Generating Syntactically Diverse Translations with Syntactic Codes

朱 中元　中山 英樹
東京大学　大学院情報理工学系研究科
{shu, nakayama}@nlab.ci.i.u-tokyo.ac.jp

January 20, 2019

1 Introduction

When using machine translation products, users may desire to see candidate translations other than the best one. In this scenario, users usually expect the machine translation system to provide more candidates with drastically different syntactic structures. However, it is undesirable that the candidate translations contain less appropriate words which alter the meaning or nuance of the source sentence.

To obtain diverse sentences, conventional NMT models allow one to sample translations using the beam search algorithm, but the sampled translation results usually share similar syntactic structures. Recently, various methods [7, 12] are proposed to generate sentences with high diversity. These methods encourage the model to use creative vocabulary during the generation process. Most of them evaluate the generated results with corpus-level metrics such as the discrepancy of word distributions. Although producing creative words benefits tasks such as conversation generation, however in machine translation, it can damage the translation quality by changing the original meaning.

Here, we are interested in generating multiple translations with only syntactic diversity. Once the syntactic structure is determined, we want the model to still select the best words for each candidate translation. To minimize the cost of modification to the neural machine translation (NMT) models, we prefix the target sentences with syntactic tags that indicate their structures. As the tokens are generated in left-to-right order during decoding, the NMT model is forced to plan the syntactic structure before generating words. This approach can be easily integrated to existing NMT products as no modification to the NMT model architecture is required.

In this work, we choose the syntactic tags to represent the global syntactic structure of a sentence. Information about local structures such as the number of words in a noun phrase is not desired and has to be omitted in the tags. As consequence, we choose to simplify the part-of-speech (POS) tags and use it as a global structural representation.

To eliminate the negative impact on decoding speed, we use a sequence auto-encoder with a discretization bottleneck to compress the sequences into few codes. This approach is similar to the discrete auto-encoder approach [6], where they capture utterance-level information. We show that the syntactic tag sequences can be compressed into one or two codes but still maintain their effectiveness.

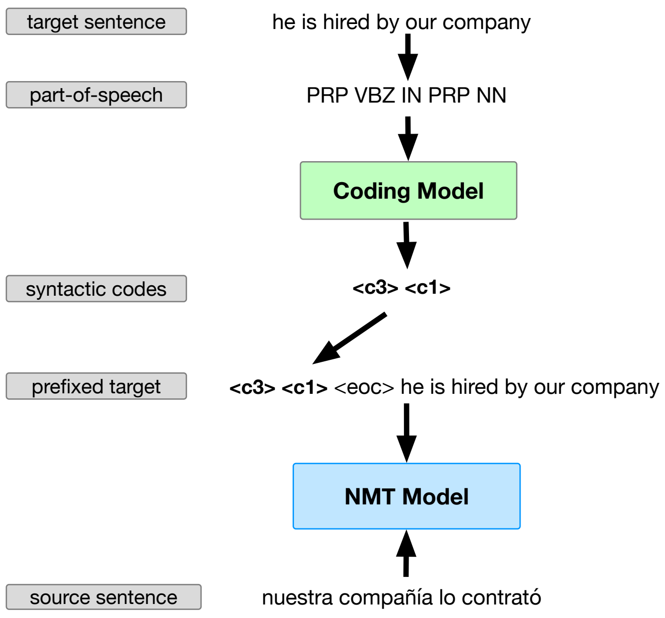

Figure 1: Proposed approach for conditioning the syntactic structure with discrete codes. A target sentence is parsed to obtain its syntactic tags. Then, based on the tags and source sentence, the coding model produces multiple discrete codes. Finally, an NMT model is trained based on the original source sentence and the target sentence prefixed by the codes.

Fig. 1 shows our proposed approach for training the NMT model. In the first phase, we parse target sentences to obtain their part-of-speech tags. Next, a coding model is trained to obtain the syntactic codes. We prefix each target sentence in the training data with the codes. The "⟨eoc⟩" token denotes the end of the codes. Finally, we train an NMT model on the modified dataset.

2 Learning Syntactic Codes

This section discusses the motivation and methodology for learning discrete syntactic codes. First, we describe an easy method to extract the global sentence structure from part-of-speech (POS) tags. Then, we describe a coding model that learns discrete syntactic codes to speed up the decoding.

2.1 Extracting Global Syntactic Structures

To construct the syntactic prior of translations, we need a representation of syntactic structures. However, it is extremely difficult for a neural network to automatically realize and extract the syntactic structure from the utterances. Therefore, we consider to derive the representation of global sentence structures from part-of-speech tags, which are available for most languages.

Normally, part-of-speech tags contain detailed information about local structures. For example, the words "destroy" and "break down" have different POS tags, but the difference between them are local. In this work, we want to use the syntactic tags only for planning the global sentence structure, other details such as words and local structures are left for the NMT model to fulfill.

Here, we reduce the POS tags to form a global structural representation with a simple heuristics:

1. Remove all tags other than "**N**", "**V**", "**PRP**", "**,**" and "**.**". Note that all tags begin with "N" (e.g. **NNS**) are mapped to "**N**", and tags begin with "V" (e.g. **VBD**) are mapped to "**V**".

2. Remove duplicated consecutive tags.

The following list gives an example of the process:

```
   Input: He found a fox behind the wall.
POS Tags: PRP VBD DT NN IN DT NN .
  Step 1: PRP V N N .
  Step 2: PRP V N .
```

It should be noted that other syntactic parse results can also be potentially useful to represent the global structure; this is left for future works to explore. In our experiments, we found simplified POS tags to be effective and convenient for implementation.

2.2 Code Learning

Prefixing the target sentences with the simplified tags will significantly increase the sequence length, thus worsen the latency of translating a sentence. To mitigate the impact, we propose to compress the syntactic tags into few discrete codes with a code learning model.

Code Encoder Given the sentence pair (X, Y), let the structural tags of the target sentence be

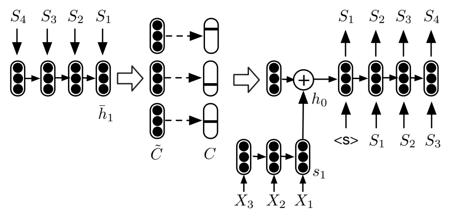

Figure 2: Architecture of code learning model. Discretization bottleneck is indicated by dashed lines.

$S_1, ..., S_T$, the encoder computes the logits of N softmax vectors:

$$\bar{h}_\mathrm{t} = \mathrm{LSTM}(\mathrm{E}(S_t), \bar{h}_{t+1}; \phi_e) , \quad (1)$$

$$[\tilde{C}_1, ..., \tilde{C}_N] = \mathrm{softmax}(f(\bar{h}_1; \phi_h)) , \quad (2)$$

where the tag sequence $S_1, ..., S_T$ is firstly encoded using a *backward* LSTM. $\mathrm{E}(\cdot)$ denotes an embedding function, and $f(\cdot)$ denotes a linear transformation. Then, we compute a set of continuous vectors $\tilde{C}_1, ..., \tilde{C}_N$, which are latterly discretized into approximated one-hot vectors $C_1, ..., C_N$ using Gumbel-Softmax repraparameterization trick [5, 8] as:

$$C_i = g(S, \epsilon; \phi)_i \quad (3)$$

$$= \mathrm{softmax}_\tau(\log \tilde{C}_i + \mathrm{gumbel}(\epsilon)) . \quad (4)$$

Here, τ is the temperature of the softmax. The computation of Gumbel-softmax relies on the Gumbel noise, which is defined as $\mathrm{gumbel}(\epsilon) = -\log(-\log(\epsilon))$. As both $\tilde{C}_i$ and ϵ are vectors in Eq. 4, the $\log(\cdot)$ here is an element-wise operation. The random variable ϵ is sampled from a uniform distribution $\mathrm{U}(0, 1)$. In Eq. 3, $g(S, \epsilon; \phi)$ represents the encoder function, which is used to produce the codes once its parameter ϕ is trained.

Decoder In the decoder, we combine the information from X and C to initialize a decoder LSTM that sequentially predicts the structural tags $S_1, ..., S_T$:

$$s_t = \mathrm{LSTM}(\mathrm{E}(X_t), s_{t+1}; \theta_x) , \quad (5)$$

$$h_0 = f([C_1, ..., C_N]; \theta_h) + s_1 , \quad (6)$$

$$h_t = \mathrm{LSTM}(\mathrm{E}(S_{t-1}), h_{t-1}; \theta_d) , \quad (7)$$

where $[C_1, ..., C_N]$ is a concatenation of N Gumbel-softmax code vectors. Note that only h_t is computed with a *forward* LSTM. Finally, we predict the probability of emitting each tag S_t with

$$P(S_t | S_{<t}, X, C) = \mathrm{softmax}(f_\mathrm{out}(h_t; \theta_\mathrm{out})) . \quad (8)$$

2.3 NMT with Structural Planning

Once the coding model is trained, the Gumbel-softmax code vectors $C_1, ..., C_N$ should be very close to one-hot vectors. We can extract N discrete codes by applying *argmax* on the softmax vectors, and prefix the target sentences in the training data with the codes, resulting in $(X, C; Y)$ training pairs. We connect the codes and target sentences with a "$\langle$eoc$\rangle$" token.

To evaluate the performance of structural planning with discrete codes, we train a regular NMT model with the modified dataset. The codes are removed from the translation results after decoding. Similarly, we can evaluate the tag planning model using $(X, S; Y)$ pairs.

3 Related Work

As generating diverse text has its application in tasks such as conversation generation, several existing works are presented to improve the diversity of language generation. Similar to [3], [2] generates diverse image captions with a conditional GAN, whereas [4] generates diverse questions with a VAE.

Some creative approaches are also proposed on this topic. [7] improves the diversity of generation by modifying the scoring function in beam search. A hyperparameter that controls the diversity is optimized by reinforcement learning. [12] learns K shared decoders, each is conditioned on a trainable pattern embedding. In each iteration, only the decoder with lowest cross-entropy receives training signal.

Existing works on language generation care about using creative vocabulary. Both [7] and [12] measure the diversity with corpus-level metrics. The former work evaluates the number of unique n-grams in the generated text, whereas the latter one measures the divergence of word distributions produced by different style-specific decoders. Our approach differs from these works by putting the focus on improving the syntactic diversity instead of choice of words.

4 Experiments

We evaluate our models on two machine translation datasets: IWSLT14 German-to-English dataset [1] and ASPEC Japanese-to-English dataset [9]. These datasets contain 178K and 3M bilingual pairs, respectively. For the IWSLT14 De-En dataset, we apply the Moses toolkit to tokenize both sides of the corpus. For the ASPEC Ja-En dataset, we use the Moses toolkit to tokenize the English side and Kytea to tokenize the Japanese side.

After tokenization, we apply byte-pair encoding [10] to segment the texts into subwords, forcing the vocabulary size of each language to be 20K and 40K for the IWSLT dataset and ASPEC dataset.

4.1 Evaluation of NMT Models

Model Architecture: To create a strong NMT baselines, we test two types of NMT models in our experiments: deep LSTM model and transformer model [11]. Both models are broadly used in industry recently.

	Model	BLEU
De-En	Deep LSTM baseline	29.5
	+ tag planning	20.5
	+ discrete plan (N=1, K=4)	29.5
	+ discrete plan (N=2, K=4)	**29.7**
Ja-En	Deep LSTM baseline	25.2
	+tag planning	18.7
	+discrete plan (N=1, K=4)	25.3
	+discrete plan (N=2, K=4)	**25.6**
	Transformer baseline	26.0
	+tag planning	26.8
	+discrete plan (N=1, K=4)	27.3
	+discrete plan (N=2, K=4)	**27.4**

Table 1: Performance evaluation for NMT models trained with different approaches. The BLEU(%) scores are reported using beam size of 5.

For the deep LSTM model, we use two layers of bidirectional LSTM encoders with two layers of LSTM decoders in the NMT model. The hidden layers have 256 units for the IWSLT De-En task and 1000 units for the ASPEC Ja-En task. For the transformer model, we follow the settings of the base transformer. The hidden size is 512, both the encoder and the decoder have 6 layers.

Evaluation Results: Table 1 shows the resultant BLEU scores of different models, which indicates that our proposed planning approach does not degrade the translation quality in both translation tasks. When we use the part-of-speech tags as a prior to directly control the sentence structure, a drop in BLEU scores is observed for LSTM-based models.

We can observe an improvement on translation performance especially for the transformer models. The improvement may be the consequence of being able to simultaneously explore drastically different candidates. A recent study [7] also showed that beam search performance can be improved by increasing the diversity of the candidate space.

4.2 Evaluation of Syntactic Diversity

In our approach, a single code is not guaranteed to be aligned to a specific structure. Therefore, we can not evaluate the diversity with the divergence between word distributions as [12] does. In order to quantitatively evaluate the diversity of generated translations, we propose to use a simple BLEU discrepancy metric. Suppose Y is a list of candidate translations,

Dataset	**Model**	**DP**	**POS-DP**
De-En	baseline NMT	35.99	27.72
	discrete plan	**39.79**	**36.51**
Ja-En	baseline NMT	25.89	20.28
	discrete plan	**52.10**	**45.42**

Table 2: Evaluation of discrepancy scores (DP) for the baseline model and the discrete planning approach. We also report the discrepancy scores of the part-of-speech tags of candidates (POS-DP).

	AP の過程について述べた。
BS	`process of the AP is described .`
	`a process of AP is described .`
	`reaction of AP is described .`
PL	`the process of AP is described .`
	`this paper describes the process of AP.`
	`here was described on process of AP .`

Table 3: A comparison of translation candidates produced by beam search (BS) and the proposed planning approach (PL) for the given Japanese sentence.

we compute the discrepancy with

$$\mathrm{DP}(Y) = \frac{1}{|Y|(|Y|-1)} \sum_{y \in Y} \sum_{y' \in Y, y' \neq y} 1 - \Delta(y, y'), \quad (9)$$

where $\Delta(y, y')$ computes the BLEU score of two candidates. This equation computes the mean value of 1 - BLEU between candidate pairs. The score is higher when the candidates have less similarity.

In order to evaluate the syntacitc diversity, we further propose a POS-DP metric, in which $\Delta(y, y')$ computes the BLEU scores using the POS tags of two candidates. Therefore, lexical diversity is completely omitted in the scores.

We report the averaged discrepancy scores in Table 2. We first apply beam search on the baseline model with a beam size of 4 to obtain four candidates for measuring the baseline syntactic diversity. Then we sample translations from a NMT with discrete code prior ($N = 1, K = 4$). As there are only four possible codes, we sample four translations and measure the discrepancy scores of them.

We can see that the proposed method achieves higher discrepancy scores, which means that the candidate translations are less similar to each other. A larger gap in diversity is observed when evaluating with the POS-DP metric. It indicates that the proposed method creates more diversity in terms of syntactic structure. In Table 3, we give a comparison of the candidate translations sampled by the beam search and the proposed approach.

5 Acknowledgement

The research results have been achieved by the Commissioned Research of National Institute of Information and Communications Technology (NICT).

6 Conclusion

In this paper, we learn fixed-number discrete codes to capture the syntactic structure of translations. During translation, the syntactic codes are generated first to constrain the word prediction. Experiments show that the quality of the best translation is still guaranteed even under constraint. Performance gain is observed when using the transformer model. By switching with different syntactic codes, we can sample translations with drastically different structures.

References

[1] Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*, 2014.

[2] Bo Dai, Dahua Lin, Raquel Urtasun, and Sanja Fidler. Towards diverse and natural image descriptions via a conditional gan. *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2989–2998, 2017.

[3] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. Toward controlled generation of text. In *ICML*, 2017.

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

[5] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *CoRR*, Vol. abs/1611.01144, , 2016.

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

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

[8] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *CoRR*, Vol. abs/1611.00712, , 2016.

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

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

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

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