SEQUENCE-TO-SEQUENCE RNNS FOR TEXT SUMMARIZATION

Ramesh Nallapati, Bing Xiang & Bowen Zhou
Watson Question Answering Algorithms
IBM Watson
1011 Kitchawan Road, Yorktown Heights, NY 10598
{nallapati,bingxia,zhou}@us.ibm.com

ABSTRACT

In this work, we cast text summarization as a sequence-to-sequence problem and apply the attentional encoder-decoder RNN that has been shown to be successful for Machine Translation (Bahdanau et al. (2014)). Our experiments show that the proposed architecture significantly outperforms the state-of-the art model of Rush et al. (2015) on the Gigaword dataset without any additional tuning. We also propose additional extensions to the standard architecture, which we show contribute to further improvement in performance.

1 INTRODUCTION AND RELATED WORK

Deep learning based sequence-to-sequence models have been successful in many problems such as machine translation (Bahdanau et al. (2014)), speech recognition (Bahdanau et al. (2015)) and video captioning (Venugopalan et al. (2015)). In this work, we focus on the task of text summarization, which can also be naturally thought of as mapping an input sequence of words in a source document to a target sequence of words called summary. In the framework of sequence-to-sequence models, a very relevant model to our task is the attentional RNN encoder-decoder model proposed in Bahdanau et al. (2014), which has produced state-of-the-art performance in machine translation (MT). Since MT is also a task that maps one word-sequence to another, the attentional RNN encoder-decoder is a natural candidate for summarization too.

Despite the similarities, there are some key differences between summarization and machine translation. In summarization, the target (summary) is typically very short and does not depend very much on the length of the source (document). Additionally, a key challenge in summarization is to optimally compress the original document in a lossy manner such that the key concepts in the original document are preserved, unlike in MT, where the translation is expected to be loss-less. Hence, it remains to be tested whether the models that succeeded in machine translation would perform equally well here. In this work, we aim to answer precisely this question.

A vast majority of the past work in summarization has been extractive, which consists of identifying key sentences or passages in the source and reproducing them as summary (Erkan & Radev (2004)). There has also been some work on abstractive summarization using traditional machine translation based models (Banko et al. (2000)). In the framework of deep learning, the closest to our model is the recent work of Rush et al. (2015), in which, the authors use convolutional models to encode the source and a context-sensitive attentional feed-forward neural network to generate the summary, and they produced state-of-the-art results on the Gigaword and DUC datasets. For a more thorough comparison of past work, please refer to the aforementioned work.

2 MODELS, EXPERIMENTS AND RESULTS

In this work, we used the annotated Gigaword corpus as described in Rush et al. (2015). We used the scripts made available by the authors of this work[1] to preprocess the data, which resulted in about 3.8M training examples. We also made small modifications to the script to extract not only the tokenized words, but also system-generated parts-of-speech tags and named-entity tags. For our system, we used Kyunghyun Cho's Theano code [2] as the starting point.

[1] https://github.com/facebook/NAMAS

[2] https://github.com/kyunghyuncho/dl4mt-material

Training: For all the models we discuss below, we used 100-dimension word-embeddings pre-trained by the word2vec algorithm (Mikolov et al. (2013)) on a separate dataset comprising news stories, and we allowed the embeddings to be updated during training. The encoder consists of a bidirectional GRU-RNN (Chung et al. (2014)), each with a hidden state dimension of 200. The decoder consists of a uni-directional GRU-RNN with the same hidden-state size, an attention mechanism over the source-hidden states and a soft-max layer over target vocabulary to generate words. We kept the source and target vocabularies separate for computational efficiency, since the target vocabulary is much smaller. When we used only the first sentence of the document as the source, as done in Rush et al. (2015), the encoder vocabulary size was 119,505 and that of the decoder stood at 68,885. We used Adadelta (Zeiler (2012)) for training, with an initial learning rate of 0.001. We used a batch-size of 50 and randomly shuffled the training data at every epoch, while sorting every 10 batches according to their lengths to speed up training. We did not use any dropout or regularization, but we applied gradient clipping. We used early stopping based on the validation set and used the best model on the validation set to report all performance numbers. Other than this, we did not do any task specific fine-tuning.

Decoding and evaluation: At decode-time, we used beam search of size 5 to generate the summary, and limit the size of summary to a maximum of 30 words, since this is the maximum size we noticed in the sampled validation set. We report F1-scores from the *full-length* version of Rouge-1, Rouge-2 and Rouge-L using the official evaluation script [3], as well as the percentage of tokens in the system summary that occur in the source (which we call 'src. copy rate' in Table 1), on a randomly sampled subset of 2,000 examples from the unfiltered validation set.[4] We also compare the performance of our models on a randomly sampled subset of 2,000 examples from the test set. In addition, we also compare recall and F1 numbers of some of our models with those of the best system in Rush et al. (2015), on the exact test set used in their work. In the rest of the section, we describe various models we considered and report their performance.

Model #1: *words-1sent*: In our first encoder-decoder model, we used only the first sentence of the document as the source. The training time for this model on a single gpu was 13 hours per epoch, with convergence reaching in 19 epochs. We report the performance of this model on both validation and test sets in Table 1, in rows #1 and #9.

Model #2: *words-lvt2k-1sent*: In this variant, we use the large vocabulary trick (LVT) described in Jean et al. (2014). In our experiments, we restricted the decoder-vocabulary of each batch to words in the source documents of that batch and added the most frequent words in the target dictionary on top of these, until the vocabulary size reached a fixed size (we found 2,000 to be good enough in our validation experiments). We call this the LVT-vocabulary. The results show that this model achieves similar Rouge numbers as Model # 1, but not surprisingly, relies more on the source vocabulary as indicated in the last column of Table 1 . In the rest of the models described below, we persist with this trick because it cuts down the training time per epoch by nearly three times, and makes this and all subsequent models converge in only 50%-75% of the epochs needed for Model #1.

Models #3 and #4: *words-lvt2k-(2|5)sent*: These models are exactly same as Model #2, except for that they are trained on the first 2 and 5 sentences of the source document respectively. The table shows that the model trained on 2 sentences improves over the one trained on 1 sentence (Model #2) while the one trained on 5 sentences is not as good as the one trained on two. We therefore fixed the source length at 2 sentences for subsequent models. It is worth mentioning these models seem to borrow more words from the source than the previous models as indicated by the last column of the table, but this can be attributed to increase in source vocabulary size from using more sentences.

Model #5: *words-lvt2k-2sent-hier*: Since we used two sentences from source document, we implemented a hierarchical encoder with a second bi-directional RNN layer running at the sentence-level as described in Li et al. (2015), while retaining a single layer decoder with its attention operating over the top layer of the encoder. Unfortunately, this model did not produce any performance gains. As shown in the last column, this model relies somewhat too less on the source-vocabulary, perhaps owing to the coarser attention over sentence-level representations of the source.

[3] `http://www.berouge.com/Pages/default.aspx`

[4] The preprocessing script also produces a filtered validation set that is less noisy than the unfiltered set and is closer to the distribution of the training set, but we do not report numbers on it.

Model #6: *feats-lvt2k-2sent*: Here, we exploit the parts-of-speech and named-entity tags in the annotated gigaword corpus to augment the input embeddings on the source side. For each tag-type, we create an additional look-up table of embeddings corresponding to its own vocabulary, and for each token in the source we concatenate the embeddings of all its tags into a single vector. We repeat the process for also *tf* and *idf* weights of each token which we convert into one-hot representations by discretizing the real-values into one of 50 buckets each. In total, our embedding vector grew from the original 100 to 155. This simple extension has produced noticeable performance gain compared to its counterpart Model #3 as shown in Table 1 , demonstrating the utility of syntax based features in this task. This model also has the additional advantage of lower reliance on source vocabulary, indicating better abstractive ability.

Models #7 and #8: *(words|feats)-lvt2k-2sent-exp*: In these models, we use the corresponding (words|features)-lvt2k models, but augment the LVT-vocabulary with 1-nearest-neighbors of all words in the source document as measured by cosine similarity in the learned embeddings space. We do this before topping off the LVT-vocabulary with the most frequent words from the target dictionary, and make sure that the LVT-vocabulary size is still maintained at 2,000. The results in the table show that both models improve performance over their unexpanded counterparts on Rouge metrics as well as on source vocabulary reliance, with Model #8 emerging as the overall best system on two out of three Rouge metrics, and Model #7 taking the honors on Rouge-2. On the test set sampled by us, we see the same trend as in validation, where Model #8 outperforms Model #1 on two of the three Rouge metrics, as shown in rows #9 and #10.

Comparison with state-of-the-art: Rush et al. (2015) reported *recall-only* from *full-length* version of Rouge,[5] but the authors kindly provided us with their F1 numbers as well. In the full-length version of Rouge, we believe F1 is a better evaluation metric than recall because it allows us to compare on an equal footing, systems that produce different summary lengths. Recall, on the other hand, may unfairly reward systems that produce longer but noisy summaries. In our work, we hard-constrained our system summary length to a maximum of 30 words as specified earlier, and we found that the average system summary length from all our models (7.8 to 8.3) agrees very closely with that of the ground truth on the validation set (about 8.7 words), without any specific tuning. We obtained the exact test set of 1,951 examples used by Rush et al. (2015) and compared the performance of our Models #1 and Models #2 with their best system on both Recall as well as F1, as displayed in rows #11 through #16 in Table 1. [6] The table shows that our Models #1 and #2 significantly outperform the state of the art model of Rush et al. (2015), on both recall and F1.[7] In addition, our models also exhibit better abstractive ability as shown by the *src. copy rate* metric in the last column of the table.

3 CONCLUSION

We will display representative summaries from our models in the near future, but we would like to mention that even when they do not coincide with gold summaries, the system summaries are surprisingly good and would easily pass muster for a human generated summary in most cases. Our results strongly demonstrate that sequence-to-sequence models are extremely promising for summarization. Some of the other lessons we learned from our experiments are: (i) the LVT-trick is very useful for summarization as it improves training speed while not sacrificing performance; (ii) traditional methods such as vocabulary expansion and syntax-based features can boost performance of deep learning based models. As part of our ongoing work, we are investigating ways to effectively generate rare words in the summary, which appears to be a glaring weakness in the existing models.

ACKNOWLEDGMENTS

We thank Kyunghyun Cho for his easy-to-understand neural machine translation code and Alexander Rush for providing us with the test data and for helping us make as fair a comparison as possible with their system.

[5] based on personal communication with the authors.

[6] The reason we did not evaluate our best models on the test set provided by the authors of Rush et al. (2015) is that this set included neither the sentences subsequent to the first one in the source documents, nor the NLP annotations, either or both of which are needed by our Models #3 through #8.

[7] The recall numbers of Rush et al. (2015) displayed in row #11 of this table are slightly higher than those reported in the original paper because these are the latest numbers made available to us by the authors on the test set they shared with us.

#	Model name	Rouge-1	Rouge-2	Rouge-L	Src. copy rate (%)
Full-length F1 on validation set					
1	words-1sent	34.84	18.03	32.78	73.65
2	words-lvt2k-1sent	34.38	17.53	32.21	82.03
3	words-lvt2k-2sent	34.64	17.96	32.63	87.26
4	words-lvt2k-5sent	34.50	17.81	32.65	90.62
5	words-lvt2k-2sent-hier	30.71	13.25	28.70	**63.69**
6	feats-lvt2k-2sent	35.25	17.66	32.91	79.77
7	words-lvt2k-2sent-exp	34.99	**18.19**	32.99	75.76
8	feats-lvt2k-2sent-exp	**35.25**	17.98	**33.18**	76.85
Full length F1 on our internal test set					
9	Our baseline model (Model #1)	35.02	**17.71**	32.53	**76.75**
10	Our best model (Model #8)	**35.30**	17.58	**32.88**	77.11
Full length Recall on the test set used by Rush et al. (2015)					
11	SOTA (Rush et al. (2015))	31.47	12.73	28.54	91.50
12	Model #1	**34.85**	**17.20**	**32.74**	**74.57**
13	Model #2	34.02	16.04	31.67	85.53
Full length F1 on the test set used by Rush et al. (2015)					
14	SOTA (Rush et al. (2015))	29.78	11.89	26.97	91.50
15	Model #1	**32.76**	**16.17**	**30.73**	**74.57**
16	Model #2	32.01	15.07	29.75	85.53

Table 1: Performance comparison of various models. Please refer to Section 2 for explanation of notation.

REFERENCES

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014. URL http://arxiv.org/abs/1409.0473.

Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. End-to-end attention-based large vocabulary speech recognition. *CoRR*, abs/1508.04395, 2015. URL http://arxiv.org/abs/1508.04395.

Michele Banko, Vibhu O. Mittal, and Michael J Witbrock. Headline generation based on statistical translation. *In Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, 22:318–325, 2000.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014. URL http://arxiv.org/abs/1412.3555.

G. Erkan and D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479, 2004.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. On using very large target vocabulary for neural machine translation. *CoRR*, abs/1412.2007, 2014. URL http://arxiv.org/abs/1412.2007.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. A hierarchical neural autoencoder for paragraphs and documents. *CoRR*, abs/1506.01057, 2015. URL http://arxiv.org/abs/1506.01057.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013. URL http://arxiv.org/abs/1310.4546.

Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *CoRR*, abs/1509.00685, 2015. URL http://arxiv.org/abs/1509.00685.

Subhashini Venugopalan, Marcus Rohrbach, Jeff Donahue, Raymond J. Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence - video to text. *CoRR*, abs/1505.00487, 2015. URL http://arxiv.org/abs/1505.00487.

Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012. URL http://arxiv.org/abs/1212.5701.