Transformer-based Model for Single Documents Neural Summarization

Elozino Egonmwan and **Yllias Chali**
University of Lethbridge
Lethbridge, AB, Canada
{elozino.egonmwan, yllias.chali}@uleth.ca

Abstract

We propose a system that improves performance on single document summarization task using the CNN/DailyMail and Newsroom datasets. It follows the popular encoder-decoder paradigm, but with an extra focus on the encoder. The intuition is that the probability of correctly decoding an information significantly lies in the pattern and correctness of the encoder. Hence we introduce, encode – encode – decode. A framework that encodes the source text first with a transformer, then a sequence-to-sequence (`seq2seq`) model. We find that the transformer and seq2seq model complement themselves adequately, making for a richer encoded vector representation. We also find that paying more attention to the vocabulary of target words during abstraction improves performance. We experiment our hypothesis and framework on the task of extractive and abstractive single document summarization and evaluate using the standard CNN/DailyMail dataset and the recently released Newsroom dataset.

1 Introduction

Document summarization has been an active area of research, especially on the CNN/DailyMail dataset. Even with recent progress (Gehrmann et al., 2018; Chen and Bansal, 2018), there is still some work to be done in the field. Although `extractive summarization` seem to be less challenging because new words are not generated, identifying salient parts of the document without any guide in the form of a query, is a substantial problem to tackle.

Earlier approaches for extractive summarization use manual-feature engineering implemented with graphs (Parveen and Strube, 2015; Erkan and Radev, 2004), integer linear programming (ILP) (Boudin et al., 2015; Nayeem and Chali, 2017). More recent approaches are data-driven and implement a variety of neural networks (Jadhav and Rajan, 2018; Narayan et al., 2017) majorly with an encoder-decoder framework (Narayan et al., 2018; Cheng and Lapata, 2016).

Similar to the work of Nallapati et al. (2017), we consider the extractive summarization task as a sequence classification problem. A major challenge with this approach, is the fact that the training data is not sequentially labelled. Hence creating one from the abstractive ground-truth summary, is crucial. We improve on Nallapati et al. (2017)'s approach to generate this labelled data, and evaluation shows that our extractive labels are more accurate. Another hurdle in this task, is the imbalance in the created data, that is, most of the document's sentences are labelled 0 (excluded from the summary) than 1, because just a few sentences actually make up a summary. Hence the neural extractor tends to be biased and suffer from a lot of false-negative labels. We also present a simple approach to reduce this bias. Most importantly, our neural extractor uses the recent bidirectional transformer encoder (Vaswani et al., 2017) with details provided in Section 3.1.

More interesting than extractive summaries, abstractive summaries correlate better with summaries that a human would present. `Abstractive summarization` does not simply reproduce salient parts of the document verbatim, but rewrites them in a concise form, usually introducing novel words along the way by utilizing some key abstraction techniques such as paraphrasing (Gupta et al., 2018), compression (Filippova et al., 2015) or sentence fusion (Barzilay and McKeown, 2005). However, it is met with major challenges like grammatical correctness and repetition of words especially when generating long-worded sentences. Nonetheless remarkable progress have been achieved with the

Proceedings of the 3rd Workshop on Neural Generation and Translation (WNGT 2019), pages 70–79
Hong Kong, China, November 4, 2019. ©2019 Association for Computational Linguistics
www.aclweb.org/anthology/D19-56%2d

use of seq2seq models (Gehrmann et al., 2018; See et al., 2017; Chopra et al., 2016; Rush et al., 2015) and a reward instead of loss function via deep-reinforcement learning (Chen and Bansal, 2018; Paulus et al., 2017; Ranzato et al., 2015).

We see abstractive summarization in same light as several other authors (Chen and Bansal, 2018; Hsu et al., 2018; Liu et al., 2018) – extract salient sentences and then abstract; thus sharing similar advantages as the popular divide-and-conquer algorithm. More-so, it mitigates the problem of information redundancy, since the mini-source, ie extracted document, contains distinct salient sentences. Our abstractive model is a blend of the transformer and seq2seq model. We notice improvements using this framework in the abstractive setting. This is because, to generate coherent and grammatically correct sentences, we need to be able to learn long-term dependency relations. The transformer complements the seq2seq model in this regard with its multi-head self attention. Also the individual attention heads in the transformer model mimics behavior related to the syntactic and semantic structure of the sentence (Vaswani et al., 2017, 2018). Hence, the transformer produces a richer meaningful vector representation of the input, from which we can encode a fixed state vector for decoding.

The main contributions of this work are:

- We present a simple algorithm for building a sentence-labelled corpus for extractive summarization training that produces more accurate results.

- We propose a novel framework for the task of extractive single document summarization that improves the current state-of-the-art on two specific datasets.

- We introduce the encode - encode - decode paradigm using two complementary models, transformer and seq2seq for generating abstractive summaries that improves current top performance on two specific datasets.

2 Task Definition

Given a document $D = (S_1, ..., S_n)$ with n sentences comprising of a set of words $D_W = \{d_1, ..., d_w\}$, the task is to produce an *extractive* (S_E) or *abstractive* (S_A) summary that contains salient information in D, where $S_E \subseteq D_W$ and $S_A = \{w_1, ..., w_s\} \mid \exists w_i \notin D_W$.

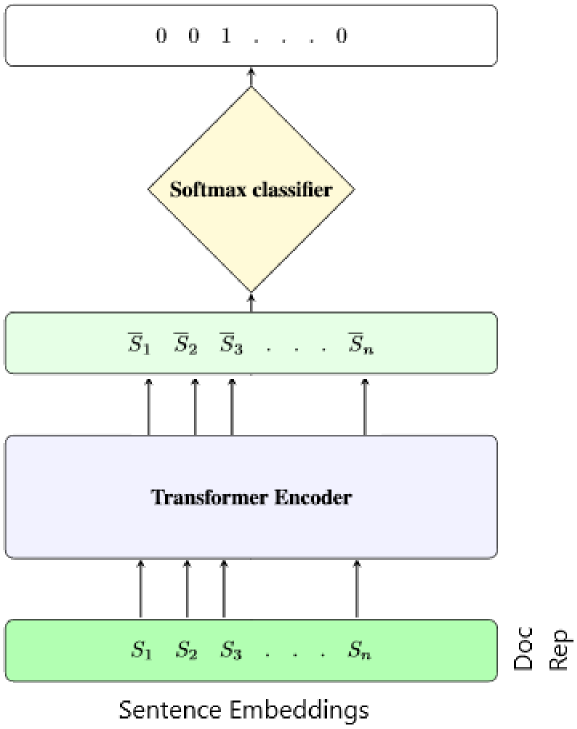
0 0 1 . . . 0
Softmax classifier
$\bar{S}_1$ $\bar{S}_2$ $\bar{S}_3$ . . . $\bar{S}_n$
Transformer Encoder
S_1 S_2 S_3 . . . S_n
Doc Rep
Sentence Embeddings

Figure 1: Extractive Model Architecture

3 Method

We describe our summarization model in two modules – *Extraction* and *Abstraction*. The abstraction module simply learns to paraphrase and compress the output of the extracted document sentences.

3.1 Extraction

As illustrated in Figure 1, our model classifies each sentence in a document as being summary-worthy or not. However, in order to enhance this sequence classification process, we encode the input document with a TRANSFORMER. A logistic classifier then learns to label each sentence in the transformed document.

3.1.1 TRANSFORMER Encoder

The input to the Transformer is the document representation, which is a concatenation of the vector representation of its sentences. Each sentence representation is obtained by averaging the vector representation of its constituent words.

$$S_i = 1/m \sum_{i=1}^{m} w_i \tag{1}$$

$$D_j = S_1 \| S_2 \| \ldots \| S_n \tag{2}$$

The transformer encoder is composed of 6 stacked identical layers. Each layer contains 2 sub-layers with multi-head self attention and position-wise fully connected feed-forward network respectively. Full details with implementation are provided in (Vaswani et al., 2017, 2018). The bidirectional Transformer often referred to as the Transformer encoder learns a rich representation of the document that captures long-range syntactic and semantic dependency between the sentences.

3.1.2 Sentence Extraction

The final layer of our extraction model is a softmax layer which performs the classification. We learn the probability of including a sentence in the summary,

$$y_p^i = softmax(WS_i^{`} + b) \tag{3}$$

where W and b are trainable parameters and $S_i^{`}$ is the transformed representation of the i^{th} sentence in document D_j, by minimizing the cross-entropy loss

$$L = -(y_t log(y_p) + (1 - y_t) log(1 - y_p)) \tag{4}$$

between the predicted probabilities, y_p and true sentence-labels, y_t during training.

3.1.3 Extractive Training

Filtering Currently, no extractive summarization dataset exists. Hence it is customary to create one from the abstractive ground-truth summaries (Chen and Bansal, 2018; Nallapati et al., 2017). We observe however, that some summaries are more abstractive than others. Since the extractive labels are usually gotten by doing some n-gram overlap matching, the greater the abstractiveness of the ground-truth the more inaccurate the tuned extractive labels are. We filter out such samples [1] as illustrated in Table 1. In our work, we consider a reference summary R_j as *overly abstractive* if it has zero bigram overlap with the corresponding document D_j, excluding stop words.

$$\#bigram(D_j, R_j) == 0 \tag{5}$$

See et al. (2017) and Paulus et al. (2017) truncate source documents to 400 tokens and target summaries to 100 tokens. We totally exclude documents with more than 30 sentences and truncate or pad as necessary to 20 sentences per document. From the over 280,000 and 1.3M training pairs in the CNN/DM and Newsroom training dataset respectively, our filtering yields approximately 150,000 and 250,000 abstractive summarization sub-dataset. We report evaluation scores using the training sets as-is versus our filtered training sets, to show that filtering the training samples does improve results.

Document: world-renowned chef, author and emmy winning television personality anthony bourdain visits quebec in the next episode of “ anthony bourdain : parts unknown, ” airing sunday, may 5, at 9 p.m. et. follow the show on twitter and facebook.
Summary: 11 things to know about quebec. o canada! our home and delicious land.’

Table 1: Example of an *overly abstractive* summary with zero bigram overlap with the document from a CNN/DM training sample.

Tuning We use a very simple approach to create extractive labels for our neural extractor. We hypothesize that each reference summary sentence originates from at least one document sentence. The goal is to identify the most-likely document sentence. Different from Nallapati et al. (2017)’s approach to greedily add sentences to the summary that maximizes the ROUGE score, our approach is more similar to Chen and Bansal (2018)’s model that calculates the individual reference sentence-level score as per its similarity with each sentence in the corresponding document. However, our sentence-level similarity score is based on its bigram overlap:

$$score(R_j^t) = amax_i(bigram(D_j^i, R_j^t)) \tag{6}$$

for each t^{th} sentence in the reference summary, R_j, per i^{th} sentence in document D_j, in contrast to Chen and Bansal (2018)’s that uses ROUGE-L$_{recall}$ score. Additionally, for every time both words in the set of bigrams-overlap are stopwords, we decrement the similarity score by 1, for example, *(on, the)* is an invalid bigram-overlap while *(the, President)* is valid. We do this, to capture more important similarities instead of trivial ones.

For statistical purposes, we evaluate our extractive trainer for tuning the document’s sentences to

[1] Filtering is used only for the training set, to ensure that evaluation comparisons on the test set with existing models are fair

0's and 1's against (Nallapati et al., 2017)'s which is our foundation.

Extractive Trainer	R-1	R-2	R-L
Ours	49.5	27.8	45.8
Ours + filter	**51.4**	**31.7**	**50.3**
(Nallapati et al., 2017)	48.4	27.5	44.4

Table 2: **ROUGE-F1** (%) scores of manually crafted extractive trainers for producing sentence-level extractive labels for CNN/DM.

We apply our tuned dataset to the neural extractive summarizer explained in Sections 3.1.1 and 3.1.2 and report results in Tables 3 and 4.

Imbalanced Extractive Labels Because a summary is a snippet of the document, the majority of the labels are rightly 0 (excluded from the summary). Hence a high classification accuracy does not necessarily translate to a highly salient summary. Therefore, we consider the F1_score, which is a weighted average of the precision and recall, and apply an early stopping criteria when minimizing the loss, if the F1_score does not increase after a set number of training epochs. Additionally during training, we synthetically balance the labels, by forcing some random sentences to be labelled as 1 and subsequently masking their weights.

Number of sentences to extract The number of extracted sentences is not trivial, as this significantly affects the summary length and hence evaluation scores. Chen and Bansal (2018) introduced a stop criterion in their reinforcement learning process. We implemented a basic subjective approach based on the dataset. Since the gold summaries are typically 3 or 4 sentences long, we extract the top 3 sentences by default, but proceed to additionally extract a 4th sentence if the confidence score from the softmax function is greater than 0.55.

3.2 Abstraction

The input to our abstraction module is a subset of the document's sentences which comprises of the output of the extraction phase from Section 3.1.2. For each document D_j, initially comprising of n sentences, we abstract its extracted sentences,

$$S_j^E = \{S_j^1, S_j^2, ..., S_j^m\} \tag{7}$$

where $m < n$ and $S_j^E \subseteq D_j$, by learning to jointly paraphrase (Gupta et al., 2018) and compress (Filippova et al., 2015). We add one more encoding layer to the standard encoder-aligner-decoder (Bahdanau et al., 2014; Luong et al., 2015), ie, encode-encode-align-decode. The intuition is to seemingly improve the performance of the decoder by providing an interpretable and richly encoded sequence. For this, we interleave two efficient models – transformer (Vaswani et al., 2017) and sequence-to-sequence (Sutskever et al., 2014), specifically GRU-RNN (Chung et al., 2014; Cho et al., 2014). Details are presented in subsequent subsections.

3.2.1 Encoder – TRANSFORMER

The transformer encoder has same implementation from Vaswani et al. (2017) as explained in Section 3.1.1, except the inputs are sentence-level vector representations not document. Also, the sentence representations in this module are not averaged constituent word representations as in the extraction module but concatenated. That is, for each i^{th} sentence in equation 7, its vector representation, is the concatenation of its constituent word embeddings

$$S_j^i = w_1 \| w_2 \| \ldots \| w_n \tag{8}$$

The output of equation 8 serves as the input vector representation to the transformer encoder. We use the transformer-encoder during abstraction as sort of a pre-training module of the input sentence.

3.2.2 Encoder – GRU-RNN

We use a single layer uni-directional GRU-RNN whose input is the output of the transformer. The GRU-RNN encoder (Chung et al., 2014; Cho et al., 2014) produces fixed-state vector representation of the transformed input sequence using the following equations:

$$z = \sigma(s_t U^z + x_{t-1} W^z) \tag{9}$$

$$r = \sigma(s_t U^r + x_{t-1} W^r) \tag{10}$$

$$h = tanh(s_t U^h + (x_{t-1} \odot r) W^h) \tag{11}$$

$$x_t = (1 - z) \odot h + z \odot x_{t-1} \tag{12}$$

where r and z are the reset and update gates respectively, W and U are the network's parameters, x_t is the hidden state vector at timestep t, s_t is the input vector and $\odot$ represents the Hadamard product.

Extractive Model	R-1	R-2	R-L
LEAD (See et al., 2017)	40.3	17.7	36.5
LEAD (Narayan et al., 2018)	39.6	17.7	36.2
LEAD (ours)	40.1	17.6	36.0
(Nallapati et al., 2017)	39.6	16.2	35.3
REFRESH (Narayan et al., 2018)	40.0	18.2	36.6
FAST (Chen and Bansal, 2018)	41.4	18.7	37.7
NEUSUM (Zhou et al., 2018)	41.6	19.0	37.0
Content Selector (Gehrmann et al., 2018)	42.0	15.9	37.3
TRANS-ext	41.0	18.4	36.9
TRANS-ext + filter	**42.8**	**21.1**	**38.4**

Table 3: **ROUGE-F1** (%) scores (with 95% confidence interval) of various extractive models on the **CNN/DM** test set. The first section shows LEAD-3 model scores. The second section shows scores for baseline models. The third section shows our model's scores

Extractive Model	R-1	R-2	R-L
LEAD* (Grusky et al., 2018)	30.49	21.27	28.42
TextRank* (Barrios et al., 2016)	22.77	9.79	18.98
TRANS-ext	37.21	25.17	32.41
TRANS-ext + filter	**41.52**	**30.62**	**36.96**

Table 4: **ROUGE-F1** (%) scores (with 95% confidence interval) of various extractive models on the **Newsroom** released test set. * marks results taken from Grusky et al. (2018)

3.3 Decoder – GRU-RNN

The fixed-state vector representation produced by the GRU-RNN encoder is used as initial state for the decoder. At each time step, the decoder receives the previously generated word, y_{t-1} and hidden state s_{t-1} at time step t_{-1}. The output word, y_t at each time step, is a softmax probability of the vector in equation 11 over the set of vocabulary words, V.

4 Experiments

We used pre-trained 300-dimensional $gloVe$[2] word-embeddings (Pennington et al., 2014). The transformer encoder was setup with the $transformer\_base$ hyperparameter setting from the tensor2tensor library (Vaswani et al., 2018)[3], but the hidden size and dropout were reset to 300 and 0.0 respectively. We also use 300 hidden units for the GRU-RNN encoder. The tensor2tensor library comes with pre-processed/tokenized versions of the dataset, we however perform these operations independently. For abstraction, our target vocabulary is a set of approximately 50,000 and 80,000 words for CNN/DM and Newsroom corpus respectively. It contains words in our target training and test sets that occur at least twice. Experiments showed that using this subset of vocabulary words as opposed to over 320,000 vocabulary words contained in $gloVe$ improves both training time and performance of the model. During the abstractive training, we match summary sentence with its corresponding extracted document sentence using equation 6 and learn to minimize the seq2seq loss implemented in `tensorflow API`[4] with `AdamOptimizer` (Kingma and Ba, 2014). We employ early stopping when the validation loss does not decrease after 5 epochs. We apply gradient clipping at 5.0 (Pascanu et al., 2013). We use greedy-decoding during training and validation and set the maximum number of iterations to 5 times the target sentence length. Beam-search decoding is used during inference.

4.1 Datasets

We evaluate our models on the non-anonymized version of the CNN-DM corpus (Hermann et al., 2015; Nallapati et al., 2016) and the recent **Newsroom** dataset (Grusky et al., 2018) released by Connected Experiences Lab[5]. The Newsroom

[2] https://nlp.stanford.edu/projects/glove/

[3] https://github.com/tensorflow/tensor2tensor

[4] https://www.tensorflow.org/api_docs/python/tf/contrib/seq2seq/sequence_loss

[5] https://summari.es

Abstractive Model	R-1	R-2	R-L
RL+Intra-Att (Paulus et al., 2017)	41.16	15.75	**39.08**
KIGN+Pred (Li et al., 2018)	38.95	17.12	35.68
FAST (Chen and Bansal, 2018)	40.88	17.80	38.54
Bottom-Up (Gehrmann et al., 2018)	41.22	18.68	38.34
TRANS-ext + abs	41.05	17.87	36.73
TRANS-ext + filter +abs	**41.89**	**18.90**	38.92

Table 5: **ROUGE-F1** (%) scores (with 95% confidence interval) of various abstractive models on the **CNN/DM** test set.

Abstractive Model	R-1	R-2	R-L
Abs-N* (Rush et al., 2015)	5.88	0.39	5.32
Pointer* (See et al., 2017)	26.02	13.25	22.43
TRANS-ext + abs	33.81	15.37	28.92
TRANS-ext + filter + abs	**35.74**	**16.52**	**30.17**

Table 6: **ROUGE-F1** (%) scores (with 95% confidence interval) of various abstractive models on the **Newsroom** released test set. * marks results taken from Grusky et al. (2018)

corpus contains over 1.3M news articles together with various metadata information such as the title, summary, coverage and compression ratio. CNN/DM summaries are twice as long as Newsroom summaries with average word lengths of 66 and 26 respectively.

4.2 Evaluation

Following previous works (See et al., 2017; Nallapati et al., 2017; Chen and Bansal, 2018), we evaluate both datasets on standard ROUGE-1, ROUGE-2 and ROUGE-L (Lin, 2004). It calculates the appropriate n-gram word-overlap between the reference and system summaries.

4.3 Results Analysis

We used the official `pyrouge` script[6] with option[7]. Table 3 and 5 presents extractive and abstractive results on the CNN/DM dataset respectively, while Tables 4 and 6 for the Newsroom dataset. For clarity, we present results separately for each model and dataset.

Our baseline non-filtered extractive (TRANS-ext) model is highly competitive with top models. Our TRANS-ext + filter produces an average of about +1 and +9 points across reported ROUGE variants on the CNN/DM and Newsroom datasets respectively, showing that our model does a better job at identifying the most salient parts of the document than existing state-of-the-art extractive models. We observe the large margin in the Newsroom dataset results, as existing baselines are just the LEAD-3 and TEXTRANK of (Barrios et al., 2016). The Newsroom dataset was recently released and is yet to be thoroughly explored, however it is a larger dataset and contains more diverse summaries as analyzed by Grusky et al. (2018).

We also experimented with the empirical outcome of using imbalanced extractive labels which usually leads to bias towards the majority class. Interestingly, our extractive model has +20% F_Score increase when trained with balanced labels. Switching the transformer encoder with a seq2seq encoder, resulted in a drop of about 2 ROUGE points, showing that the transformer encoder does learn features that adds meaning to the vector representation of our input sequence.

Our baseline non-filtered abstractive (TRANS-ext + abs) model is also highly competitive with top models, with a drop of -0.81 ROUGE-2 points against Gehrmann et al. (2018)'s model which is the current state-of-the art. Our TRANS-ext + filter + abs produces an average of about +0.5 and +7 points across reported ROUGE variants on the CNN/DM and Newsroom datasets respectively, showing empirically that our model is an improvement of existing abstractive summarization models.

On the abstractiveness of our summaries, after aligning with the ground-truth as explained in Section 3.2 about 60% of our extracted document sentences were paraphrased and compressed.

[6] `https://github.com/andersjo/pyrouge/tree/master/tools/ROUGE-1.5.5`

[7] -n 2 -w 1.2 -m -a -c 95

O: the two clubs, who occupy the top two spots in spain's top flight, are set to face each other at the nou camp on sunday. **G:** real madrid face barcelona in the nou camp **R:** real madrid will travel to the nou camp to face barcelona on sunday.
O: dangelo conner, from new york, filmed himself messing around with the powerful weapon in a friend's apartment, first waving it around, then sending volts coursing through a coke can . **G:** dangelo conner from new york was fooling around with his gun **R:** dangelo conner, from new york ,was fooling around with stun gun.
O: jamie peacock broke his try drought with a double for leeds in their win over salford on sunday. **G:** jamie adam scored to win over salford for leeds **R:** jamie peacock scored two tries for leeds in their win over salford.
O: britain's lewis hamilton made the perfect start to his world title defense by winning the opening race of the f1 season in australia sunday to lead a mercedes one-two in melbourne . **G:** lewis hamilton wins first race of season in australia **R:** lewis hamilton wins opening race of 2015 f1 season in australia .

Table 7: Examples of some of our generated paraphrases from the CNN/DM dataset, where **O**, **G**, **R** represents Originating document sentence, our model's Generated paraphrase and Reference sentences from the ground-truth summary respectively.

We highlight examples of some of the generated paraphrases in Table 7. Table 7 show that our paraphrases are well formed, abstractive (*e.g powerful weapon – gun, messing around – fooling around*), capable of performing syntactic manipulations (*e.g for leeds in their win over sadford – win over salford for leeds*) and compression as seen in all the examples.

5 Related Work

Summarization has remained an interesting and important NLP task for years due to its diverse applications - news headline generation, weather forecasting, emails filtering, medical cases, recommendation systems, machine reading comprehension MRC and so forth (Khargharia et al., 2018).

Early summarization models were mostly extractive and manual-feature engineered (Knight and Marcu, 2000; Jing and McKeown, 2000; Dorr et al., 2003; Berg-Kirkpatrick et al., 2011). With the introduction of neural networks (Sutskever et al., 2014) and availability of large training data, deep learning became a viable approach (Rush et al., 2015; Chopra et al., 2016).

Extraction has been handled on different levels of granularity – word (Cheng and Lapata, 2016), phrases (Bui et al., 2016; Gehrmann et al., 2018), sentence (Cheng and Lapata, 2016; Nallapati et al., 2016, 2017) each with its challenges. Word and phrase level extraction although more concise usually suffers from grammatical incorrectness, while sentence-level extraction are too lengthy and sometimes contain redundant information. Hence Berg-Kirkpatrick et al. (2011); Filippova et al. (2015); Durrett et al. (2016) learn to extract and compress at sentence-level.

Identifying the likely most salient part of the text as summary-worthy is very crucial. Some authors have employed integer linear programming (Martins and Smith, 2009; Gillick and Favre, 2009; Boudin et al., 2015), graph concepts (Erkan and Radev, 2004; Parveen et al., 2015; Parveen and Strube, 2015), ranking with reinforcement learning (Narayan et al., 2018) and mostly related to our work – binary classification (Shen et al., 2007; Nallapati et al., 2017; Chen and Bansal, 2018)

Our binary classification architecture differs significantly from existing models because it uses a transformer as the building block instead of a bidirectional GRU-RNN (Nallapati et al., 2017), or bidirectional LSTM-RNN (Chen and Bansal, 2018). To the best of our knowledge, our utilization of the transformer encoder model as a building block for binary classification is novel, although the transformer has been successfully used for language understanding (Devlin et al., 2018), machine translation (MT) (Vaswani et al., 2017) and paraphrase generation (Zhao et al., 2018).

For generation of abstractive summaries, before the ubiquitous use of neural nets, manually crafted rules and graph techniques were utilized with considerable success. Barzilay and McKeown (2005); Cheung and Penn (2014) fused two sentences into one using their dependency parsed trees. Recently, sequence-to-sequence models (Sutskever et al., 2014) with attention (Bahdanau et al., 2014; Chopra et al., 2016), copy mechanism (Vinyals et al., 2015; Gu et al., 2016), pointer-generator (See et al., 2017), graph-based attention (Tan et al., 2017) have been explored. Since the system generated summaries are usually evaluated on ROUGE, its been beneficial to directly optimize this metric during training via a suitable policy using reinforcement learning (Paulus et al., 2017; Celikyilmaz et al., 2018).

Similar to Rush et al. (2015); Chen and Bansal (2018) we abstract by simplifying our extracted sentences. We jointly learn to paraphrase and compress, but different from existing models purely based on RNN, we implement a blend of two proven efficient models – transformer encoder and GRU-RNN. Zhao et al. (2018) paraphrased with a transformer-decoder, we find that using the GRU-RNN decoder but with a two-level stack of hybrid encoders (transformer and GRU-RNN) gives better performance. To the best of our knowledge, this architectural blend is novel.

6 Conclusion

We proposed two frameworks for extractive and abstractive summarization and demonstrated that they each improve results over existing state-of-the art. Our models are simple to train, and the intuition/hypothesis behind the formulation are straightforward and logical. The scientific correctness is provable, as parts of our model architecture have been used in other NLG-related tasks such as MT with state-of-the art results.

Acknowledgments

We would like to thank the anonymous reviewers for their useful comments. The research reported in this paper was conducted at the University of Lethbridge and supported by Alberta Innovates and Alberta Education.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *CoRR*.

Regina Barzilay and Kathleen R McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 481–490. Association for Computational Linguistics.

Florian Boudin, Hugo Mougard, and Benoit Favre. 2015. Concept-based summarization using integer linear programming: From concept pruning to multiple optimal solutions. In *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2015*.

Duy Duc An Bui, Guilherme Del Fiol, John F Hurdle, and Siddhartha Jonnalagadda. 2016. Extractive text summarization system to aid data extraction from full text in systematic review development. *Journal of biomedical informatics*, 64:265–272.

Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep communicating agents for abstractive summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1662–1675.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 484–494.

Jackie Chi Kit Cheung and Gerald Penn. 2014. Unsupervised sentence enhancement for automatic summarization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 775–786.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Bonnie Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL 03 on Text summarization workshop-Volume 5*, pages 1–8. Association for Computational Linguistics.

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-based single-document summarization with compression and anaphoricity constraints. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1998–2008.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Katja Filippova, Enrique Alfonseca, Carlos A Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence compression by deletion with lstms. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 360–368.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109.

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, pages 10–18. Association for Computational Linguistics.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1631–1640.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141.

Aishwarya Jadhav and Vaibhav Rajan. 2018. Extractive summarization with swap-net: Sentences and words from alternating pointer networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 142–151.

Hongyan Jing and Kathleen R McKeown. 2000. Cut and paste based text summarization. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 178–185. Association for Computational Linguistics.

Debabrata Khargharia, Nabajit Newar, and Nomi Baruah. 2018. Applications of text summarization. *International Journal of Advanced Research in Computer Science*, 9(3).

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization-step one: Sentence compression. *AAAI/IAAI*, 2000:703–710.

Chenliang Li, Weiran Xu, Si Li, and Sheng Gao. 2018. Guiding generation for abstractive text summarization based on key information guide network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 55–60.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*.

Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

André FT Martins and Noah A Smith. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, pages 1–9. Association for Computational Linguistics.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*, pages 3075–3081.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Ça glar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *CoNLL 2016*, page 280.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 1747–1759.

Shashi Narayan, Nikos Papasarantopoulos, Shay B Cohen, and Mirella Lapata. 2017. Neural extractive summarization with side information. *arXiv preprint arXiv:1704.04530*.

Mir Tafseer Nayeem and Yllias Chali. 2017. Extract with order for coherent multi-document summarization. In *Proceedings of TextGraphs-11: the Workshop on Graph-based Methods for Natural Language Processing*, pages 51–56.

Daraksha Parveen, Hans-Martin Ramsl, and Michael Strube. 2015. Topical coherence for graph-based extractive summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1949–1954.

Daraksha Parveen and Michael Strube. 2015. Integrating importance, non-redundancy and coherence in graph-based extractive summarization. In *IJCAI*, pages 1298–1304.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Dou Shen, Jian-Tao Sun, Hua Li, Qiang Yang, and Zheng Chen. 2007. Document summarization using conditional random fields. In *IJCAI*, volume 7, pages 2862–2867.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, et al. 2018. Tensor2tensor for neural machine translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Papers)*, volume 1, pages 193–199.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, pages 2692–2700. MIT Press.

Sanqiang Zhao, Rui Meng, Daqing He, Andi Saptono, and Bambang Parmanto. 2018. Integrating transformer and paraphrase rules for sentence simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3164–3173, Brussels, Belgium. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–663.