsequence (LCS). An advantage of using LCS is that it does not require consecutive matches but in-sequence matches that reflect sentence level word order. (6)

$$\text{ROUGE-N} = \frac{\sum_{S \in \{ReferemceSummaries\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count(gram_n)}$$

Each ROUGE contains 3 scores, i.e. recall, precision and F1-score.

$$Recall = \frac{number\_of\_overlapping\_words}{total\_words\_in\_reference\_summary} \quad (1)$$

$$Precision = \frac{number\_of\_overlapping\_words}{total\_words\_in\_system\_summary} \quad (2)$$

$$F_1 = \frac{2}{recall^{-1} + precision^{-1}} \quad (3)$$

4.2 TF-IDF Cosine Similarity

The term frequency - inverse document frequency (also called tf-idf), is a well know method to evaluate how important a word is in a document. In this evaluation method the term-frequency is used to represent each term in the vector space.

$$TF(i,j) = \frac{Term\, i\, frequency\, in\, Document\, j}{Total\, words\, in\, Document\, j} \quad (4)$$

$$IDF(i) = \log_2(\frac{Total\, Documents}{documents\, with\, term\, i}) \quad (5)$$

$$TF - IDF = TF(i,j) * IDF(i) \quad (6)$$

The cosine similarity between two vectors (or two documents on the Vector Space) is a measure that calculates the cosine of the angle between them. This metric is a measurement of orientation and not magnitude. Cosine Similarity will generate a metric that says how related two documents ar.(5)

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

4.3 spaCy

spaCy is a natural language processing module that evaluates the similarity between sentences with its own built-in 300-dimension word vectors. spaCy also uses cosine similarity to compute the distances between two texts. spaCy's similarity model assumes a pretty general-purpose definition of similarity, so the result can be quite high even when two sentences don't seem to be related from a human perspective. To alleviate this effect, we pre-processed the text into relevant parts including removing stopwords and punctuation.(2)

4.4 Universal Sentence Encoder

The Universal Sentence Encoder (USE) encodes text into 512-dimensional vectors that can be used for text classification, semantic similarity, clustering and other natural language tasks. The model is trained and optimized for greater-than-word length text, such as sentences, phrases or short paragraphs. The model is trained with a deep averaging network (DAN) encoder. For the pairwise semantic similarity task, the model directly assess the similarity of the sentence embeddings produced by the encoder. It first computes the cosine similarity of the two sentence embeddings and then use arccos to convert the cosine similarity into an angular distance.(4)

$$\text{sim}(\mathbf{u}, \mathbf{v}) = \left(1 - \arccos\left(\frac{\mathbf{u} \cdot \mathbf{v}}{||\mathbf{u}||\ ||\mathbf{v}||}\right)/\pi\right)$$

4.5 Human Evaluation

Human evaluation was conducted only on summary generation by one member of the team. Fifty examples were selected from the generated headlines and another fifty from generated summaries. For each example the human judge read through the news article and the generated summary/headline but not the gold standard summary/headline, and then gave a score to each generated text based on the quality of the text defined by the following criteria: 1) Does the generated text include all the significant information of the news article? 2) Does the generated text include insignificant/non-relevant information? 3) Fluency and structure of the generated text. 4) If generated text is longer than original article, zero score will be given.

5 Results

The output scores of all automatic evaluation models are values ranging from 0 to 1, with 1 meaning the two texts are the same and 0 meaning no similarity between the two texts.

Overall, the two evaluation methods using high-dimensional word vectors for semantic analysis, i.e. USE and spaCy, give higher scores than the two methods that are based only on word count and frequency without taking into account the semantic information of the text. USE is currently considered the most accurate model in determining semantic similarity between texts thanks to its 512-dimensional word vector trained on millions of documents.

5.1 Automatic Evaluation Results

5.1.1 Headline Generation Results

We observe in Figure 7 that TF-IDF and ROUGE give very low scores for headline generation with TF-IDF at 0.05-0.20 and ROUGE at 0.10-0.30. We believe this is a reasonable result given that the generated headline is only 15 to 20 words. The probability of overlapping word or sequence between two texts is low, especially if the article is much longer than the headline. USE and spaCy give relatively high score on the headlines with both at 0.50-0.65 and the highest score is given by USE at 0.6699. The difference between the word-vector models and the non-word-vector models indicates that even though some headline pairs may not have overlapping words or sequence but they convey similar meanings, i.e. semantic similarity.

Overall, the optimizer rmsprop performs better than adam. The number of epochs also improves performance.

5.1.2 Summary Generation Results

As shown in Figure 8, the summary generation scores are higher than those of headline generation for all evaluation models. We believe the main reason for this is that the longer the two texts, the more likely they have overlapping words and overlapping semantic information, which leads to higher similarity scores.

	TF-IDF	USE	spaCy
Seq2Seq			
adam - 10 epochs - 15 words	0.0592	0.5430	0.5289
adam - 10 epochs - 20 words	0.0545	0.5314	0.5428
rmsprop - 10 epochs - 15 words	0.0934	0.5758	0.5627
rmsprop - 30 epochs - 15 words	0.1786	0.6699	0.6593

	adam - 10 epochs - 15 words			adam - 10 epochs - 20 words		
	F-score	Precision	Recall	F-score	Precision	Recall
ROUGE-1	0.1168	0.1242	0.1135	0.1122	0.1146	0.1134
ROUGE-2	0.0149	0.0159	0.0149	0.0114	0.0120	0.0113
ROUGE-3	0.0028	0.0029	0.0033	0.0010	0.0011	0.0010
ROUGE-L	0.1444	0.1518	0.1406	0.1394	0.1414	0.1405

	rmsprop - 10 epochs - 15 words			rmsprop - 30 epochs - 15 words		
	F-score	Precision	Recall	F-score	Precision	Recall
ROUGE-1	0.1460	0.1553	0.1414	0.2504	0.2689	0.2408
ROUGE-2	0.0279	0.0297	0.0275	0.0696	0.0758	0.0670
ROUGE-3	0.0067	0.0070	0.0070	0.0245	0.0275	0.0236
ROUGE-L	0.1743	0.1830	0.1694	0.2879	0.3044	0.2784

Figure 7: Automatic Evaluation of Headline Generation

As in the results of headline generation, in summary generation we also observe higher scores given by USE and spaCy than TF-IDF and ROUGE probably for the same reason that USE and spaCy use high-dimensional word vectors.

The unsupervised model GloVe+TextRank performs better than the two supervised models. The best score 0.9033 is given to GloVe+TextRank by spaCy. But this is because the unsupervised model directly extracts entire sentences as its summary. It should be evaluated by humans.

	TF-IDF	USE	spaCy
Supervised			
Attention	0.0638	0.6055	0.7746
Pointer Network	0.3113	0.8080	0.8786
Unsupervised			
GloVe + TextRank	0.3299	0.8103	0.9003

	Attention			Pointer Network			GloVe + TextRank		
	F-score	Precision	Recall	F-score	Precision	Recall	F-score	Precision	Recall
ROUGE-1	0.1659	0.2463	0.1260	0.4265	0.4660	0.3939	0.3975	0.3277	0.5161
ROUGE-2	0.0174	0.0261	0.0131	0.1955	0.2137	0.1804	0.1692	0.1397	0.2196
ROUGE-3	0.0052	0.0080	0.0039	0.1170	0.1281	0.1079	0.1037	0.0858	0.1341
ROUGE-L	0.1582	0.2187	0.1246	0.0808	0.0886	0.0745	0.3094	0.2627	0.3820

Figure 8: Automatic Evaluation of Summary Generation

5.2 Human Evaluation Results

Human evaluation shows very different scores from those of automatic evaluation. During human evaluation we observed that in general the generated summaries are not ideal summaries by human standards even though they do convey some important information of the article. Figure 9 shows the scores given by our human judge on a scale of 0 to 10.

The model that performs the best is Pointer Network. The human judge found that this model is capable of capturing the important information of the news and generating human readable and comprehensible summaries. Over 23% of all scored summaries received a high score of 9 or 10. What brings down the score in this model is the fluency. The generated summaries are often not well phrased and have missing words, but the gist of the article is properly conveyed.

GloVe+TextRank also performs relatively well. The problem with this model is that it outputs sentences in order of its importance, thereby making the summary non-comprehensible and illogical.

The worst performing model is Encoder-Decoder with Attention. The summaries generated by this model are mostly irrelevant to the news article.

	Human Evaluation
Headline	
seq2seq	**2.2759**
Summary	
Attention	**0.0000**
Pointer Network	**6.2619**
GloVe+TextRank	**5.1613**

Figure 9: Averaged Human Evaluation Results

6 Discussion and Conclusion

One interesting observation in the results of this paper is that the results of human evaluation and automatic evaluation are not consistent. In fact, these two evaluation methods are not comparable. Automatic evaluation is comparing generated text to gold standard text in a relatively objective manner while human evaluation is completely subjective despite using a set of criteria. There should be more than one good summary for each article and the definition of good summary is dependent on the reader. A summary that is not similar to the gold standard summary is not necessarily a bad summary.

However one observation should raise a red flag. The human judge found that the Encoder-Decoder with Attention model generated summaries that are mostly irrelevant to the news article but the USE and spaCy models give high scores for this model when comparing the generated summaries to the gold standard summaries. Future work may involve verifying the causes for this result.

Even though we used the same evaluation methods for both headline and summary generation, we believe it is reasonable to argue that we should evaluate headline with a different set of criteria. The purpose of headline is not only to convey important information but also to attract the reader's attention. The ability of a headline to attract attention and create curiosity is a criterion hard to judge and must involve a great amount of human judgement. Future work may explore this area.

Overall, we found that our results are far from reaching human parity in text summarization despite good results in some models. To improve performance, future work may need to create models that are able to grasp the semantic gist of the text and produce fluent text based on the gathered semantic data.

7 Statement of Contributions

Rui Ma: Attention and Pointer Network.
Yun-Fei Cheng: Encoder-decoder and GloVe + TextRank.
Xuer Liang: Evaluation models and human evaluation

References

[1] News summary dataset. [Online]. Available: https://www.kaggle.com/sunnysai12345/news-summary

[2] Word vectors and semantic similarity. [Online]. Available: https://spacy.io/usage/vectors-similarity

[3] C. D. M. Abigail See, Peter J. Liu, "Get to the point: Summarization with pointer-generator networks," 2017.

[4] S.-y. K. N. H. N. L. R. S. J. N. C. M. G.-C. S. Y. C. T. Y.-H. S. B. S. R. K. Daniel Cer, Yinfei Yang. (2018) Universal sentence encoder. [Online]. Available: https://arxiv.org/abs/1803.11175

[5] M. A. Faisal Rahutomo, Teruaki Kitasuka. (2012) Semantic cosine similarity. [Online]. Available: https://www.researchgate.net/publication/262525676_Semantic_Cosine_Similarity

[6] C.-Y. Lin. (2004) Rouge: A package for automatic evaluation of summaries. [Online]. Available: https://www.aclweb.org/anthology/W04-1013/

[7] R. Mihalcea and P. Tarau, "Textrank: Bringing order into text," in *EMNLP*, 2004.

[8] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[9] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to sequence learning with neural networks," *Advances in NIPS*, 2014.

[10] P. N. e. a. Vaswani A, Shazeer N, "Attention is all you need," 2017.