Generating Diverse Translations with Sentence Codes

Raphael Shu
The University of Tokyo
shu@nlab.ci.i.u-tokyo.ac.jp

Hideki Nakayama
The University of Tokyo
nakayama@ci.i.u-tokyo.ac.jp

Kyunghyun Cho
New York University
CIFAR Azrieli Global Scholar
kyunghyun.cho@nyu.edu

Abstract

Users of machine translation systems may desire to obtain multiple candidates translated in different ways. In this work, we attempt to obtain diverse translations by using sentence codes to condition the sentence generation. We describe two methods to extract the codes, either with or without the help of syntax information. For diverse generation, we sample multiple candidates, each of which conditioned on a unique code. Experiments show that the sampled translations have much higher diversity scores when using reasonable sentence codes, where the translation quality is still on par with the baselines even under strong constraint imposed by the codes. In qualitative analysis, we show that our method is able to generate paraphrase translations with drastically different structures. The proposed approach can be easily adopted to existing translation systems as no modification to the model is required.

1 Introduction

When using machine translation systems, users may desire to see different candidate translations other than the best one. In this scenario, users usually expect the system to show candidates with different sentence structures.

To obtain diverse translations, conventional neural machine translation (NMT) models allow one to sample translations using the beam search algorithm, however, they usually share similar sentence structures. Recently, various methods (Li et al., 2016; Xu et al., 2018) are proposed for diverse generation. These methods encourage the model to use creative vocabulary to achieve high diversity. Although producing creative words benefits tasks in the dialog domain, when applied to machine translation, it can hurt the translation quality by changing the original meaning.

In this work, we are interested in generating multiple valid translations with high diversity. To achieve this, we propose to construct the codes based on semantics-level or syntax-level information of target-side sentences.

To generate diverse translations, we constrain the generation model by specifying a particular code as a semantic or syntactic assignment. More concretely, we prefix the target-side sentences with the codes. Then, an NMT model is trained with the original source sentences and the prefixed target sentences. As the model generates tokens in left-to-right order, the probability of emitting each word is predicted conditioned on the assigned code. As each assignment is supposed to correspond to a sentence structure, the candidate translations sampled with different assignments are expected to have high diversity.

We can think such model as a mixture-of-expert translation model where each expert is capable of producing translations with a certain style indicated by the code. In the inference time, code assignments are given to the model so that a selection of experts are picked to generate translations.

The key question is how to extract such sentence codes. Here, we explore two approaches. First, a simple unsupervised method is tested, which clusters the sentence embeddings and use the cluster ids as the code assignments. Next, to capture only the structural variation of sentences, we turn to syntax. We encode the structure of constituent parse trees into discrete codes with a tree autoencoder.

Experiments on two machine translation datasets show that a set of highly diverse translations can be obtained with reasonable mechanism for extracting the sentence codes, while the sampled candidates still have BLEU scores on par with the baselines.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1823–1827
Florence, Italy, July 28 - August 2, 2019.

2 Proposed Approach

2.1 Extracting Sentence Codes

Our approach produces diverse translations by conditioning sentence generation with the sentence codes. Ideally, we would like the codes to capture the information about the sentence structures rather than utterances. To extract such codes from target sentences, we explore two methods.

Semantic Coding Model The first method extracts sentence codes from unsupervisedly learned semantic information. We cluster the sentence embeddings produced by pre-trained models into a fixed number of clusters, then use the cluster ids as discrete priors to condition the sentence generation. In this work, we test two semantic coding models. The first model is based on BERT (Devlin et al., 2018), where the vectors corresponding to the "[CLS]" token are clustered.

The second model produces sentence embeddings by averaging FastText word embeddings (Bojanowski et al., 2017). Comparing to the hidden states of BERT, word embeddings are expected to contain less syntactic information as the word order is ignored during training.

Syntactic Coding Model To explicitly capture the syntactic diversity, we also consider to derive the sentence codes from the parse trees produced by a constituency parser. As the utterance-level information is not desired, the terminal nodes are removed from the parse trees.

To obtain the sentence codes, we use a TreeLSTM-based auto-encoder similar to Socher et al. (2011), which encodes the syntactic information into a single discrete code. As illustrated in Fig. 1 (a), a TreeLSTM cell (Tai et al., 2015) computes a recurrent state based on a given input vector and the states of N_i child nodes:

$$h_i = f_{cell}(x_i, h_{i1}, h_{i2}, ..., h_{iN_i}; \theta). \quad (1)$$

The tree auto-encoder model is shown in Fig. 1 (c), where the encoder computes a latent tree representation. As the decoder has to unroll the vector representation following a reversed tree structure to predict the non-terminal labels, the standard TreeLSTM equation cannot be directly applied. To compute along the reversed tree, we modify Eq. 1 for computing the hidden state of the j-th child node given the parent-node state h_i:

$$h_{ij} = f_{dec}(h_i; \theta_j), \quad (2)$$

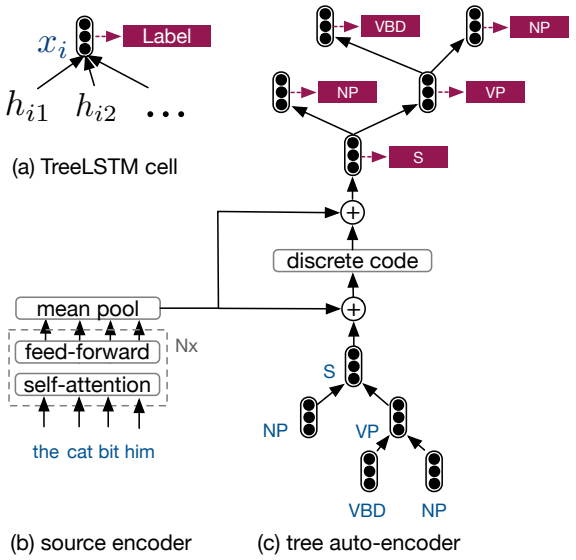

Figure 1: Architecture of the TreeLSTM-based auto-encoder with a discretization bottleneck for learning the sentence codes.

where the internal implementation of the recurrent function is same as Eq. 1, however, each node has a different parameterization depending on its position among siblings. Note that in the decoder side, no input vectors are fed to the recurrent computation. Finally, the decoder states are used to predict target labels, whereas the model is optimized with cross-entropy loss.

As the source sentence already provides hints on the target-side sentence structure, we feed the source information to the tree auto-encoder to encourage the latent representation to capture the syntax that cannot be inferred from the source sentence. To obtain the sentence codes from the latent tree representation, we apply *improved semantic hashing* (Kaiser and Bengio, 2018) to the hidden state of the root node, which discretizes the vector into a 8-bit code (binary vector). When performing improved semantic hashing, the forward pass computes two operations: binarization and saturated sigmoid, resulting in two vectors. One of these two vectors are randomly selected for the next computation. However, in the backward pass, the gradient always flows through the vector produced by saturated sigmoid. As the model is trained together with the bottleneck, the codes are optimized directly to minimize the loss function.

2.2 Diverse Generation with Code Assignment

Once we obtain the sentence codes, we prefix the target-side sentences in the training data with the corresponding codes. The resultant target sentence has a form of "⟨c12⟩ ⟨eoc⟩ Here is a translation.". The "⟨eoc⟩" token separates the code and words.

We train a regular NMT model with the modified training dataset. To generate diverse translations, we first obtain top-K codes from the probability distribution of code prediction. In detail, we select K sentence codes with the highest probabilities. Then, conditioning on each code, we let the beam search continue to generate the sentence, resulting in K translations conditioned on different codes.

3 Related Work

Existing works for diverse text generation can be categorized into two major categories. The approaches in the first categoriy sample diverse sequences by varying a hidden representation. Jain et al. (2017) generates diverse questions by injecting Gaussian noise to the latent in a VAE for encouraging the creativity of results. Xu et al. (2018) learns K shared decoders, conditioned on different pattern rewriting embeddings. The former method is evaluated by assessing the ability of generating unique and unseen results, whereas the latter is evaluated with the number of unique uni/bi-grams and the divergence of word distributions produced by different decoders. Independent to this work, Shen et al. (2019) also explores mixture-of-expert models with an ensemble of learners. The paper discusses multiple training strategies and found the multiple choice learning works best.

The second category of approaches attempts to improve the diversity by improving decoding algorithms. Li et al. (2016) modifies the scoring function in beam search to encourage the algorithm to promote hypotheses containing words from different ancestral hypotheses, which is also evaluated with the number of unique uni/bi-grams. Kulikov et al. (2018) uses an iterative beam search approach to generate diverse dialogs.

Comparing to these works, we focus on generating translations with different sentence structures. We still use beam search to search for best words in every decoding steps under the constraint of code assignment. Our approach also comes with the advantage that no modification to the NMT model architecture is required.

4 Experiments

4.1 Experimental Settings

We evaluate our models on two machine translation datasets: ASPEC Japanese-to-English dataset (Nakazawa et al., 2016) and WMT14 German-to-English dataset. The datasets contain 3M and 4.5M bilingual pairs respectively. For the ASPEC Ja-En dataset, we use the Moses toolkit (Koehn et al., 2007) to tokenize the English side and Kytea (Neubig et al., 2011) to tokenize the Japanese side. After tokenization, we apply byte-pair encoding (Sennrich et al., 2016) to segment the texts into subwords, forcing the vocabulary size of each language to be 40k. For WMT14 De-En dataset, we use sentencepiece (Kudo and Richardson, 2018) to segment the words to ensure a vocabulary size of 32k.

In evaluation, we report *tokenized BLEU* for ASPEC Ja-En dataset. For WMT14 De-En dataset, BLEU scores are generated using SacreBleu toolkit (Post, 2018). For models that produce sentence codes during decoding, the codes are removed from translation results before evaluating BLEU scores.

4.2 Obtaining Sentence Codes

For the semantic coding model based on BERT, we cluster the hidden state of "[CLS]" token into 256 clusters with k-means algorithm. The cluster ids are then used as sentence codes. For models using FastText Embeddings, pre-trained vectors (Common Crawl, 2M words) are used. Please note that the number of clusters is a hyperparameter, here we choose the number of clusters to match the number of unique codes in the syntax-based model.

To train the syntax coding model, we parse target-side sentences with Stanford CFG parser (Klein and Manning, 2003). The TreeLSTM-based auto-encoder is implemented with DGL,[1] which is trained using AdaGrad optimizer for faster convergence. We found it helpful to pre-train the model without the discretization bottleneck for achieving higher label accuracy.

[1] https://www.dgl.ai/

Model	BLEU	Oracle	DP
ASPEC Ja → En			
Transformer Baseline	27.1	-	22.4
+Diverse Dec (Li et al., 2016)	26.9	-	26.2
+ Random Codes	27.0	-	4.9
+ Semantic Coding (BERT)	26.8	28.8	30.6
+ Semantic Coding (FastText)	27.3	28.5	31.1
+ Syntactic Coding	27.4	29.5	**39.8**
WMT14 De → En			
Transformer Baseline	29.4	-	28.2
+Diverse Dec (Li et al., 2016)	29.1	-	31.0
+ Random Codes	29.5	-	3.8
+ Semantic Coding (BERT)	29.3	29.4	21.7
+ Semantic Coding (FastText)	28.5	29.2	28.8
+ Syntactic Coding	29.3	30.7	**33.0**

Table 1: Results for different approaches. The BLEU(%) are reported for the first sampled candidate. Oracle BLEU scores are produced with reference codes. Diversity scores (DP) are evaluated with Eq. 3.

	Tg 以上 の 温度 で I を 消去 できた。 (Japanese)
A	1. It is possible to eliminate I at temperatures above Tg . 2. It is possible to eliminate I at temperatures higher than Tg . 3. It is possible to eliminate I at the temperature above Tg .
B	1. above Tg , I was able to be eliminated . 2. It was found that the photoresists were eliminated at temperatures above Tg . 3. at the temperature above Tg , I was able to be eliminated .
C	1. I could be eliminated at temperatures above Tg . 2. I was removed at temperatures above Tg . 3. It was possible to eliminate I at temperatures above Tg .

Table 2: A comparison of candidates produced by beam search (A), semantic coding model based on BERT (B) and syntactic coding model (C) in Ja-En task

4.3 Quantitive Evaluation of Diversity

As we are interested in the diversity among sampled candidates, the diversity metric based on the divergence between word distributions (Xu et al., 2018) can not be applied in this case. In order to qualitatively evaluate the diversity of generated translations, we propose to use a BLEU-based discrepancy metric. Suppose Y is a list of candidate translations, we compute the diversity score with

$$\mathrm{DP}(Y) = \frac{1}{|Y|(|Y|-1)} \sum_{y \in Y} \sum_{y' \in Y, y' \neq y} 1 - \Delta(y, y'), \quad (3)$$

where $\Delta(y, y')$ returns the BLEU score of two candidates. The equation gives a higher diversity score when each candidate contains more unique n-grams.

4.4 Experiment Results

We use Base Transformer architecture (Vaswani et al., 2017) for all models. The results are summarized in Table 1. We sample three candidates with different models, and report the averaged diversity score. The BLEU(%) is reported for the candidate with highest confidence (log-probability). A detailed table with BLEU scores of all three candidates can be found in supplementary material, where the BLEU scores of the second and third candidates are on par with the baseline.

We compare the proposed approach to three baselines. The first baseline samples three candidates using standard beam search. We also tested the diverse decoding approach (Li et al., 2016). The coefficient γ is chosen to maximize the diversity with no more than 0.5 BLEU degradation. The third baseline uses random codes for conditioning.

As shown in the table, the model based on BERT sentence embeddings achieves higher diversity in ASPEC dataset, which contains only formal texts. However, it fails to deliver similar results in WMT14 dataset, which is more informal. This may be due to the difficulty in clustering BERT vectors which were never trained to work with clustering. The model using FastText embeddings is shown to be more robust across the datasets, although it also fails to outperform the diverse decoding baseline in WMT14 dataset.

In contrast, syntax-based models achieve much higher diversity in both datasets. We found the results generated by this model has more diverse structures rather than word choices. By comparing the BLEU scores, no significant degradation is observed in translation quality. As a control experiment, using random codes does not contributes to the diversity. As a confirmation that the sentence codes have strong impact on sentence generation, the models using codes derived from references (oracle codes) achieve much higher BLEU scores.

5 Analysis and Conclusion

Table 2 gives samples of the candidate translations produced by the models conditioning on different discrete codes, compared to the candidates produced by beam search. We can see that the candidate translations produced by beam search has only minor grammatical differences. In contrast, the translation results sampled with the syntactic coding model have drastically different grammars. By examining the results, we found the syntax-based model tends to produce one translation in active voice and another in passive voice.

To summarize, we show a diverse set of translations can be obtained with sentence codes when a reasonable external mechanism is used to produce the codes. When a good syntax parser exists, the syntax-based approach works better in terms of diversity. The source code for extracting discrete codes from parse trees will be publicly available.

Acknowledgement

The research results have been achieved by "Research and Development of Deep Learning Technology for Advanced Multilingual Speech Translation", the Commissioned Research of National Institute of Information and Communications Technology (NICT), JAPAN. This work was partially supported by JSPS KAKENHI Grant Number JP16H05872, Samsung Advanced Institute of Technology (Next Generation Deep Learning: from pattern recognition to AI) and Samsung Electronics (Improving Deep Learning using Latent Structure).

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

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

Unnat Jain, Ziyu Zhang, and Alexander G. Schwing. 2017. Creativity: Generating diverse questions using variational autoencoders. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5415–5424.

Lukasz Kaiser and Samy Bengio. 2018. Discrete autoencoders for sequence models. *CoRR*, abs/1801.09797.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *ACL*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *EMNLP*.

Ilya Kulikov, Alexander H. Miller, Kyunghyun Cho, and Jason Weston. 2018. Importance of a search strategy in neural dialogue modelling. *CoRR*, abs/1811.00907.

Jiwei Li, Will Monroe, and Daniel Jurafsky. 2016. A simple, fast diverse decoding algorithm for neural generation. *CoRR*, abs/1611.08562.

Toshiaki Nakazawa, Manabu Yaguchi, Kiyotaka Uchimoto, Masao Utiyama, Eiichiro Sumita, Sadao Kurohashi, and Hitoshi Isahara. 2016. Aspec: Asian scientific paper excerpt corpus. In *LREC*.

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011. Pointwise prediction for robust, adaptable japanese morphological analysis. In *ACL*, pages 529–533.

Matt Post. 2018. A call for clarity in reporting bleu scores. In *WMT*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909.

Tianxiao Shen, Myle Ott, Michael Auli, and Marc'Aurelio Ranzato. 2019. Mixture models for diverse machine translation: Tricks of the trade.

Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y Ng. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in neural information processing systems*, pages 801–809.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *ACL*.

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

Qiongkai Xu, Juyan Zhang, Lizhen Qu, Lexing Xie, and Richard Nock. 2018. D-page: Diverse paraphrase generation. *CoRR*, abs/1808.04364.