# News Article Summarization with Attention-based Deep Recurrent Neural Networks

**Hujia Yu**
Department of Management Science & Engineering
Stanford University
hujiay@stanford.edu

**Chang Yue**
Department of Electrical Engineering
Stanford University
changyue@stanford.edu

**Chao Wang**
Department of Electrical Engineering
Stanford University
cwang17@stanford.edu

## Abstract

This paper focuses on approaches to building a text automatic summarization model for news articles, generating a one-sentence summarization that mimics the style of a news title given some paragraphs. We managed to build and train two relatively complex deep learning models that outperformed our baseline model, which is a simple feed forward neural network. We explored Recurrent Neural Network models with encoder-decoder using LSTM and GRU cells, and with/without attention. We obtained some results that we then measured by calculating their respective ROUGE scores with respect to the actual references. For future work, we believe abstractive method of text summarization is a power way of summarizing texts, and we will continue with this approach. We think that the deficiencies currently embedded in our language model can be improved by better fine-tuning the model, more deep-learning method exploration, as well as larger training dataset.

## 1 Introduction

A news article often has a brief title that summarizes its content for readers to decide whether they want to read further. Therefore, a well-summarized title is crucial in delivering the main point of the news article to its potential readers. As a result, automatic text summarization techniques have a huge potential for news articles in that it expedites the process of summarizing a given documents for humans and, if models are well trained, generates the summary with a high accuracy. Our goal is to build a text automatic summarization model with deep learning algorithms that can output a one-sentence summarization given an article.

Models developed for text auto summarization has immediate applications in news articles title generations and beyond, such as machine translation, image captioning, as well as video summarization. There are in general two types of summarization techniques: extractive summarization and abstractive summarization, where the former summarizes articles by selecting a subset of words that retains

the most important points, and the latter generates a summary based on semantic understanding of the input. In this project we focus on abstractive approaches.

We approached the problem with attentive recurrent neural networks model, as proposed by Chopra et al [3]. We further explored the encoder-decoder model with GRU and LSTM cells. In addition, We also briefly investigated models with stacked RNNs. We wanted to see whether differing layers of RNNs can capture semantic meanings between words as we intended. Our dataset is 'The Signal Media One-Million News Articles Dataset', which contains a million news articles and their titles in the form of a jsonl file.

## 2 Background and Related Work

Significant amount of work has been done on automatic summarization. Since earlier models greatly focused on extractive methods, abstractive summarization is a relatively new research area with less experimentations. However, recent advance in research using abstractive methods has made it quite an popular field. In 2015, Cho et al described state-of-art performance of attention-based encoder-decoder networks, for which output possesses the same structure of input [1]. In the same year, Rush et al developed a neural attention feed-forward model for sentence-level summarization task which performed well on the DUC-2004 competition [2]. In 2016, Chopra et al designed attentive recurrent neural networks to improve the results on the same task [3]. The models we implemented are inspired by these three papers.

## 3 Approach

This section gives a brief overview of the architectures we approached with.

### 3.1 Baseline model using feed-forward neural network

We used a simple feed-forward neural network model as the baseline, where we first concatenated a window of C=5 words of current outputs using Glove.6B.100d embeddings, and then converted the vector to a 128d hidden state h. The next output word is then computed by applying onto h a function suggested by Rush et al [1].

$$p(y_{i+1}|y_c, x; \theta) \propto exp(Vh)$$
$$\tilde{y}_c = [Ey_{i-C+1}, ..., Ey_i]$$
$$h = tanh(U\tilde{y}_c)$$

### 3.2 Recurrent Neural Network Encoder-decoder

#### 3.2.1 RNNs

A recurrent neural network (RNN) is a class of neural network specialized at handling a variable-length input sequence $x = (x_1, ..., x_T)$ and optionally a corresponding variable-length output sequence $y = (y_1, ..., y_T)$ []. It uses an internal hidden state **h** to capture both the current input and the previous hidden state. A simple recurrent network (SRN) is shown below.

$$h_t = \sigma_h(W_h x_t + U_h h_{t-1} + b_h)$$
$$y_t = \sigma_y(W_y h_t + b_y)$$

W, U and b are parameters and they are the same at each time step. $\sigma_h$ and $\sigma_y$ are activation functions. Stacked (deep) RNNs are layered RNNs, where each layer extracts information from the previous layer.

#### 3.2.2 LSTMs and GRUs

Long short-term memory (LSTM) is a RNN architecture with complex hidden unit computations. Basically, by introducing gates (e.g. input, forget, output) and memory cells, it allows memorizing and forgetting over a long distance of training and the model is more immune to vanishing gradient

problem. Gated recurrent unit (GRU) can be treated as a simplified LSTM, as it has fewer parameters and fewer gates. LSTM and GRU are two basic units of our encoder-decoder models.

#### 3.2.3 Bidirectional RNNs

Bidirectional Recurrent Neural Network (BRNN) is another version of RNN. Unlike standard RNNs which only capture information from the current and past states, BRNNs determine outputs by inputs from the future, too. A standard version of BRNN is shown below:

$$\begin{aligned}\overrightarrow{h}_t &= f_H(\overrightarrow{W}x_t + \overrightarrow{V}\overrightarrow{h}_{t-1} + \overrightarrow{b}) \\ \overleftarrow{h}_t &= f_H(\overleftarrow{W}x_t + \overleftarrow{V}\overleftarrow{h}_{t+1} + \overleftarrow{b}) \\ y_t &= f_O(\overrightarrow{U}\overrightarrow{h}_t + \overleftarrow{U}\overleftarrow{h}_t + c)\end{aligned}$$

with right arrow representing forward direction and left arrow representing backward direction. W, U and V are parameters and they are different for different directions.

#### 3.2.4 Encoder-decoder network

An encoder-decoder framework is a general framework based on neural networks that aims at handling the mapping between highly structured input and output [1]. It consists of an encoder and a decoder. The encoder reads the input x and generates a continuous-space representation c, later the decoder generates final output y conditioned on c by computing the conditional probability. That is, the decoder starts from the final hidden state of encoder.

$$\begin{aligned}c &= f_{encoder}(x) \\ p(Y|x) &= f_{decoder}(c)\end{aligned}$$

Since our inputs and outputs are sentences, RNN becomes a natural choice for encoder and decoder. We tried all different types of RNNs described in above sections.

#### 3.2.5 Encoder-decoder with attention mechanism

For the simple encoder-decoder model, encoder returns only one vector of fixed dimensionality, which is not expressive in many cases, especially when the input gets long. Attentive encoder is designed to deal with this problem [4]. The idea behind is to compute context vector $c_t$ for every step t of the decoder. We used the model described in Chopra et al, details are formulated below.

$$\begin{aligned}z_{ik} &= \textstyle\sum_{h=-q/2}^{q/2} a_{i+h} \cdot b^k_{q/2+h} \\ c_t &= \textstyle\sum_{j=1}^{M} \alpha_{j,t-1} x_j \\ \alpha_{j,t-1} &= \frac{exp(z_j \cdot h_{t-1})}{\sum_{i=1}^{M} exp(z_i \cdot h_{t-1})}, z_i = [z_{i1}, ..., z_{id}]\end{aligned}$$

Here the full embedding for $x_i$ is $a_i = x_i + l_i$, where $l_i$ is a learned embedding. $b^k_j$ is the $j^{th}$ column of the learned weight matrix $B^k$. $c_t$ is a weighted average of $x_1, ...x_M$, and $\alpha$ is computed by taking softmax of $z_j$ and $h_{t-1}$.

## 4 Experiments

### 4.1 Data

For text summarization, one of the difficulties is the lack of quality summaries of large given dataset. Specifically, there are essentially no datasets that have paired human-generated summaries. For this reason, we have investigated a variety of popular datasets that have been used in the past. One famous dataset to use is the Gigaword, which was used in Chopra et al [3]. This dataset consists of news articles that are collected from a variety of news papers, and it uses the first sentence of the article as its summarization. Though it has been a popular data source, we find many of the summarizations generated this way to be unsatisfying to capture the main points in the articles. Therefore we decided to use The Signal Media One-Million News Articles Dataset, which is a

corpus that consists of a million news articles, in which the training summarizations are simply each articles respective title. Though this dataset is less experimented with before, we found the results generated from the dataset to be promising.

#### 4.1.1 Data Preprocessing

We used the first three paragraphs of each article in the dataset to be the original document input, which has a mean length of 154 words with standard deviation of 51 words. The titles were used as training target, which are 10-15 words long in general. Data preprocessing includes extracting contents and titles out of the original dataset and converting to proper formats, removals of samples that have too many rare words, removal of stop words, and lowercasing the inputs. The final dataset used consists of 800K samples, and is then separated to training set, cross validation set and test set with ratios of 0.70, 0.15, and 0.15.

### 4.2 Training

We used pre-trained GloVe 100d as our word embedding vectors, which are trained on aggregated global word-word co-occurrence statistics [5]. Input documents are tokenized and padded before training. In order to have a consistent training inputs, we limit the input size to be 100 words, and output size of 15 words.

### 4.3 Evaluation

The evaluation metrics are ROUGE-1 and ROUGE-2 on the test sets [6]. ROUGE-N is an n-gram recall between a candidate summary and a set of reference summaries [6]. It computes the fraction of matching n-grams with respect to the total number of n-grams in the reference summary. Python has a package to compute ROUGE-N.

$$ROUGE - N = \frac{\sum_{S \in reference summary} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in reference summary} \sum_{gram_n \in S} Count(gram_n)}$$

### 4.4 Models

We explored a variety of models with different layers and individual units. Specifically, we started with simple feed-forward neural network baseline (indicated by simply baseline in the results), then we trained a 4-layer RNN model with GRU units as encoder cells with and without attention, and 4-layer RNN model with LSTM units with and without attention. Lastly we got rid of one layer of each and trained 3-layer RNNs with GRUs and LSTMs as cells, with and without attention, respectively. In addition, we played with a 2-layer bidirectional RNN to gain a better understanding of efficiency of models. We used hidden size of 300, 400, 512 and batch size of 64, 128 for training. We found the better performing set of parameters are hidden size of 512, and batch size of 128.

We used RMSprop as our training updater, which divides the learning rates by an exponentially decaying average of squared gradients to resolve Adagrads radically diminishing learning rates. We also used MSE as loss function. The results are shown in Results below.

# 5 Results

## 5.1 Quantitative Performance

| Model | ROUGE-1 | ROUGE-2 |
|---|---|---|
| Baseline | 9.31 | 1.77 |
| A-RNN-GRU 4 | 13.64 | 4.23 |
| RNN-GRU 4 | 11.95 | 3.62 |
| A-RNN-LSTM 4 | 20.52 | 5.48 |
| RNN-LSTM 4 | 12.98 | 3.63 |
| A-RNN-GRU 3 | 12.64 | 3.49 |
| RNN-GRU 3 | 10.91 | 3.04 |
| A-RNN-LSTM 3 | 18.77 | 5.17 |
| RNN-LSTM 3 | 11.93 | 3.43 |
| Bidirectional-RNN 2 | 10.85 | 2.12 |

According to our results above, our best performing models are 3-layer and 4-layer RNN with LSTM as encoder and decoder cells with attention, from which we obtained ROUGE-1 scores of 20.52 and 18.77, respectively. We think this is largely because of advantages of LSTM cells that they have more complicated layer structures and they contain memorizing and forgetting cells, which allowed for better performance.

Also, for every RNN model that we used LSTM and GRU cells for, the one with LSTM always performed significantly better than that with GRU cells. This is interesting because a GRU is basically an LSTM but without an output gate, but the results show that LSTM are able to capture more significant information than GRUs.

In general, the models with attention performed better than those without, considering other parameters the same, which is what we had expected, since the purpose of adding attention is to account more for the hidden state. Also, adding more layers to the model will almost always improve the result by some margin, this is expected because more layers allowed for more hyper-parameters that could fit better to the training data. However, the higher-layered models are at risks of over fitting and vanishing gradients due to significant amount of existing hyper-parameters.

Overall, RNN-LSTM was significantly better than other models that we have implemented in several regards. First of all, the model had significantly better performance as already discussed before. Also, the model was much quicker to converge upon training.

### 5.2 Qualitative Performance

**input:**
Home \u00bb\rStyle \u00bb The Return Of The Nike Air Max Sensation Has 80\u2019s Babies Hyped!\tPosted on Sep 22, 2015\rIf you were a basketball fan who was born in the 80s, you were lucky enough to witness the beauty that is 90s basketball. It was truly a great time to be basketball fan. If you played close attention to what the players were wearing on their feet you would have also noticed the wide array of footwear these player used to rock. One of those happens to the the Nike Air Max Sensation, which is also set to receive the retro treatment this year!

**summary:** The Nike Air Max Sensation return baby
**reference:** The Return Of The Nike Air Max Sensation Has 80s Babies Hyped!

**input:**
VETERANS saluted Worcester's first ever breakfast club for ex-soldiers which won over hearts, minds and bellies. \n \nThe Worcester Breakfast Club for HM Forces Veterans met at the Postal Order in Foregate Street at 10am on Saturday. \n \nThe club is designed to allow veterans a place to meet, socialise, eat and drink, giving hunger and loneliness their marching orders. \n \nFather-of-two Dave Carney, aged 43, of Merrimans Hill, Worcester, set up the club after being inspired by other similar clubs across the country. \n \nHe said: \"As you can see from the picture, we had a good response.

**summary:** Jumpshot marketing analytics platform reveals the entire customer journey

**reference:** Jumpshot Gives Marketers Renewed Visibility Into Paid and Organic Keywords With Launch of Jumpshot Elite

We present here some snapshots of outputs generated by our model. The above sample outputs are some of the predictions of our trained model, where the input is the first 100 words of the input article, the summary is what the model had output, and the reference is the actual title of the article for that input. According to our sample outputs, the model is able to capture the key words in the input most of the times. However, the output meaning can be a bit off frequently due to incorrect grammar, as well as inability to capture all of the semantic meaning of the main point of input paragraphs. Specifically, we found the main issues with our model to be repetitive key words, and incorrect grammar that adds ambiguity to interpretation of the output.

For example, in the sample output 2, the summary is: The Nike Air Max Sensation return baby, where the actual reference is: The Return Of The Nike Air Max Sensation Has 80s Babies Hyped! This output word has a high ROUGE score due to correctly predicted words, however, the semantic meaning is quite different from the actual title. We believe this error points to deficiencies in the language model.

## 6 Conclusion and Future Work

In this paper, we discussed our approaches to building a text automatic summarization model for news articles, generating one-sentence summarization that mimics the style of news titles given some paragraphs. We managed to build and train two relatively complex deep learning models and outperformed our baseline model, which is a simple feed forward neural network. We explored Recurrent Neural Network models with encoders-decoders using LSTM and GRU cells, and with/without attention, and we obtained some results that were measured by calculating ROUGE scores with respect to the actual references. We believe abstractive method of text summarization is a power way of summarizing texts, and we will continue with this approach in the future. We think the deficiencies currently embedded in our language model can be improved by better fine-tuning the model, more deep learning method exploration, as well as larger training dataset.

Furthermore, we would be interested to find out the performance of our model on a different test set such as DUC. We are currently training, and testing our language model on the same sources

of references, just with dataset separation. It is quite popular among text summarization to test the output against DUC test items, which are news-related articles about topics that were relevant between 1994 and 2007. Since DUC has four references to test against, the ROUGE score would be more thoroughly reflexive of the performance of the model as well.

## References


[1] Cho, K., Courville, A. & Bengio. Y. (2014) Learning Phrase Representations using RNN EncoderDecoder for Statistical Machine Translation. *2014 Conference on Empirical Methods on Natural Language Processing (EMNLP).*

[2] Rush, A.M., Chopra, S. & Weston, J. (2015) A Neural Attention Model for Abstractive Sentence Summarization. *2015 Conference on Empirical Methods on Natural Language Processing (EMNLP).*

[3] Chopra, S., Auli M. & Rush, A.M. (2016) Abstractive Sentence Summarization with Attentive Recurrent Neural Networks. *2016 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology.*

[4] Bahdanau, D., Cho, K. & Bengio. Y. (2014) Neural machine translation by jointly learning to align and translate. *2015 Conference on International Conference on Learning Representation(ICLR).*

[5] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In EMNLP, volume 14, pages 15321543, 2014.

[6] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summariza- tion branches out: Proceedings of the ACL-04 workshop, volume 8, 2004.*