Neural Headline Generation on Abstract Meaning Representation

Sho Takase[†] **Jun Suzuki**[‡] **Naoaki Okazaki**[†] **Tsutomu Hirao**[‡] **Masaaki Nagata**[‡]
Graduate School of Information Sciences, Tohoku University[†]
NTT Communication Science Laboratories, NTT Corporation[‡]
{takase, okazaki}@ecei.tohoku.ac.jp
{suzuki.jun, hirao.tsutomu, nagata.masaaki}@lab.ntt.co.jp

Abstract

Neural network-based encoder-decoder models are among recent attractive methodologies for tackling natural language generation tasks. This paper investigates the usefulness of structural syntactic and semantic information additionally incorporated in a baseline neural attention-based model. We encode results obtained from an abstract meaning representation (AMR) parser using a modified version of Tree-LSTM. Our proposed attention-based AMR encoder-decoder model improves headline generation benchmarks compared with the baseline neural attention-based model.

1 Introduction

Neural network-based encoder-decoder models are cutting-edge methodologies for tackling natural language generation (NLG) tasks, *i.e.*, machine translation (Cho et al., 2014), image captioning (Vinyals et al., 2015), video description (Venugopalan et al., 2015), and headline generation (Rush et al., 2015).

This paper also shares a similar goal and motivation to previous work: improving the encoder-decoder models for natural language generation. There are several directions for enhancement. This paper respects the fact that NLP researchers have expended an enormous amount of effort to develop fundamental NLP techniques such as POS tagging, dependency parsing, named entity recognition, and semantic role labeling. Intuitively, this structural, syntactic, and semantic information underlying input text has the potential for improving the quality of NLG tasks. However, to the best of our knowledge, there is no clear evidence that syntactic and semantic information can enhance the recently developed encoder-decoder models in NLG tasks.

To answer this research question, this paper proposes and evaluates a headline generation method based on an encoder-decoder architecture on *Abstract Meaning Representation (AMR)*. The method is essentially an extension of *attention-based summarization (ABS)* (Rush et al., 2015). Our proposed method encodes results obtained from an AMR parser by using a modified version of Tree-LSTM encoder (Tai et al., 2015) as additional information of the baseline ABS model. Conceptually, the reason for using AMR for headline generation is that information presented in AMR, such as predicate-argument structures and named entities, can be effective clues when producing shorter summaries (headlines) from original longer sentences. We expect that the quality of headlines will improve with this reasonable combination (ABS and AMR).

2 Attention-based summarization (ABS)

ABS proposed in Rush et al. (2015) has achieved state-of-the-art performance on the benchmark data of headline generation including the DUC-2004 dataset (Over et al., 2007). Figure 1 illustrates the model structure of ABS. The model predicts a word sequence (summary) based on the combination of the neural network language model and an input sentence encoder.

Let V be a vocabulary. $\boldsymbol{x}_i$ is the i-th indicator vector corresponding to the i-th word in the input sentence. Suppose we have M words of an input sentence. $\boldsymbol{X}$ represents an input sentence, which

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 1054–1059,
Austin, Texas, November 1-5, 2016.

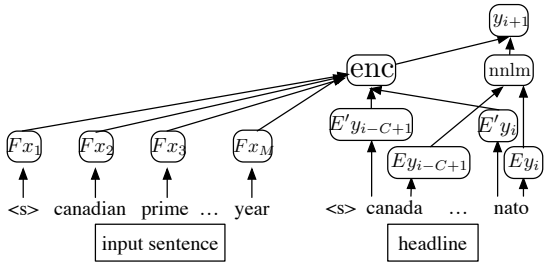

Figure 1: Model structure of 'attention-based summarization (ABS)'.

is represented as a sequence of indicator vectors, whose length is M. That is, $\boldsymbol{x}_i \in \{0,1\}^{|V|}$, and $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M)$. Similarly, let $\boldsymbol{Y}$ represent a sequence of indicator vectors $\boldsymbol{Y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_L)$, whose length is L. Here, we assume $L < M$. $\boldsymbol{Y}_{C,i}$ is a short notation of the list of vectors, which consists of the sub-sequence in $\boldsymbol{Y}$ from $\boldsymbol{y}_{i-C+1}$ to $\boldsymbol{y}_i$. We assume a one-hot vector for a special start symbol, such as "$\langle$S$\rangle$", when $i < 1$. Then, ABS outputs a summary $\hat{\boldsymbol{Y}}$ given an input sentence $\boldsymbol{X}$ as follows:

$$\hat{\boldsymbol{Y}} = \arg\max_{\boldsymbol{Y}} \Big\{ \log p(\boldsymbol{Y}|\boldsymbol{X}) \Big\}, \quad (1)$$

$$\log p(\boldsymbol{Y}|\boldsymbol{X}) \approx \sum_{i=0}^{L-1} \log p(\boldsymbol{y}_{i+1}|\boldsymbol{X}, \boldsymbol{Y}_{C,i}), \quad (2)$$

$$p(\boldsymbol{y}_{i+1}|\boldsymbol{X}, \boldsymbol{Y}_{C,i}) \propto \exp\big(\mathrm{nnlm}(\boldsymbol{Y}_{C,i}) + \mathrm{enc}(\boldsymbol{X}, \boldsymbol{Y}_{C,i})\big), \quad (3)$$

where $\mathrm{nnlm}(\boldsymbol{Y}_{C,i})$ is a feed-forward neural network language model proposed in (Bengio et al., 2003), and $\mathrm{enc}(\boldsymbol{X}, \boldsymbol{Y}_{C,i})$ is an input sentence encoder with attention mechanism.

This paper uses D and H as denoting sizes (dimensions) of vectors for word embedding and hidden layer, respectively. Let $\boldsymbol{E} \in \mathbb{R}^{D\times|V|}$ be an embedding matrix of output words. Moreover, let $\boldsymbol{U} \in \mathbb{R}^{H\times(CD)}$ and $\boldsymbol{O} \in \mathbb{R}^{|V|\times H}$ be weight matrices of hidden and output layers, respectively[1]. Using the above notations, $\mathrm{nnlm}(\boldsymbol{Y}_{C,i})$ in Equation 3 can be written as follows:

$$\mathrm{nnlm}(\boldsymbol{Y}_{C,i}) = \boldsymbol{O}\boldsymbol{h}, \quad \boldsymbol{h} = \tanh(\boldsymbol{U}\tilde{\boldsymbol{y}}_c), \quad (4)$$

[1]Following Rush et al. (2015), we omit bias terms throughout the paper for readability, though each weight matrix also has a bias term.

where $\tilde{\boldsymbol{y}}_c$ is a concatenation of output embedding vectors from $i - C + 1$ to i, that is, $\tilde{\boldsymbol{y}}_c = (\boldsymbol{E}\boldsymbol{y}_{i-C+1} \cdots \boldsymbol{E}\boldsymbol{y}_i)$. Therefore, $\tilde{\boldsymbol{y}}_c$ is a (CD) dimensional vector.

Next, $\boldsymbol{F} \in \mathbb{R}^{D\times|V|}$ and $\boldsymbol{E}' \in \mathbb{R}^{D\times|V|}$ denote embedding matrices of input and output words, respectively. $\boldsymbol{O}' \in \mathbb{R}^{|V|\times D}$ is a weight matrix for the output layer. $\boldsymbol{P} \in \mathbb{R}^{D\times(CD)}$ is a weight matrix for mapping embedding of C output words onto embedding of input words. $\tilde{\boldsymbol{X}}$ is a matrix form of a list of input embeddings, namely, $\tilde{\boldsymbol{X}} = \big[\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_M\big]$, where $\tilde{\boldsymbol{x}}_i = \boldsymbol{F}\boldsymbol{x}_i$. Then, $\mathrm{enc}(\boldsymbol{X}, \boldsymbol{Y}_{C,i})$ is defined as the following equations:

$$\mathrm{enc}(\boldsymbol{X}, \boldsymbol{Y}_{C,i}) = \boldsymbol{O}'\bar{\boldsymbol{X}}\boldsymbol{p}, \quad (5)$$

$$\boldsymbol{p} \propto \exp(\tilde{\boldsymbol{X}}^{\mathrm{T}}\boldsymbol{P}\tilde{\boldsymbol{y}}'_c), \quad (6)$$

where $\tilde{\boldsymbol{y}}'_c$ is a concatenation of output embedding vectors from $i - C + 1$ to i similar to $\tilde{\boldsymbol{y}}_c$, that is, $\tilde{\boldsymbol{y}}'_c = (\boldsymbol{E}'\boldsymbol{y}_{i-C+1} \cdots \boldsymbol{E}'\boldsymbol{y}_i)$. Moreover, $\bar{\boldsymbol{X}}$ is a matrix form of a list of averaged input word embeddings within window size Q, namely, $\bar{\boldsymbol{X}} = [\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_M]$, where $\bar{\boldsymbol{x}}_i = \sum_{q=i-Q}^{i+Q} \frac{1}{Q}\tilde{\boldsymbol{x}}_q$.

Equation 6 is generally referred to as the attention model, which is introduced to encode a relationship between input words and the previous C output words. For example, if the previous C output words are assumed to align to $\boldsymbol{x}_i$, then the surrounding Q words $(\boldsymbol{x}_{i-Q}, \ldots, \boldsymbol{x}_{i+Q})$ are highly weighted by Equation 5.

3 Proposed Method

Our assumption here is that syntactic and semantic features of an input sentence can greatly help for generating a headline. For example, the meanings of subjects, predicates, and objects in a generated headline should correspond to the ones appearing in an input sentence. Thus, we incorporate syntactic and semantic features into the framework of headline generation. This paper uses an AMR as a case study of the additional features.

3.1 AMR

An AMR is a rooted, directed, acyclic graph that encodes the meaning of a sentence. Nodes in an AMR graph represent 'concepts', and directed edges represent a relationship between nodes. Concepts

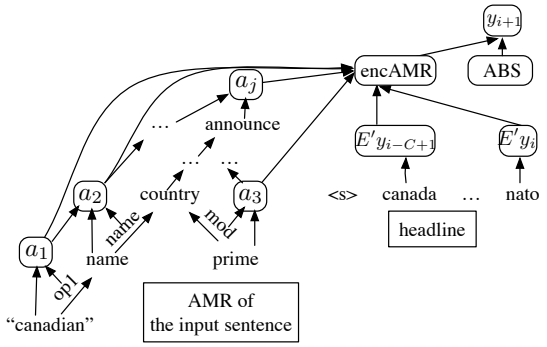

Figure 2: Model structure of our proposed attention-based AMR encoder; it outputs a headline using ABS and encoded AMR with attention.

consist of English words, PropBank event predicates, and special labels such as "person". For edges, AMR has approximately 100 relations (Banarescu et al., 2013) including semantic roles based on the PropBank annotations in OntoNotes (Hovy et al., 2006). To acquire AMRs for input sentences, we use the state-of-the-art transition-based AMR parser (Wang et al., 2015).

3.2 Attention-Based AMR Encoder

Figure 2 shows a brief sketch of the model structure of our attention-based AMR encoder model. We utilize a variant of child-sum Tree-LSTM originally proposed in (Tai et al., 2015) to encode syntactic and semantic information obtained from output of the AMR parser into certain fixed-length embedding vectors. To simplify the computation, we transform a DAG structure of AMR parser output to a tree structure, which we refer to as "*tree-converted AMR structure*". This transformation can be performed by separating multiple head nodes, which often appear for representing coreferential concepts, to a corresponding number of out-edges to head nodes. Then, we straightforwardly modify Tree-LSTM to also encode edge labels since AMR provides both node and edge labels, and original Tree-LSTM only encodes node labels.

Let $\boldsymbol{n}_j$ and $\boldsymbol{e}_j$ be N and E dimensional embeddings for labels assigned to the j-th node, and the out-edge directed to its parent node[2]. $\boldsymbol{W}_{in}$, $\boldsymbol{W}_{fn}$, $\boldsymbol{W}_{on}$, $\boldsymbol{W}_{un} \in \mathbb{R}^{D\times N}$ are weight matrices for node embeddings $\boldsymbol{n}_j$[3]. Similarly, $\boldsymbol{W}_{ie}$, $\boldsymbol{W}_{fe}$, $\boldsymbol{W}_{oe}$, $\boldsymbol{W}_{ue} \in \mathbb{R}^{D\times E}$ are weight matrices for edge embeddings $\boldsymbol{e}_j$. $\boldsymbol{W}_{ih}$, $\boldsymbol{W}_{fh}$, $\boldsymbol{W}_{oh}$, $\boldsymbol{W}_{uh} \in \mathbb{R}^{D\times D}$ are weight matrices for output vectors connected from child nodes. $B(j)$ represents a set of nodes, which have a direct edge to the j-th node in our tree-converted AMR structure. Then, we define embedding $\boldsymbol{a}_j$ obtained at node j in tree-converted AMR structure via Tree-LSTM as follows:

$$\tilde{\boldsymbol{h}}_j = \sum_{k\in B(j)} \boldsymbol{a}_k, \tag{7}$$

$$\boldsymbol{i}_j = \sigma\Big(\boldsymbol{W}_{in}\boldsymbol{n}_j + \boldsymbol{W}_{ie}\boldsymbol{e}_j + \boldsymbol{W}_{ih}\tilde{\boldsymbol{h}}_j\Big), \tag{8}$$

$$\boldsymbol{f}_{jk} = \sigma\Big(\boldsymbol{W}_{fn}\boldsymbol{n}_j + \boldsymbol{W}_{fe}\boldsymbol{e}_j + \boldsymbol{W}_{fh}\boldsymbol{a}_k\Big), \tag{9}$$

$$\boldsymbol{o}_j = \sigma\Big(\boldsymbol{W}_{on}\boldsymbol{n}_j + \boldsymbol{W}_{oe}\boldsymbol{e}_j + \boldsymbol{W}_{oh}\tilde{\boldsymbol{h}}_j\Big), \tag{10}$$

$$\boldsymbol{u}_j = \tanh\Big(\boldsymbol{W}_{un}\boldsymbol{n}_j + \boldsymbol{W}_{ue}\boldsymbol{e}_j + \boldsymbol{W}_{uh}\tilde{\boldsymbol{h}}_j\Big), \tag{11}$$

$$\boldsymbol{c}_j = \boldsymbol{i}_j \odot \boldsymbol{u}_j \sum_{k\in B(j)} \boldsymbol{f}_{jk} \odot \boldsymbol{c}_k, \tag{12}$$

$$\boldsymbol{a}_j = \boldsymbol{o}_j \odot \tanh(\boldsymbol{c}_j). \tag{13}$$

Let J represent the number of nodes in tree-converted AMR structure obtained from a given input sentence. We introduce $\boldsymbol{A} \in \mathbb{R}^{D\times J}$ as a matrix form of a list of hidden states $\boldsymbol{a}_j$ for all j, namely, $\boldsymbol{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_J]$. Let $\boldsymbol{O}'' \in \mathbb{R}^{|V|\times D}$ be a weight matrix for the output layer. Let $S \in \mathbb{R}^{D\times(CD)}$ be a weight matrix for mapping the context embedding of C output words onto embeddings obtained from nodes. Then, we define the attention-based AMR encoder '$\mathrm{encAMR}(\boldsymbol{A}, \boldsymbol{Y}_{C,i})$' as follows:

$$\mathrm{encAMR}(\boldsymbol{A}, \boldsymbol{Y}_{C,i}) = \boldsymbol{O}''\boldsymbol{A}\boldsymbol{s}, \tag{14}$$

$$\boldsymbol{s} \propto \exp(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{S}\tilde{\boldsymbol{y}}'_c). \tag{15}$$

Finally, we combine our attention-based AMR encoder shown in Equation 14 as an additional term of Equation 3 to build our headline generation system.

4 Experiments

To demonstrate the effectiveness of our proposed method, we conducted experiments on benchmark data of the abstractive headline generation task described in Rush et al. (2015).

[2]We prepare a special edge embedding for a root node.

[3]As with Equation 4, all the bias terms are omitted, though each weight matrix has one.

	DUC-2004			Gigaword test data used in (Rush et al., 2015)			Gigaword Our sampled test data		
Method	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
ABS (Rush et al., 2015)	26.55	7.06	22.05	30.88	12.22	27.77	–	–	–
ABS (re-run)	28.05	7.38	23.15	31.26	12.46	28.25	32.93	13.43	29.80
ABS+AMR	***28.80**	***7.83**	***23.62**	**31.64**	***12.94**	**28.54**	***33.43**	***13.93**	**30.20**
ABS+AMR(w/o attn)	28.28	7.21	23.12	30.89	12.40	27.94	31.32	12.83	28.46

Table 1: Results of methods on each dataset. We marked $*$ on the ABS+AMR results if we observed statistical difference ($p < 0.05$) between ABS (re-run) and ABS+AMR on the t-test. (R-1: ROUGE-1, R-2: ROUGE-2, R-L: ROUGE-L)

For a fair comparison, we followed their evaluation setting. The training data was obtained from the first sentence and the headline of a document in the annotated Gigaword corpus (Napoles et al., 2012)[4]. The development data is DUC-2003 data, and test data are both DUC-2004 (Over et al., 2007) and sentence-headline pairs obtained from the annotated Gigaword corpus as well as training data[5]. All of the generated headlines were evaluated by ROUGE (Lin, 2004)[6]. For evaluation on DUC-2004, we removed strings after 75-characters for each generated headline as described in the DUC-2004 evaluation. For evaluation on Gigaword, we forced the system outputs to be at most 8 words as in Rush et al. (2015) since the average length of headline in Gigaword is 8.3 words. For the preprocessing for all data, all letters were converted to lower case, all digits were replaced with '#', and words appearing less than five times with 'UNK'. Note that, for further evaluation, we prepared 2,000 sentence-headline pairs randomly sampled from the test data section of the Gigaword corpus as our additional test data.

In our experiments, we refer to the baseline neural attention-based abstractive summarization method described in Rush et al. (2015) as "**ABS**", and our proposed method of incorporating AMR structural information by a neural encoder to the baseline method described in Section 3 as "**ABS+AMR**". Additionally, we also evaluated the performance of the AMR encoder *without* the attention mechanism, which we refer to as "**ABS+AMR(w/o attn)**", to investigate the contribution of the attention mechanism on the AMR encoder. For the parameter estimation (training), we used stochastic gradient descent to learn parameters. We tried several values for the initial learning rate, and selected the value that achieved the best performance for each method. We decayed the learning rate by half if the log-likelihood on the validation set did not improve for an epoch. Hyper-parameters we selected were $D = 200$, $H = 400$, $N = 200$, $E = 50$, $C = 5$, and $Q = 2$. We re-normalized the embedding after each epoch (Hinton et al., 2012).

For ABS+AMR, we used the two-step training scheme to accelerate the training speed. The first phase learns the parameters of the ABS. The second phase trains the parameters of the AMR encoder by using 1 million training pairs while the parameters of the baseline ABS were fixed and unchanged to prevent overfitting.

Table 1 shows the recall of ROUGE (Lin, 2004) on each dataset. ABS (re-run) represents the performance of ABS re-trained by the distributed scripts[7]. We can see that the proposed method, ABS+AMR, outperforms the baseline ABS on all datasets. In particular, ABS+AMR achieved statistically significant gain from ABS (re-run) for ROUGE-1 and ROUGE-2 on DUC-2004. However in contrast, we observed that the improvements on Gigaword (the same test data as Rush et al. (2015)) seem to be limited compared with the DUC-2004 dataset. We assume that this limited gain is caused largely by the quality of AMR parsing results. This means that the

[4]Training data can be obtained by using the script distributed by the authors of Rush et al. (2015).

[5]Gigaword test data can be obtained from `https://github.com/harvardnlp/sent-summary`

[6]We used the ROUGE-1.5.5 script with option "`-n2 -m -b75 -d`", and computed the average of each ROUGE score.

[7]`https://github.com/facebook/NAMAS`

I(1): crown prince abdallah ibn abdel aziz left saturday at the head of saudi arabia 's delegation to the islamic summit in islamabad , the official news agency spa reported .
G: saudi crown prince leaves for islamic summit
A: crown prince leaves for islamic summit in saudi arabia
P: saudi crown prince leaves for islamic summit in riyadh

I(2): a massive gothic revival building once christened the lunatic asylum west of the <unk> was auctioned off for $ #.# million -lrb- euro# .# million -rrb- .
G: massive ##th century us mental hospital fetches $ #.# million at auction
A: west african art sells for $ #.# million in
P: west african art auctioned off for $ #.# million

I(3): brooklyn , the new bastion of cool for many new yorkers , is poised to go mainstream chic .
G: high-end retailers are scouting sites in brooklyn
A: new yorkers are poised to go mainstream with chic
P: new york city is poised to go mainstream chic

Figure 3: Examples of generated headlines on Gigaword. **I**: input, **G**: true headline, **A**: ABS (re-run), and **P**: ABS+AMR.

Gigaword test data provided by Rush et al. (2015) is already pre-processed. Therefore, the quality of the AMR parsing results seems relatively worse on this pre-processed data since, for example, many low-occurrence words in the data were already replaced with "UNK". To provide evidence of this assumption, we also evaluated the performance on our randomly selected 2,000 sentence-headline test data also taken from the test data section of the annotated Gigaword corpus. "Gigaword (randomly sampled)" in Table 1 shows the results of this setting. We found the statistical difference between ABS(re-run) and ABS+AMR on ROUGE-1 and ROUGE-2.

We can also observe that ABS+AMR achieved the best ROUGE-1 scores on all of the test data. According to this fact, ABS+AMR tends to successfully yield semantically important words. In other words, embeddings encoded through the AMR encoder are useful for capturing important concepts in input sentences. Figure 3 supports this observation. For example, ABS+AMR successfully added the correct modifier 'saudi' to "crown prince" in the first example. Moreover, ABS+AMR generated a consistent subject in the third example.

The comparison between ABS+AMR(w/o attn) and ABS+AMR (with attention) suggests that the attention mechanism is necessary for AMR encoding. In other words, the encoder without the attention mechanism tends to be overfitting.

5 Related Work

Recently, the Recurrent Neural Network (RNN) and its variant have been applied successfully to various NLP tasks. For headline generation tasks, Chopra et al. (2016) exploited the RNN decoder (and its variant) with the attention mechanism instead of the method of Rush et al. (2015): the combination of the feed-forward neural network language model and attention-based sentence encoder. Nallapati et al. (2016) also adapted the RNN encoder-decoder with attention for headline generation tasks. Moreover, they made some efforts such as hierarchical attention to improve the performance. In addition to using a variant of RNN, Gulcehre et al. (2016) proposed a method to handle infrequent words in natural language generation. Note that these recent developments do not conflict with our method using the AMR encoder. This is because the AMR encoder can be straightforwardly incorporated into their methods as we have done in this paper, incorporating the AMR encoder into the baseline. We believe that our AMR encoder can possibly further improve the performance of their methods. We will test that hypothesis in future study.

6 Conclusion

This paper mainly discussed the usefulness of incorporating structural syntactic and semantic information into novel attention-based encoder-decoder models on headline generation tasks. We selected abstract meaning representation (AMR) as syntactic and semantic information, and proposed an attention-based AMR encoder-decoder model. The experimental results of headline generation benchmark data showed that our attention-based AMR encoder-decoder model successfully improved standard automatic evaluation measures of headline generation tasks, ROUGE-1, ROUGE-2, and ROUGE-L. We believe that our results provide empirical evidence that syntactic and semantic information obtained from an automatic parser can help to improve the neural encoder-decoder approach in NLG tasks.

Acknowledgments

We thank the anonymous reviewers for their insightful comments and suggestions.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *The Journal of Machine Learning Research*, 3:1137–1155.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1724–1734.

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 (NAACL-HLT 2016)*, pages 93–98.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the Unknown Words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 140–149.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving Neural Networks by Preventing Co-adaptation of Feature Detectors. *CoRR*, abs/1207.0580.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% Solution. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2006)*, pages 57–60.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out: Proceedings of the Association for Computational Linguistics Workshop*, pages 74–81.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL 2016)*, pages 280–290.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated Gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 95–100.

Paul Over, Hoa Dang, and Donna Harman. 2007. DUC in Context. *Information Processing and Management*, 43(6):1506–1520.

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 (EMNLP 2015)*, pages 379–389.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2015)*, pages 1556–1566.

Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. 2015. Translating Videos to Natural Language Using Deep Recurrent Neural Networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2015)*, pages 1494–1504.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and Tell: A Neural Image Caption Generator. In *Proceedings of the Computer Vision and Pattern Recognition (CVPR 2015)*, pages 3156–3164.

Chuan Wang, Nianwen Xue, and Sameer Pradhan. 2015. A Transition-based Algorithm for AMR Parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2015)*, pages 366–375.