Convolutional Encoder Approach to Sentence Simplification

Yotam Manne [*1] **Guy Azov** [*1]

Abstract

Sentence simplification aims to simplify the content and structure of complex sentences, and thus make them easier to interpret for human readers, and easier to process for downstream NLP applications. In this paper, we adapt an architecture of Encoder-Decoder model presented by (Gehring et al., 2016). This model was originally developed for Neural Machine Translation and achieved good results. Due to the similarity of NMT to sentence simplification, we think that this adaption is a natural step in the research of the task.

1. Introduction

The goal of sentence simplification is to convert complex sentences into simpler ones so that they are more understandable and accessible, while still keeping their original information content and meaning. Sentence simplification has a number of practical applications: it is useful for bilingual education and other language-learning contexts. It can help patients with linguistic and cognitive disabilities (Carroll et al., 1999). Sentence simplification can also be used to improve performance in other NLP tasks ((Niklaus et al., 2017); (Chandrasekar et al., 1996);(Beigman Klebanov et al., 2004). Convolutional neural networks (CNN) utilize layers with convolving filters that are applied to local features. Originally invented for computer vision, CNN models have subsequently been shown to be effective for NLP and have achieved excellent results in semantic parsing (Yih et al., 2014), search query retrieval (Shen et al., 2014), sentence modeling (Kalchbrenner et al., 2014), and other traditional NLP tasks. In this paper we wish to answer the question whether replacing the classic RNN encoder of a seq2seq model with a convolutional encoder can yield better results in terms of BLEU(Papineni et al., 2002) and SARI(Xu et al., 2016) scores of the output sentences.

[*]Equal contribution [1]Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv, Israel. Correspondence to: Yotam Manne <yotammanne@mail.tau.ac.il>, Guy Azov <guyazov@mail.tau.ac.il>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

2. Related Work

In previous studies, researchers of sentence-level simplification mostly address the simplification task as a machine translation problem. (Specia, 2010) use statistical machine translation approach implemented in Moses toolkit (Koehn et al., 2007) to translate the original sentences to the simplified ones. (Wang et al., 2016) were the first to suggest using a NMT model for text simplification. They used a LSTM encoder - decoder seq2seq model, but due to the lack of an adequate dataset they used a number-based sequences instead of natural language data. (Coster & Kauchak, 2011) introduced a new dataset of aligned sentence pairs taken from Wikipedia and Simple English Wikipedia, the dataset is widely used in many sentence simplification researches, and set the ground for new and better datasets to be created. (Zhang et al., 2017) suggested a constrained seq2seq neural model for sentence simplification, their model combines world level and sentence level simplifications and yields better results than various baselines. (Meng et al., 2015) proposed using a convolutional neural network to encode the source language for NMT. Our work is based on the model that was presented by (Gehring et al., 2016) for NMT, which uses two convolutional neural networks as an encoder, and an attention based recurrent neural network as the decoder.

3. Our Approach

We chose to adapt a NMT model to the sentence simplification task. Most of the seq2seq neural models we encountered were based on RNN encoder – decoder, however we decided to encode the source sentences with a Convolutional Neural Network instead. (Gehring et al., 2016) used a similar approach for NMT. (Di Palo & Parde, 2019) tried it too for sentence classification. But as far as we know, we are the first to try this architecture for sentence simplification. CNNs computation, contrary to RNNs, can be parallelized, optimization is easier since the number of non-linearities is fixed and independent of the input length and last because they outperform the LSTM accuracy in (Wu et al., 2016).

4. Model (3)

4.1. Encoder Architecture

One of the challenges of using CNNs encoders is the loss of word ordering. In order to solve it, (Gehring et al., 2016) proposes to use position embeddings in addition to the pre-trained word embeddings. See table 1. Let u_j be the j^{th} word in the source sentence, w_j it's word embedding and l_j it's position embedding, then:

$$e_j = l_j + w_j$$

As suggested by (Gehring et al., 2016) The encoder consists of two stacked convolutional networks: CNN-a's output z_j used for creating the attention matrix A that is used at decoding time. Simultaneously, CNN-c's output z'_j is used to produce the conditional input c_i by a simple dot product between the attention vector a_i with it.

$$z_j = CNN_a(\mathbf{e})_j,\ z'_j = CNN_c(\mathbf{e})_j$$

The CNNs do not contain pooling layers which are commonly used for down-sampling, i.e., the full source sequence length will be retained after the networks has been applied. (Gehring et al., 2016) shows best results when CNN-a contains 2-3 times more layers than CNN-c.

Word	Position	Representation
we	1	WordEmbedding(we) + PositionEmbedding(1)
need	2	WordEmbedding(need) + PositionEmbedding(2)
a	3	WordEmbedding(a) + PositionEmbedding(3)
vacation	4	WordEmbedding(vacation) + PositionEmbedding(4)

Table 1. Embedding of a full sentence

4.2. Decoder Architecture

4.2.1. Preliminaries

- h_i denotes the hidden state/output of the LSTM.
- c_i denotes the conditional input to the LSTM.
- g_i denotes the embedding of the previous output of the LSTM. This gets concatenated with c_i as input to the LSTM

4.2.2. Attention (1)

The attention mechanism is based on the outputs of CNN-c z'_j. At time step i the conditional input c_i is computed via a dot product attention mechanism (Luong et al., 2015). We transform the decoder hidden state h_i by a linear layer with weights W_d and b_d to match the size of the embedding of the previous target word g_i and then sum the two representations to yield d_i:

$$d_i = W_d h_i + b_d + g_i$$

Next, we generate the attention matrix A as follows:

$$a_{ij} = \frac{exp\left(d_i^T z_j\right)}{\sum_{t=1}^{m} exp\left(d_i^T z_t\right)}$$

Instead of generating a_{ij} individually, we can generate the entire $\mathbf{a_i}$ in one go, by modifying the equation slightly:

$$\mathbf{a_i} = \text{softmax}(d_i^T \mathbf{z})$$

Finally, we generate c_i as:

$$c_i = \sum_{j=1}^{m} a_{ij} z'_j$$

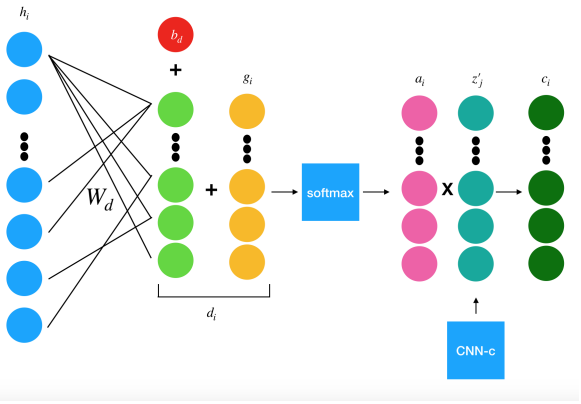

Figure 1. The dot product attention mechanism

4.2.3. The Decoder(2)

We use LSTMs (Hochreiter & Schmidhuber, 1997) for the decoder network whose state s_i comprises of a cell vector and a hidden vector h_i which is output by the LSTM at each time step. We concatenate c_i and g_i, and feed them into the LSTM . The decoder output h_{i+1} is transfromed by a linear layer with weights W_o and bias b_o to the target vocabulary size V, then a softmax layer is applied to create a distribution over all possible words. The most probable word will be selected as the decoder's output y_{i+1}.

$$y_{i+1} = argmax(softmax(W_o h_{i+1} + b_o))$$

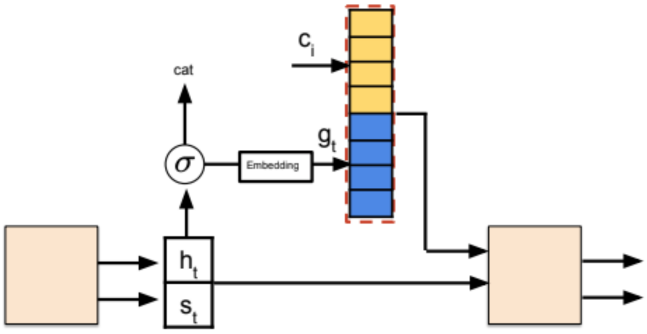

Figure 2. Block diagram of the Decoder flow and architecture

5. Experimental Setup

5.1. Datasets

5.1.1. Simple English Wikipedia (Coster & Kauchak, 2011)

A sentence aligned dataset taken from parallel articles in English Wikipedia and Simple English Wikipedia. This dataset contains 167K pairs of sentences and is one of the largest datasets used for sentence simplification. While examining this dataset we noticed a few problems – Many sentences contain special characters, URLs, gibberish, excess use of punctuation and more. Such anomalies can corrupt the training procedure and cause unreliable results.

5.1.2. Newsela (Xu et al., 2015)

A simplification corpus of news articles, re-written by professional editors to meet the readability standards for children at multiple grade levels. Each sentence in the corpus is rewritten in up to 6 different level of complexity. The creators of this dataset mapped all the problems that exist in the Simple Wikipedia corpus and addressed them in their research. The Newsela dataset contains 141K pairs of aligned sentences. Our model supports both datasets but because of the problems we mentioned above we used the Newsela corpus for training and evaluation.

5.2. Data Preprocessing

To use the data we needed some pre-processing. Two aligned lists of sentences were constructed from the raw data. From each list a vocabulary which maps each word to a unique integer ID was created. Using the mentioned vocabularies, every sentence was converted to a list of word IDs. Each tokenized sentence is fed later as input to our model, which uses GloVe embeddings (Pennington et al., 2014) to represent each word in lower dimensional space.

5.3. Experiments

5.3.1. Control Experiment

Since we didn't find any similar models that were tested on the sentence simplification task, we wanted first to get some intuition about the expected results. Therefore we implemented a 'classic' LSTM encoder – decoder model and trained it with our dataset. We saw that until the end of the training the loss value was constantly decreasing, which suggests that the model is indeed learning as expected. Nevertheless, when examining the "control model" 's output over the evaluation set we noticed that the predicted sentences are mostly grammatically correct, but has no contextual relation whatsoever to the input sentence:

Example:

- Source: Japanese-American troops were once put in their own units
- Predicted: Suddenly enemy fighters attacked the patrol

While gaining some intuition from the control experiment, it's results were irrelevant to make any conclusions, so we decided to move on to train and test the main model.

5.3.2. Main Experiment

First, for sanity check, we forced the model to overfit over one (small) batch. Once successful, we tuned the model's parameters according to (Gehring et al., 2016) and (Neishi et al., 2017). We set the number of layers in CNN-a and CNN-c to 15 and 5 respectively and made residual connection between layers. Each layer consists of 512 hidden units and kernel size of 3. Our decoder consists of a 4 layers bidirectional-LSTM network, with 512 hidden units in each layer. We used a batch size of 256, and the optimization was done by Adam optimizer (Kingma & Ba, 2014) with a learning rate of 0.001. To improve learning, we added a Teacher Forcing mechanism (Williams & Zipser, 1989) with probability 0.5. We limited our scope to sentences up to length 10, in order to focus on the model's correctness rather than dealing with phenomena related to long sequences. The model was implemented with Pytorch, and training and evaluation were conducted with a single Nvidia Geforce RTX 2080 GPU.

5.4. Results

State-of-the-art results in sentence simplification are measured per corpus with two main evaluation metrics:

- BLEU (Papineni et al., 2002)

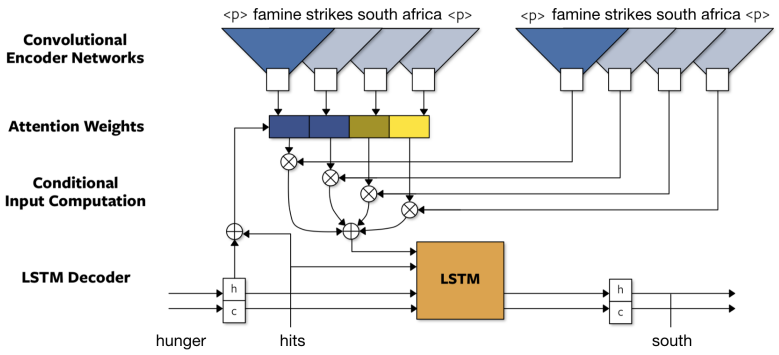

Figure 3. Model architecture, adapted from (Gehring et al., 2016)

- SARI (Xu et al., 2016)

Table 2 presents the models that achieved state-of-the-art results to date. Unfortunately, all our attempts to reach sane results failed badly. All the output sentences consisted of a single word repeating multiple times:

- Source sentence: ”california to help students not fluent in english .”
- Target sentence: ”english language learners get extra help in school”
- Predicted sentence: ”the the the the the the the the the”

The loss curve (4) indicated that the model didn't converge. We found it irrelevant to evaluate such results with BLEU or SARI and compare them with state-of-the-art.

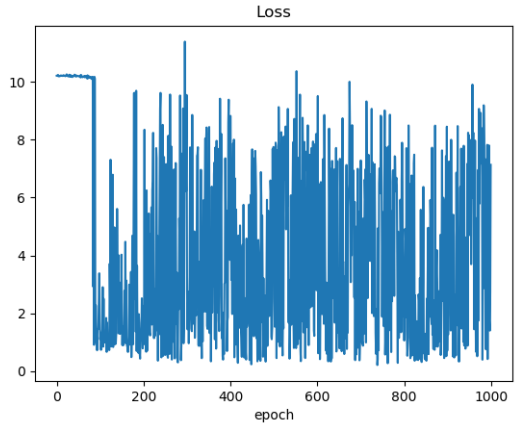

Figure 4. Loss curve of 1000 epoches of training

Model	BLEU	SARI
Pointer + Multi-task Entailment and Paraphrase Generation (Guo et al., 2018)	11.14	33.22
Hybrid (Narayan & Gardent, 2014)	14.46	30.00
NSELSTM-S ((Vu et al., 2018)	22.62	29.58

Table 2. State-of-the-art sentence simplification results over the Newsela corpus to date.

6. Conclusions and Future Work

We believe that good results are achievable with our model. We noticed that reaching zero loss in the overfitting test took a great amount of epochs. This might suggest that we were underfitting in the main experiment and that more runtime with stronger hardware can help. (Kriz et al., 2019) suggested a custom loss function dedicated to sentence simplification that can be integrated and help improve results. We used greedy decoding both in train and evaluation time, however, a beam search algorithm will probably be more accurate. While examining the data we saw that some source sentences differ from their simple version only by 1-2 words. We think that a copy mechanism like the one (Mathews et al., 2018) suggested can also increase the model's accuracy. Due to the lack of convergence of the original model we couldn't proceed to implementing the improvements above. Although it's not a great start in the world of research, the lessons we've learned during this project are priceless. We obtained first intuition of designing and testing a new ML model, and gained valuable experience with Keras and Pytorch libraries. No doubt that in future projects these foundations will help us achieve successful results.

References

Beigman Klebanov, B., Knight, K., and Marcu, D. Text simplification for information-seeking applications. In Meersman, R. and Tari, Z. (eds.), *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE*, pp. 735–747, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

Carroll, J., Minnen, G., Pearce, D., Canning, Y., Devlin, S., and Tait, J. Simplifying text for language-impaired readers. In *In Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (EACL*, pp. 269–270, 1999.

Chandrasekar, R., Doran, C., and Srinivas, B. Motivations and methods for text simplification. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 2*, COLING '96, pp. 1041–1044, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics.

Coster, W. and Kauchak, D. Simple English Wikipedia: A new text simplification task. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 665–669, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

Di Palo, F. and Parde, N. Enriching neural models with targeted features for dementia detection. *arXiv preprint arXiv:1906.05483*, 2019.

Gehring, J., Auli, M., Grangier, D., and Dauphin, Y. N. A convolutional encoder model for neural machine translation. *CoRR*, abs/1611.02344, 2016.

Guo, H., Pasunuru, R., and Bansal, M. Dynamic multi-level multi-task learning for sentence simplification. *arXiv preprint arXiv:1806.07304*, 2018.

Hochreiter, S. and Schmidhuber, J. Lstm can solve hard long time lag problems. In *Advances in neural information processing systems*, pp. 473–479, 1997.

Kalchbrenner, N., Grefenstette, E., and Blunsom, P. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pp. 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Kriz, R., Sedoc, J., Apidianaki, M., Zheng, C., Kumar, G., Miltsakaki, E., and Callison-Burch, C. Complexity-weighted loss and diverse reranking for sentence simplification. *arXiv preprint arXiv:1904.02767*, 2019.

Luong, M.-T., Pham, H., and Manning, C. D. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

Mathews, A., Xie, L., and He, X. Simplifying sentences with sequence to sequence models. *arXiv preprint arXiv:1805.05557*, 2018.

Meng, F., Lu, Z., Wang, M., Li, H., Jiang, W., and Liu, Q. Encoding source language with convolutional neural network for machine translation. *arXiv preprint arXiv:1503.01838*, 2015.

Narayan, S. and Gardent, C. Hybrid simplification using deep semantics and machine translation. 2014.

Neishi, M., Sakuma, J., Tohda, S., Ishiwatari, S., Yoshinaga, N., and Toyoda, M. A bag of useful tricks for practical neural machine translation: Embedding layer initialization and large batch size. In *Proceedings of the 4th Workshop on Asian Translation (WAT2017)*, pp. 99–109, 2017.

Niklaus, C., Bermeitinger, B., Handschuh, S., and Freitas, A. A sentence simplification system for improving relation extraction. *arXiv preprint arXiv:1703.09013*, 2017.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318. Association for Computational Linguistics, 2002.

Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

Shen, Y., He, X., Gao, J., Deng, L., and Mesnil, G. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*, pp. 373–374. ACM, 2014.

Specia, L. Translating from complex to simplified sentences. In Pardo, T. A. S., Branco, A., Klautau, A., Vieira, R., and de Lima, V. L. S. (eds.), *Computational Processing of the Portuguese Language*, pp. 30–39, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

Vu, T., Hu, B., Munkhdalai, T., and Yu, H. Sentence simplification with memory-augmented neural networks. *arXiv preprint arXiv:1804.07445*, 2018.

Wang, T., Chen, P., Rochford, J., and Qiang, J. Text simplification using neural machine translation. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

Williams, R. J. and Zipser, D. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989. doi: 10.1162/neco.1989.1.2.270.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., and Dean, J. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.

Xu, W., Callison-Burch, C., and Napoles, C. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297, 2015.

Xu, W., Napoles, C., Pavlick, E., Chen, Q., and Callison-Burch, C. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415, 2016.

Yih, W.-t., He, X., and Meek, C. Semantic parsing for single-relation question answering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 643–648, 2014.

Zhang, Y., Ye, Z., Feng, Y., Zhao, D., and Yan, R. A constrained sequence-to-sequence neural model for sentence simplification. *CoRR*, abs/1704.02312, 2017.