Neural Network-Based Abstract Generation for Opinions and Arguments

Lu Wang
College of Computer and Information Science
Northeastern University
Boston, MA 02115
luwang@ccs.neu.edu

Wang Ling
Google DeepMind
London, N1 0AE
lingwang@google.com

Abstract

We study the problem of generating abstractive summaries for opinionated text. We propose an attention-based neural network model that is able to absorb information from multiple text units to construct informative, concise, and fluent summaries. An importance-based sampling method is designed to allow the encoder to integrate information from an important subset of input. Automatic evaluation indicates that our system outperforms state-of-the-art abstractive and extractive summarization systems on two newly collected datasets of movie reviews and arguments. Our system summaries are also rated as more informative and grammatical in human evaluation.

1 Introduction

Collecting opinions from others is an integral part of our daily activities. Discovering what other people think can help us navigate through different aspects of life, ranging from making decisions on regular tasks to judging fundamental societal issues and forming personal ideology. To efficiently absorb the massive amount of opinionated information, there is a pressing need for automated systems that can generate concise and fluent opinion summary about an entity or a topic. In spite of substantial researches in opinion summarization, the most prominent approaches mainly rely on extractive summarization methods, where phrases or sentences from the original documents are selected for inclusion in the summary (Hu and Liu, 2004; Lerman et al., 2009). One of the problems that extractive methods suffer from is that they unavoidably include secondary or redundant information. On the contrary, abstractive summarization methods, which are able to generate text beyond the original input, can produce more coherent and concise summaries.

In this paper, we present an attention-based neural network model for generating abstractive summaries of opinionated text. Our system takes as input a set of text units containing opinions about the same topic (e.g. reviews for a movie, or arguments...
for a controversial social issue), and then outputs a one-sentence abstractive summary that describes the opinion consensus of the input.

Specifically, we investigate our abstract generation model on two types of opinionated text: movie reviews and arguments on controversial topics. Examples are displayed in Figure 1. The first example contains a set of professional reviews (or critics) about movie “The Martian” and an opinion consensus written by an editor. It would be more useful to automatically generate fluent opinion consensus rather than simply extracting features (e.g. plot, music, etc) and opinion phrases as done in previous summarization work (Zhuang et al., 2006; Li et al., 2010). The second example lists a set of arguments on “death penalty”, where each argument supports the central claim “death penalty deters crime”. Arguments, as a special type of opinionated text, contain reasons to persuade or inform people on certain issues. Given a set of arguments on the same topic, we aim at investigating the capability of our abstract generation system for the novel task of claim generation.

Existing abstract generation systems for opinionated text mostly take an approach that first identifies salient phrases, and then merges them into sentences (Bing et al., 2015; Ganesan et al., 2010). Those systems are not capable of generating new words, and the output summary may suffer from ungrammatical structure. Another line of work requires a large amount of human input to enforce summary quality. For example, Gerani et al. (2014) utilize a set of templates constructed by human, which are filled by extracted phrases to generate grammatical sentences that serve different discourse functions.

To address the challenges above, we propose to use an attention-based abstract generation model — a data-driven approach trained to generate informative, concise, and fluent opinion summaries. Our method is based on the recently proposed framework of neural encoder-decoder models (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014a), which translates a sentence in a source language into a target language. Different from previous work, our summarization system is designed to support multiple input text units. An attention-based model (Bahdanau et al., 2014) is deployed to allow the encoder to automatically search for salient information within context. Furthermore, we propose an importance-based sampling method so that the encoder can integrate information from an important subset of input text. The importance score of a text unit is estimated from a novel regression model with pairwise preference-based regularizer. With importance-based sampling, our model can be trained within manageable time, and is still able to learn from diversified input.

We demonstrate the effectiveness of our model on two newly collected datasets for movie reviews and arguments. Automatic evaluation by BLEU (Papineni et al., 2002) indicates that our system outperforms the state-of-the-art extract-based and abstract-based methods on both tasks. For example, we achieved a BLEU score of 24.88 on Rotten Tomatoes movie reviews, compared to 19.72 by an abstractive opinion summarization system from Ganesan et al. (2010). ROUGE evaluation (Lin and Hovy, 2003) also indicates that our system summaries have reasonable information coverage. Human judges further rated our summaries to be more informative and grammatical than compared systems.

2 Data Collection

We collected two datasets for movie reviews and arguments on controversial topics with gold-standard abstracts. Rotten Tomatoes (www.rottentomatoes.com) is a movie review website that aggregates both professional critics and user-generated reviews (henceforth Rotten Tomatoes). For each movie, a one-sentence critic consensus is constructed by an editor to summarize the opinions in professional critics. We crawled 246,164 critics and their opinion consensus for 3,731 movies (i.e. around 66 reviews per movie on average). We select 2,458 movies for training, 536 movies for validation and 737 movies for testing. The opinion consensus is treated as the gold-standard summary.

We also collect an argumentation dataset from idebate.org (henceforth Idebate), which is a Wikipedia-style website for gathering pro and con arguments on controversial issues. The arguments under each debate (or topic) are organized into dif-

---

1The datasets can be downloaded from http://www.ccs.neu.edu/home/luwang/.
ferent “for” and “against” points. Each point contains a one-sentence central claim constructed by the editors to summarize the corresponding arguments, and is treated as the gold-standard. For instance, on a debate about “death penalty”, one claim is “the death penalty deters crime” with an argument “enacting the death penalty may save lives by reducing the rate of violent crime” (Figure 1). We crawled 676 debates with 2,259 claims. We treat each sentence as an argument, which results in 17,359 arguments in total. 450 debates are used for training, 67 debates for validation, and 150 debates for testing.

3 The Neural Network-Based Abstract Generation Model

In this section, we first define our problem in Section 3.1, followed by model description. In general, we utilize a Long Short-Term Memory network for generating abstracts (Section 3.2) from a latent representation computed by an attention-based encoder (Section 3.3). The encoder is designed to search for relevant information from input to better inform the abstract generation process. We also discuss an importance-based sampling method to allow encoder to integrate information from an important subset of input (Sections 3.4 and 3.5). Post-processing (Section 3.6) is conducted to re-rank the generations and pick the best one as the final summary.

3.1 Problem Formulation

In summarization, the goal is to generate a summary $y$, composed by the sequence of words $y_1, ..., |y|$. Unlike previous neural encoder-decoder approaches which decode from only one input, our input consists of an arbitrary number of reviews or arguments (henceforth text units wherever there is no ambiguity), denoted as $x = \{x^1, ..., x^M\}$. Each text unit $x^k$ is composed by a sequence of words $x^k_1, ..., x^k_{|x^k|}$. Each word takes the form of a representation vector, which is initialized randomly or by pre-trained embeddings (Mikolov et al., 2013), and updated during training. The summarization task is defined as finding $\hat{y}$, which is the most likely sequence of words $\hat{y}_1, ..., \hat{y}_N$ such that:

$$\hat{y} = \arg\max_y \log P(y|x)$$

where $\log P(y|x)$ denotes the conditional log-likelihood of the output sequence $y$, given the input text units $x$. In the next sections, we describe the attention model used to model $\log P(y|x)$.

3.2 Decoder

Similar as previous work (Sutskever et al., 2014b; Bahdanau et al., 2014), we decompose $\log P(y|x)$ into a sequence of word-level predictions:

$$\log P(y|x) = \sum_{j=1, ..., |y|} \log P(y_j|y_1, ..., y_{j-1}, x)$$

where each word $y_j$ is predicted conditional on the previously generated $y_1, ..., y_{j-1}$ and input $x$. The probability is estimated by standard word softmax:

$$p(y_j|y_1, ..., y_{j-1}, x) = softmax(h_j)$$

$h_j$ is the Recurrent Neural Networks (RNNs) state variable at timestamp $j$, which is modeled as:

$$h_j = g(y_{j-1}, h_{j-1}, s)$$

Here $g$ is a recurrent update function for generating the new state $h_j$ from the representation of previously generated word $y_{j-1}$ (obtained from a word lookup table), the previous state $h_{j-1}$, and the input text representation $s$ (see Section 3.3).

In this work, we implement $g$ using a Long Short-Term Memory (LSTM) network (Hochreiter and Schmidhuber, 1997), which has been shown to be effective at capturing long range dependencies. Here we summarize the update rules for LSTM cells, and refer readers to the original work (Hochreiter and Schmidhuber, 1997) for more details. Given an arbitrary input vector $u_j$ at timestamp $j - 1$ and the previous state $h_{j-1}$, a typical LSTM defines the following update rules:

$$i_j = \sigma(W_{iu}u_j + W_{ih}h_{j-1} + W_{ic}c_{j-1} + b_i)$$

$$f_j = \sigma(W_{fu}u_j + W_{fh}h_{j-1} + W_{fc}c_{j-1} + b_f)$$

$$c_j = f_j \odot c_{j-1} + i_j \odot \tanh(W_{eu}u_j + W_{eh}h_{j-1} + b_c)$$

$$o_j = \sigma(W_{ou}u_j + W_{oh}h_{j-1} + W_{oc}c_j + b_o)$$

$$h_j = o_j \odot \tanh(c_j)$$

$\sigma$ is component-wise logistic sigmoid function, and $\odot$ denotes Hadamard product. Projection matrices
$W_{ss}$ and biases $b_s$ are parameters to be learned during training.

Long range dependencies are captured by the cell memory $c_j$, which is updated linearly to avoid the vanishing gradient problem. It is accomplished by predicting two vectors $i_j$ and $f_j$, which determine what to keep and what to forget from the current timestamp. Vector $o_j$ then decides on what information from the new cell memory $c_j$ can be passed to the new state $h_j$. Finally, the model concatenates the representation of previous output word $y_{j-1}$ and the input representation $s$ (see Section 3.3) as $u_j$, which serves as the input at each timestamp.

### 3.3 Encoder

The representation of input text units $s$ is computed using an attention model (Bahdanau et al., 2014). Given a single text unit $x_1, ..., x_{|x|}$ and the previous state $h_j$, the model generates $s$ as a weighted sum:

$$
\sum_{i=1,...,|x|} a_i b_i
$$

where $a_i$ is the attention coefficient obtained for word $x_i$, and $b_i$ is the context dependent representation of $x_i$. In our work, we construct $b_i$ by building a bidirectional LSTM over the whole input sequence $x_1, ..., x_{|x|}$ and then combining the forward and backward states. Formally, we use the LSTM formulation from Eq. 5 to generate the forward states $h_f^1, ..., h_f^{|x|}$ by setting $u_j = x_j$ (the projection word $x_j$ using a word lookup table). Likewise, the backward states $h_b^{|x|}, ..., h_b^1$ are generated using a backward LSTM by feeding the input in the reverse order, that is, $u_j = x_{|x|-j+1}$. The coefficients $a_i$ are computed with a softmax over all input:

$$
a_i = \text{softmax}(v(b_i, h_{j-1}))
$$

where function $v$ computes the affinity of each word $x_i$ and the current output context $h_{j-1}$. How likely the input word is to be used to generate the next word in summary. We set $v(b_i, h_{j-1}) = W_s \cdot \tanh(W_{sg} b_i + W_{sh} h_{j-1})$, where $W_s$ and $W_{ss}$ are parameters to be learned.

### 3.4 Attention Over Multiple Inputs

A key distinction between our model and existing sequence-to-sequence models (Sutskever et al., 2014b; Bahdanau et al., 2014) is that our input consists of multiple separate text units. Given an input of $N$ text units, i.e. $\{x_1^k, ..., x_{|x|}^k\}_{k=1}^N$, a simple extension would be to concatenate them into one sequence as $z = x_1^1, ..., x_1^k, ..., x_{|x|}^1, ..., x_{|x|}^N$, where $SEG$ is a special token that delimits inputs.

However, there are two problems with this approach. Firstly, the model is sensitive to the order of text units. Moreover, $z$ may contain thousands of words. This will become a bottleneck for our model with a training time of $O(N|z|)$, since attention coefficients must be computed for all input words to generate each output word.

We address these two problems by sub-sampling from the input. The intuition is that even though the number of input text units is large, many of them are redundant or contain secondary information. As our task is to emphasize the main points made in the input, some of them can be removed without losing too much information. Therefore, we define an importance score $f(x^k) \in [0, 1]$ for each document $x^k$ (see Section 3.5). During training, $K$ candidates are sampled from a multinomial distribution which is constructed by normalizing $f(x^k)$ for input text units. Notice that the training process goes over the training set multiple times, and our model is still able to learn from more than $K$ text units. For testing, top-$K$ candidates with the highest importance scores are collapsed in descending order into $z$.

### 3.5 Importance Estimation

We now describe the importance estimation model, which outputs importance scores for text units. In general, we start with a ridge regression model, and add a regularizer to enforce the separation of summary-worthy text units from others.

Given a cluster of text units $\{x^1, ..., x^M\}$ and their summary $y$, we compute the number of overlapping content words between each text unit and summary $y$ as its gold-standard importance score. The scores are uniformly normalized to $[0, 1]$. Each text unit $x^k$ is represented as an $d$-dimensional feature vector $r_k \in \mathbb{R}^d$, with label $l_k$. Text units in the training data are thus denoted with a feature matrix $\mathbf{R}$ and a label vector $\mathbf{L}$. We aim at learning $f(x^k) = r_k \cdot \mathbf{w}$ by minimizing $||\mathbf{Rw} - \mathbf{L}||_2^2 + \beta \cdot ||\mathbf{w}||_2^2$. This is a standard formulation for ridge regression, and we use fea-
features in Table 1. Furthermore, pairwise preference constraints have been utilized for learning ranking models (Joachims, 2002). We then consider adding a pairwise preference-based regularizing constraint to incorporate a bias towards summary-worthy text units: \( \lambda \cdot \sum_{r \in R} \sum_{x \in T, d \in T, d > 0, l = 0} \| (r_p - r_q) \cdot w - 1 \|_2^2 \), where \( T \) is a cluster of text units to be summarized. Term \((r_p - r_q) \cdot w\) enforces the separation of summary-worthy text from the others. We further construct \( \tilde{R}' \) to contain all the pairwise differences \((r_p - r_q)\). \( \tilde{L}' \) is a vector of the same size as \( \tilde{R}' \) with each element as \( \tilde{R}' \cdot \lambda \). The objective function becomes:

\[
J(w) = \| \tilde{R}w - \tilde{L} \|_2^2 + \lambda \cdot \| \tilde{R}'w - \tilde{L}' \|_2^2 + \beta \cdot \| w \|_2^2 \tag{8}
\]

\( \lambda, \beta \) are tuned on development set. With \( \tilde{R} = \beta \cdot I_d \) and \( \lambda = \lambda \cdot I_{|R'|} \), the closed-form solution for \( w \) is:

\[
w = (\tilde{R}^T \tilde{R} + \tilde{R}'^T \tilde{L} + \beta) \cdot (\tilde{R}^T \tilde{L} + \tilde{R}'^T \tilde{L}') \tag{9}
\]

Hyper-parameters and Stop Criterion. The LSTMs (Equation 5) for the decoder and encoders are defined with states and cells of 150 dimensions. The attention of each input word and state pair is computed by being projected into a vector of 100 dimensions (Equation 6).

Training is performed via Adagrad (Duchi et al., 2011). It terminates when performance does not improve on the development set. We use BLEU (up to 4-grams) (Papineni et al., 2002) as evaluation metric, which computes the precision of n-grams in generated summaries with gold-standard abstracts as the reference. Finally, the importance-based sampling rate \( K \) is set to 5 for experiments in Sections 5.2 and 5.3.

Decoding is performed by beam search with a beam size of 20, i.e. we keep 20 most probable output sequences in stack at each step. Outputs with end of sentence token are also considered for re-ranking. Decoding stops when every beam in stack generates the end of sentence token.

5 Results

5.1 Importance Estimation Evaluation

We first evaluate the importance estimation component described in Section 3.5. We compare with Support Vector Regression (SVR) (Smola and Vapnik, 1997) and two baselines: (1) a length baseline that ranks text units based on their length, and (2) a centroid baseline that ranks text units according
to their centroidness, which is computed as the cosine similarity between a text unit and centroid of the cluster to be summarized (Erkan and Radev, 2004).

Figure 2: Evaluation of importance estimation by mean reciprocal rank (MRR), and normalized discounted cumulative gain at top 3 and 5 returned results (NDCG@3 and NDCG@5). Our regression model with pairwise preference-based regularizer uniformly outperforms baseline systems on both datasets.

We evaluate using mean reciprocal rank (MRR), and normalized discounted cumulative gain at top 3 and 5 returned results (NDCG@3). Text units are considered relevant if they have at least one overlapping content word with the gold-standard summary. From Figure 2, we can see that our importance estimation model produces uniformly better ranking performance on both datasets.

5.2 Automatic Summary Evaluation

For automatic summary evaluation, we consider three popular metrics. ROUGE (Lin and Hovy, 2003) is employed to evaluate n-grams recall of the summaries with gold-standard abstracts as reference. ROUGE-SU4 (measures unigram and skip-bigrams separated by up to four words) is reported. We also utilize BLEU, a precision-based metric, which has been used to evaluate various language generation systems (Chiang, 2005; Angeli et al., 2010; Karpathy and Fei-Fei, 2014). We further consider METEOR (Denkowski and Lavie, 2014). As a recall-oriented metric, it calculates similarity between generations and references by considering synonyms and paraphrases.

For comparisons, we first compare with an abstractive summarization method presented in Ganesan et al. (2010) on the RottenTomatoes dataset. Ganesan et al. (2010) utilize a graph-based algorithm to remove repetitive information, and merge opinionated expressions based on syntactic structures of product reviews.\(^2\) For both datasets, we consider two extractive summarization approaches: (1) LEXRANK (Erkan and Radev, 2004) is an unsupervised method that computes text centrality based on PageRank algorithm; (2) Sipos et al. (2012) propose a supervised SUBMODULAR summarization model which is trained with Support Vector Machines. In addition, LONGEST sentence is picked up as a baseline.

Four variations of our system are tested. One uses randomly initialized word embeddings. The rest of them use pre-trained word embeddings, additional features in Table 2, and their combination. For all systems, we generate a one-sentence summary.

Results are displayed in Table 3. Our system with pre-trained word embeddings and additional features achieves the best BLEU scores on both datasets (in **boldface**) with statistical significance (two-tailed Wilcoxon signed rank test, \(p < 0.05\)). Notice that our system summaries are conciser (i.e. shorter on average), which lead to higher scores on precision based-metrics, e.g. BLEU, and lower scores on recall-based metrics, e.g. METEOR and ROUGE. On RottenTomatoes dataset, where summaries generated by different systems are similar in length, our system still outperforms other methods in METEOR and ROUGE in addition to their significantly better BLEU scores. This is not true on Idebate, since the length of summaries by extract-based systems is significantly longer. But the BLEU scores of our system are considerably higher. Among our four systems, models with pre-trained word embeddings in general achieve better scores. Though additional features do not always improve the performance, we find that they help our systems converge faster.

5.3 Human Evaluation on Summary Quality

For human evaluation, we consider three aspects: informativeness that indicates how much salient information is contained in the summary, grammaticality that measures whether a summary is grammatical, and compactness that denotes whether a summary contains unnecessary information. Each aspect is rated on a 1 to 5 scale (5 is the best). The judges are

\(^2\)We do not run this model on Idebate because it relies on high redundancy to detect repetitive expressions, which is not observed on Idebate.
also asked to give a ranking on all summary variations according to their overall quality.

We randomly sampled 40 movies from RottenTomatoes test set, each of which was evaluated by 5 distinct human judges. We hired 10 proficient English speakers for evaluation. Three system summaries (LexRank, Opinionis, and our system) and human-written abstract along with 20 representative reviews were displayed for each movie. Reviews with the highest gold-standard importance scores were selected.

Results are reported in Table 4. As it can be seen, our system outperforms the abstract-based system Opinionis in all aspects, and also achieves better informativeness and grammaticality scores than LexRank, which extracts sentences in their original form. Our system summaries are ranked as the best in 18% of the evaluations, and has an average ranking of 2.3, which is higher than both Opinionis and LexRank on average. An inter-rater agreement of Krippendorff’s α of 0.71 is achieved for overall ranking. This implies that our attention-based abstract generation model can produce summaries of better quality than existing summarization systems. We also find that our system summaries are constructed in a style closer to human abstracts than others. Sample summaries are displayed in Figure 3.

### 5.4 Sampling Effect

We further investigate whether taking inputs sampled from distributions estimated by importance scores trains models with better performance than the ones learned from fixed input or uniformly-sampled input. Recall that we sample K text units based on their importance scores (Importance-Based Sampling). Here we consider two other setups: one is sampling K text units uniformly from the input (Uniform Sampling), another is picking K text units with the highest scores (Top K). We try various K values. Results in Figure 4 demonstrates that Importance-Based Sampling can produce comparable BLEU scores to Top K methods, while both of them outperform Uniform Sampling. For METEOR score, Importance-Based Sampling uniformly outperforms the other two methods.

### 5.5 Further Discussion

Finally, we discuss some other observations and potential improvements. First, applying the re-ranking component after the model generates n-best abstracts leads to better performance. Preliminary experiments show that simply picking the top-1 gener-

---

3We observe similar results on the Idebate dataset.
### Related Work

Our work belongs to the area of opinion summarization. Constructing fluent natural language opinion summaries has mainly considered product reviews (Hu and Liu, 2004; Lerman et al., 2009), community question answering (Wang et al., 2014), and editorials (Paul et al., 2010). Extractive summarization approaches are employed to identify summary-worthy sentences. For example, Hu and Liu (2004) first identify the frequent product features and then attach extracted opinion sentences to the corresponding feature. Our model instead utilizes abstract generation techniques to construct natural language summaries. As far as we know, we are also

---

**Figure 3:** Sample summaries generated by different systems on movie reviews and arguments. We only show a subset of reviews and arguments due to limited space.

**Figure 4:** Sampling effect on RottenTomatoes.
the first to study claim generation for arguments.

Recently, there has been a growing interest in generating abstractive summaries for news articles (Bing et al., 2015), spoken meetings (Wang and Cardie, 2013), and product reviews (Ganesan et al., 2010; Di Fabbrizio et al., 2014; Gerani et al., 2014). Most approaches are based on phrase extraction, from which an algorithm concatenates them into sentences (Bing et al., 2015; Ganesan et al., 2010). Nevertheless, the output summaries are not guaranteed to be grammatical. Gerani et al. (2014) then design a set of manually-constructed realization templates for producing grammatical sentences that serve different discourse functions. Our approach does not require any human-annotated rules, and can be applied in various domains.

Our task is closely related to recent advances in neural machine translation (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014a). Based on the sequence-to-sequence paradigm, RNNs-based models have been investigated for compression (Filippova et al., 2015) and summarization (Filippova et al., 2015; Rush et al., 2015; Hermann et al., 2015) at sentence-level. Built on the attention-based translation model in Bahdanau et al. (2014), Rush et al. (2015) study the problem of constructing abstract for a single sentence. Our task differs from the models presented above in that our model carries out abstractive decoding from multiple sentences instead of a single sentence.

7 Conclusion

In this work, we presented a neural approach to generate abstractive summaries for opinionated text. We employed an attention-based method that finds salient information from different input text units to generate an informative and concise summary. To cope with the large number of input text, we deploy an importance-based sampling mechanism for model training. Experiments showed that our system obtained state-of-the-art results using both automatic evaluation and human evaluation.

References


Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In Proceedings of the 23rd international conference on com-


