Using Semantic Similarity as Reward for Reinforcement Learning in Sentence Generation

Go Yasui[1] **Yoshimasa Tsuruoka**[1] **Masaaki Nagata**[2]

[1]The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
{gyasui,tsuruoka}@logos.t.u-tokyo.ac.jp

[2]NTT Communication Science Laboratories, NTT Corporation
2-4 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0237 Japan
masaaki.nagata.et@hco.ntt.co.jp

Abstract

Traditional model training for sentence generation employs cross-entropy loss as the loss function. While cross-entropy loss has convenient properties for supervised learning, it is unable to evaluate sentences as a whole, and lacks flexibility. We present the approach of training the generation model using the estimated semantic similarity between the output and reference sentences to alleviate the problems faced by the training with cross-entropy loss. We use the BERT-based scorer fine-tuned to the Semantic Textual Similarity (STS) task for semantic similarity estimation, and train the model with the estimated scores through reinforcement learning (RL). Our experiments show that reinforcement learning with semantic similarity reward improves the BLEU scores from the baseline LSTM NMT model.

1 Introduction

Sentence generation using neural networks has become a vital part of various natural language processing tasks including machine translation (Sutskever et al., 2014) and abstractive summarization (Rush et al., 2015). Most previous work on sentence generation employ cross-entropy loss between the model outputs and the ground-truth sentence to guide the maximum-likelihood training on the token-level. Differentiability of cross-entropy loss is useful for computing gradients in supervised learning; however, it lacks flexibility and may penalize the generation model for a slight shift or change in token sequence even if the sequence retains the meaning.

For instance, consider the sentence pair, "I watched a movie last night." and "I saw a film last night.". As the simple cross-entropy loss lacks the ability to properly assess semantically similar tokens, these sentences are penalized for having two token mismatches. As another example, the sentence pair "He often walked to school." and "He walked to school often." would be severely punished by the token misalignment, despite having identical meanings.

To tackle the inflexible nature of model evaluation during training, we propose an approach of using semantic similarity between the output sequence and the ground-truth sequence to train the generation model. In the proposed framework, semantic similarity of sentence pairs is estimated by a BERT-based (Devlin et al., 2018) regression model fine-tuned against Semantic Textual Similarity (Agirre et al., 2012) dataset, and the resulting score is passed back to the model using reinforcement learning strategies.

Our experiment on translation datasets suggests that the proposed method is better at improving the BLEU score than the traditional cross-entropy learning. However, since the model outputs had limited paraphrastic variations, the results are also inconclusive in supporting the effectiveness of applying the proposed method to sentence generation.

2 Related Work

2.1 Sentence Generation

Recurrent neural networks have become popular models of choice for sentence generation (Sutskever et al., 2014). These sentence generation models are generally implemented as an architecture known as an Encoder-Decoder model.

The decoder model, the portion of Encoder-

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, pages 400–406
Florence, Italy, July 28 - August 2, 2019.

Decoder responsible for generating tokens, is usually an RNN. For an intermediate representation X, output token distribution at time t $\hat{y}_t$ for the RNN decoder π_θ can be written as

$$s_{t+1} = \Phi_\theta \left(\hat{y}_t, s_t, X \right) \quad (1)$$

$$\hat{y}_{t+1} \sim \pi_\theta \left(y_t \mid \hat{y}_t, s_t, X \right) \quad (2)$$

where s_t is the hidden state of the decoder at time t, Φ_θ is the state update function, and θ is the model parameter. Since a simple RNN is known to lack the ability to handle long-term dependencies, recurrent models with more sophisticated update mechanisms such as Long Short-term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gate Recurrent Unit (GRU) (Cho et al., 2014) are used in more recent works.

Sentence generation models are typically trained using cross-entropy loss as follows:

$$L_{CE} = -\sum_{t=1}^{T} \log \pi_\theta \left(y_t \mid y_{t-1}, s_t, X \right), \quad (3)$$

where $Y = \{y_1, y_2, ..., y_T\}$ is the ground-truth sequence.

While cross-entropy loss is an effective loss function for multi-class classification problems such as sentence generation, there are a few drawbacks. Cross-entropy loss is computed by comparing the output distribution and the target distribution on every timestep, and this token-wise nature is intolerant of slight shift or reordering in output tokens. As the ground-truth distributions Y are usually one-hot distributions cross-entropy loss is also intolerant to distribution mismatch even when the two distributions represent similar but different tokens.

2.2 Reinforcement Learning for Sentence Generation

One way to avoid the problems of cross-entropy loss is to use a different criterion during the model training. Reinforcement learning, a framework in which the agent must choose a series of discrete actions to maximize the reward returned from its surrounding environment, is one of such approaches. The advantages of using RL are that the reward for an action does not have to be returned spontaneously and that the reward function does not have to be differentiable by the parameter of the agent model.

Because of these advantages, RL has often been used as a means to train sentence generation model against sentence-level metrics (Pasunuru and Bansal, 2018; Ranzato et al., 2015). Sentence-level metrics commonly used in RL settings, such as BLEU, ROUGE and METEOR, are typically not differentiable, and thus are not usable under the regular supervised training.

One of the common RL algorithms used in sentence generation is REINFORCE (Williams, 1992). REINFORCE is a relatively simple policy gradient algorithm. In the context of sentence generation, the goal of the agent is to maximize the expectation of the reward provided as the function r as in the following:

$$Maximize\, \mathbb{E}_{\hat{y}_1, ..., \hat{y}_T \sim \pi_\theta(\hat{y}_1, ..., \hat{y}_T)} \left[r \left(\hat{y}_1, ..., \hat{y}_T \right) \right], \quad (4)$$

where $\hat{Y} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_T\}$ is a series of decoder output tokens.

The loss function is the negative of the reward expectation, but the expectation is typically approximated by a single sample sequence as follows:

$$L_{RL} = \sum_t \log \pi_\theta \left(y_t \mid \hat{y}_{t-1}, s_t \right) \left(r \left(\hat{y}_{1,...,T} \right) - r_b \right), \quad (5)$$

where r_b is the baseline reward which counters the large variance of reward caused by sampling. r_b can be any function that does not contain the parameter of the sentence generation model, but usually is kept to a simple model or function to not hinder the training.

2.3 Semantic Textual Similarity

Semantic Textual Similarity (STS) (Agirre et al., 2012; Cer et al., 2017) is an NLP task of evaluating the degree of similarity between two given texts. Similarity scores must be given as continuous real values from 0 (completely dissimilar) and 5 (completely equivalent), and the model performance is measured by computing the Pearson correlation between the machine score and the human score. As STS scores are assigned as similarity scores between whole sentences and not tokens, slight token differences can lower the STS score drastically. For example, the first sentence pair shown in Table 1, “A man is playing a guitar.” and “A girl is playing a guitar.”, only has a single token mismatch, “man” and “girl”. However, the score given to the pair is 2.8, because that single mismatch causes clear contrasts in meanings between the sentences.

Table 1: Examples of STS similarity scores in STS-B dataset.

Score	Sentence Pair
2.8	A man is playing a guitar. A girl is playing a guitar.
4.2	A panda bear is eating some bamboo. A panda is eating bamboo.

On the other hand, STS scores are tolerant of modifications that do not change the meaning of sentence. This leniency is illustrated by the second sentence pair in Table 1, "A panda bear is eating some bamboo." and "A panda is eating bamboo.". Such a sentence pair would receive an unfavourable score in similarity evaluation using token-wise comparison, because every word after "panda" would be considered as a mismatched token. In contrast, the STS score given to the pair is 4.2. Omission of words "bear" and "some" in the latter sentence does not alter the meaning from the first sentence, and thus the pair is considered semantically similar.

STS is similar to other semantic comparison tasks such as textual entailment (Dagan et al., 2010) and paraphrase identification (Dolan et al., 2004). One key distinction that STS has from these two tasks is that STS expects the model to output continuous scores with interpretable intermediate values rather than discrete binary values describing whether or not given sentence pairs have certain semantic relationships.

2.4 BERT

Bidirectional Encoder Representations from Transformer (BERT) (Devlin et al., 2018) is a pre-training model based on the transformer model (Vaswani et al., 2017). Previous pre-training models such as ELMo (Peters et al., 2017) and OpenAI-GPT (Radford et al., 2018) used unidirectional language models to learn general language representations and this limited their ability to capture token relationships in both directions. Instead, BERT employs a bidirectional self-attention architecture to capture the language representations more thoroughly.

Upon its release, BERT broke numerous state-of-the-art records such as those on a general language understanding task GLUE (Wang et al., 2018), question answering task SQuAD v1.1 (Rajpurkar et al., 2016), and grounded commonsense inference task SWAG (Zellers et al., 2018). STS is one of the tasks included in GLUE.

3 Models

3.1 Sentence Generation Model

The sentence generation model π_θ used for this research is a neural machine translation (NMT) model consisting of a single-layer LSTM encoder-decoder model with attention mechanism and the softmax output layer. The model also incorporates input feeding to make itself aware of the alignment decision in the previous decoding step (Luong et al., 2015). The encoder LSTM is bidirectional while the decoder LSTM is unidirectional.

3.2 STS Estimator

The STS estimator model r_ψ consists of two modules. As described in Eq. (6), one is the BERT encoder with pooling layer B and the other is a linear output layer (with weight vector W_ψ and bias b_ψ) with ReLU activation r_ψ.

$$B\left(Y_1, Y_2\right) = \mathrm{Pool}\left(\mathrm{BERT}\left(Y_1, Y_2\right)\right), \tag{6}$$

$$r_\psi\left(Y_1, Y_2\right) = \mathrm{ReLU}\left(W_\psi \cdot B\left(Y_1, Y_2\right) + b_\psi\right). \tag{7}$$

The BERT encoder reads tokenized sentence pairs (Y_1, Y_2) joined by a separation (SEP) token and outputs intermediate representations that are then fed into the linear layer through a pooling layer. The output layer projects the input into scalar values representing the estimated STS scores for input sentence pairs.

The model r_ψ is trained using the mean squared error (MSE) to fit the corresponding real-valued label v as written in Eq. (8).

$$L_{BERT} = \left|r_\psi\left(Y_1, Y_2\right) - v\right|^2. \tag{8}$$

While the use of the BERT-based STS estimator as an evaluation mechanism allows the sentence generation model to train its outputs against sentence-wise evaluation criteria, there is a downside to this framework.

The BERT encoder expects the input sentences to be sequences of tokens. As with most sentence generation models, the outputs of the encoder-decoder model described in the previous subsection are sequences of output probability distributions of tokens.

Obtaining a single token from a probability distribution equates to performing indifferentiable operations like argmax and sampling. Consequently, the regular backpropagation algorithm cannot be applied the training of generation model. Furthermore, the scores provided by the STS estimator r_ψ are sentence-wise while the sequence generation is done token by token. There is no direct way to evaluate the effect of a single instance of token generation on a sentence-wise outcome in the setting of supervised learning. As mentioned in Section 2.2, RL is an approach that can provide solutions to these problems.

3.3 Baseline Estimator

Following the previous work (Ranzato et al., 2015), the baseline estimator Ω_ω is defined as follows:

$$\Omega_\omega(s_t) = \sigma(W_\omega \cdot s_t + b_\omega), \tag{9}$$

where W_r is a weight vector, b_ω is a bias, and σ is the logistic sigmoid function.

3.4 Model Training

Overall, the model training is separated into three stages.

The first stage is the training of BERT-based STS estimator r_ψ. The model r_ψ, with its pretrained BERT encoder, is fine-tuned using a STS dataset with the loss function described in Eq. (8). The parameter of the STS estimator is frozen from this point onward.

The second stage is the training of the NMT model using the cross-entropy loss shown in Eq. (3). This stage is necessary to allow the model training to converge. The action space in sentence generation is extremely large and applying RL from scratch would lead to slow and unstable training.

The final stage is the RL stage where we apply REINFORCE to NMT model. The loss function for REINFORCE is rewritten from Eq. (5) as follows:

$$L_{RL} = \sum_t R_t \log \pi_\theta(y_t \mid \hat{y}_{t-1}, s_t), \tag{10}$$

$$R_t = \left(\frac{1}{5} r_\psi\left(\hat{Y}, Y\right) - \Omega_\omega(s_t)\right), \tag{11}$$

where R_t is the difference between the reward r_ψ and the expected reward Ω_ω. r_ψ is multiplied by $\frac{1}{5}$ as Ω_ω is bounded in $[0, 1]$. Because using only L_{RL} in the RL stage reportedly leads to unstable training (Wu et al., 2016) the loss used in this step is a linear combination of L_{CE} and L_{RL} as follows:

$$L = \lambda L_{CE} + (1 - \lambda) L_{RL}, \tag{12}$$

where $\lambda \in [0, 1]$ is a hyperparameter. The value of λ typically is a small non-zero value.

During the RL stage, the reward prediction model Ω_ω is trained using the MSE loss as follows:

$$L_{BSE} = \left|\frac{1}{5} r_\psi\left(\hat{Y}, Y\right) - \Omega_\omega(s_t)\right|^2. \tag{13}$$

The reward predictor does not share its parameter with the NMT model.

4 Experiment

4.1 Dataset

The dataset used for fine-tuning the STS estimator is STS-B (Cer et al., 2017). The tokenizer used is a wordpiece tokenizer for BERT.

For machine translation, we used De-En parallel corpora from multi30k-dataset (Elliott et al., 2016) and WIT3 (Cettolo et al., 2012). The multi30k-dataset is comprised of textual descriptions of images while the WIT3 consists of transcribed TED talks. Each corpus provides a single validation set and multiple test sets. We chose the best models based on their scores for the validation sets and used the two newest test sets from each corpus for testing. Both corpora are tokenized using the sentencepiece BPE tokenizer with a vocabulary size of 8,000 for each language. All letters are turned to lowercase and any consecutive spaces are turned into a single space before tokenization. The source and target vocabularies are kept separate.

4.2 Training Settings

The BERT model used for the experiment is BERT-base-uncased, and is trained with a maximum sequence length of 128, batch size of 32, learning rate of 2×10^{-5} up to 6 epochs.

For the supervised (cross-entropy) training of the NMT model, we set size of hidden states for all LSTM to 256 for each direction, and use SGD with an initial learning rate of 1.0, momentum of 0.75, the learning rate decay of 0.5, and the dropout rate of 0.2. With the batch size of 128 and the maximum sequence length of 100, the

NMT model typically reached the highest estimated STS score on the validation set after less than 10 epochs.

In the RL stage, initial learning rates are set to 0.01 and 1.0×10^{-3} for the NMT model and the baseline estimator model respectively. λ is set to 0.005. The batch size is reduced to 100 but other hyperparameters are kept the same as in the supervised stage.

For a comparison, we also train a separate translation model with RL using GLEU (Wu et al., 2016). GLEU score is calculated by taking the minimum of n-gram recall and n-gram precision between output tokens and target tokens. While the GLEU score is known to correlate well with the BLEU score on the corpus-level, it also avoids some of the undesirable characteristics that the BLEU score has on the sentence-level. During the RL stage for the GLEU model, the reward measure $\frac{1}{5} r_\psi \left(\hat{Y}, Y \right)$ in Eq. (11) and Eq. (13) is replaced by $GLEU \left(\hat{Y}, Y \right)$. Other training procedures and hyperparameters are kept the same as those of the model trained using STS.

5 Results and Discussion

The BLEU scores of Cross-entropy, RL-GLEU and RL-STS models are shown in Table 2 and the sample outputs of the models during the training are displayed in Table 3.

As shown in Table 2 applying the RL step with STS improved BLEU scores for all test sets, even though the model was not directly optimized to increase the BLEU score. It can be inferred that estimated semantic similarity scores have positive correlation with the BLEU score.

As BLEU is scored using matching n-grams between the candidate and ground-truth sentences, it can be considered a better indicator of semantic similarity between sentences than cross-entropy loss. One interesting observation made during the training was that after entering the RL stage, the cross-entropy loss against the training data increased yet the BLEU scores improved. This suggests that RL using STS reward is a better training strategy for improving the semantic accuracy of output tokens than the plain cross-entropy loss training.

Table 2 also shows that RL-GLEU has better BLEU scores than RL-STS. This is inevitable considering that STS, unlike GLEU and BLEU, is not based on n-gram matching and may permit output tokens not present in a target sequence as long as the output sequence stays semantically similar to the target sequence. Such property can lead to n-gram mismatches and lower BLEU scores. It is important to note that the leniency of STS evaluation does not severely affect BLEU scores.

In fact, training with RL using STS did alter outputs of the model in ways that suggest the leniency of STS as a training objective. For instance, sentences shown in Table 3 demonstrate the cases where the RL swapped a few tokens or added an extra token to the output sentences without drastically changing the meaning of the original sentence.

Nevertheless, this kind of alterations were not abundant perhaps because of the fact that the model is never encouraged to output paraphrastic sentences during the supervised learning phase. The degree of effectiveness of our approach would be more apparent in the setting where the model outputs are more diverse, such as paraphrasing.

Another interesting characteristic of the outputs of RL-STS is that they sometimes did not properly terminate. This occurred even in cases where the cross-entropy model was able to form a complete sentence. One possible cause of this problem is the way the output sequence is tokenized before it is fed to the BERT-based estimator. Because an end-of-sentence (EOS) token is not one of the special tokens used in pretraining of BERT, any EOS token was stripped before inserting a SEP token. Consequently, the RL-STS model was not able to receive proper feedback for producing the EOS token. This can perhaps be avoided by introducing an additional loss term in Eq. (10) to penalize sequences that are not terminated.

6 Conclusion

In this paper, we focused on the disadvantages of using cross-entropy loss for sentence generation, namely its inability to handle similar tokens and its intolerance towards token reordering. To solve these problems, we proposed an approach of using the BERT-based semantic similarity estimator trained using STS dataset to evaluate the degree of meaning overlap between output sentences and ground-truth sentences. As the estimated STS scores are indifferentiable, we also incorporated REINFORCE into the training to backpropagate the gradient using RL strategies. The proposed

Table 2: BLEU and estimated STS scores for test sets in multi30k-dataset and WIT3. mscoco2017 and flickr2017 are test sets for multi30k-dataset, while TED2014 and TED2015 are test sets for WIT3. RL-GLEU and RL-STS denote models trained with REINFORCE using GLEU reward and STS reward respectively.

	mscoco2017		flickr2017		TED2014		TED2015	
Model	BLEU	STS	BLEU	STS	BLEU	STS	BLEU	STS
Cross-entropy	16.44	2.76	22.22	3.03	12.54	2.63	13.43	2.80
RL-GLEU	20.13	2.93	25.83	3.15	13.97	2.71	14.59	2.89
RL-STS	18.31	2.96	24.70	3.21	13.58	2.87	14.56	2.99

Table 3: Sample outputs of the models for the training set

Model	Output Sentences	
Ground-truth	I'll show you what I mean.	So how do we solve?
Cross-entropy	I'll show you what I mean.	So how do we solve?
RL-GLEU	I'll show you what I mean.	So how do we solve?
RL-STS	I'm going to show you what I mean.	So how do we solve problems?

method proved successful in improving the BLEU score over the baseline model trained using only the cross-entropy loss. The findings from the comparison of model outputs suggest that the STS allows lenient evaluation without severely degrading BLEU scores. However, the extent of effectiveness of the proposed method is yet to be determined. Further analysis of the method using different datasets such as those for abstractive summarization and paraphrasing, as well as human evaluation are necessary to reach a proper conclusion.

Acknowledgments

We would like to thank Kazuma Hashimoto and anonymous reviewers for helpful comments and suggestions.

References

Eneko Agirre, Mona Diab, Daniel Cer, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *SemEval*.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation. In *SemEval*.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit^3: Web inventory of transcribed and translated talks. In *EAMT*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN EncoderDecoder for Statistical Machine Translation. In *EMNLP*.

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2010. Recognizing textual entailment: Rational, evaluation and approaches–erratum. *Natural Language Engineering*, 16(1).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING*.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. 2016. Multi30k: Multilingual english-german image descriptions. In *VL*.

Sepp Hochreiter and Jrgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *EMNLP*.

Ramakanth Pasunuru and Mohit Bansal. 2018. Multi-reward reinforced summarization with saliency and entailment. In *NAACL*.

Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *ACL*.

Alec Radford, Karthik Narasimhan, Time Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, Technical report, OpenAI.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence Level Training with Recurrent Neural Networks. *arXiv:1511.06732 [cs]*. ArXiv: 1511.06732.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In *NIPS*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *EMNLP Workshop BlackboxNLP*.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, ukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. *arXiv preprint arXiv:1808.05326*.