Categorizing Semantic Representations for Neural Machine Translation

Yongjing Yin1,2; Yafu Li1,2, Fandong Meng3, Jie Zhou4, Yue Zhang2,4
1 Zhejiang University
2 School of Engineering, Westlake University
3 Pattern Recognition Center, WeChat AI, Tencent Inc
4 Institute of Advanced Technology, Westlake Institute for Advanced Study
{yinyongjing.liyafu}@westlake.edu.cn {fandongmeng.withtomzhou}@tencent.com yue.zhang@wias.org.cn

Abstract

Modern neural machine translation (NMT) models have achieved competitive performance in standard benchmarks. However, they have recently been shown to suffer limitation in compositional generalization, failing to effectively learn the translation of atoms (e.g., words) and their semantic composition (e.g., modification) from seen compounds (e.g., phrases), and thus suffering from significantly weakened translation performance on unseen compounds during inference. We address this issue by introducing categorization to the source contextualized representations. The main idea is to enhance generalization by reducing sparsity and overfitting, which is achieved by finding prototypes of token representations over the training set and integrating their embeddings into the source encoding. Experiments on a dedicated MT dataset (i.e., CoGnition) show that our method reduces compositional generalization error rates by 24% error reduction. In addition, our conceptually simple method gives consistently better results than the Transformer baseline on a range of general MT datasets.

1 Introduction

Neural machine translation (NMT) has achieved competitive performance on benchmark datasets such as WMT (Vaswani et al., 2017; Edunov et al., 2018; So et al., 2019). However, the generalization to low-resource domains (Bapna and Firat, 2019b; Zeng et al., 2019; Bapna and Firat, 2019a; Khandelwal et al., 2021) and robustness to slight input perturbations (Belinkov and Bisk, 2018; Xu et al., 2021b) are relatively low for NMT models. In addition, recent studies show that NMT systems are vulnerable to compositional generalization (Lake and Baroni, 2018; Raunak et al., 2019; Guo et al., 2020; Li et al., 2021; Dankers et al., 2021; Chaabouni et al., 2021), namely the ability to understand and produce a potentially infinite (formally exponential to the input size) number of novel combinations of known atoms (Chomsky, 2009; Montague and Thomason, 1975; Janssen and Partee, 1997; Lake and Baroni, 2018; Keysers et al., 2020a).

Take CoGnition (Li et al., 2021), a dedicated MT dataset, for example (Figure 1). Despite that certain instances of translation atoms (e.g., small, large, car, and chair) and their semantic compositions (e.g., small chair and large car) are frequent in training data, unseen compositions of the same atoms (e.g., large chair) during testing can suffer from large translation error rates. Compositional-ality is also a fundamental issue in language understanding and motivated for translation (Janssen and Partee, 1997; Janssen, 1998), which has been suggested as being essential for robust translation (Raunak et al., 2019; Li et al., 2021) and efficient low-resource learning (Chaabouni et al., 2021).

The current dominant method to NMT employs a sequence-to-sequence architecture (Sutskever et al., 2014; Vaswani et al., 2017), where an encoder is used to find representations of each input token that thoroughly integrates its sequence-level context in-
formation, and a decoder refers to such contextualized representations for generating a translation sequence. A key reason of failure on compositional generalization is that the correspondence between pairs of token sequences is modeled as a whole. Specifically, NMT models are trained end-to-end over large parallel data without disentangling the representation of individual words or phrases from that of whole token sequences. At the sequence level, the source input sample space is highly sparse mainly due to semantic composition, and small changes to a sentence can lead to out-of-distribution issues (Sagawa et al., 2020; Conklin et al., 2021; Liu et al., 2021).

Intuitively, one way to solve this problem is to decouple token-level information from the source sequence by injecting token-level translation distribution (e.g., $P(\text{petit}|\text{small})$) into the source representation. Given the fact that the source-side contextualized representations encode rich token-level translation information (Kasai et al., 2021; Xu et al., 2021a), we categorize sparse token-level contextualized source representations into a few representative prototypes over training instances, and make use of them to enrich source encoding. In this way, when encoding a sequence, the model observes less sparse prototypes of each token, thus alleviating excessively memorizing the sequence-level information.

We propose a two-stage framework to train prototype-based Transformer models (Proto-Transformer). In the first stage, we warm up an initial Transformer model which can generate reasonable representations. In the second stage, for each token, we run the trained model to extract all contextualized representations over the training corpus. Then, we perform clustering (e.g., K-Means) to obtain the prototype representations for each token. Take Figure 2 as an example, for the token “Toy”, we collect all the contextualized representations and cluster them into 3 prototypes. Finally, we extend the base model by fusing the prototype information back into the encoding process through a prototype-attention module, and continue to train the whole model until convergence.

Experimental results on CoGnition show that our method significantly improves novel composition translation by over 24% error reduction, demonstrating the effectiveness for tackling the compositional generalization problem. To further verify the effectiveness on more datasets, we conduct experiments on 10 commonly used MT benchmarks and our method gives consistent BLEU improvement. We also present empirical analysis for prototypes and quantitative analysis on compositional generalization. The comparison between the one-pass and the two-pass training procedure shows that the one-pass method is both faster and more accurate than the two-pass one, demonstrating that more generalizable prototypes extracted from early training phrase are more beneficial to compositional generalization. Additionally, quantitative analysis demonstrates that our proposed model is better at handling longer compounds and more difficult composition patterns. The code is publicly available at https://github.com/ARIES-LM/CatMT4CG.git.

## 2 Related Work

### Compositional Generalization

Recent work (Lake and Baroni, 2018; Keysers et al., 2020b) has demonstrated weak compositionality of neural models using dedicated datasets. Various methods haven been proposed to solve the issue of compositional generalization such as encoding more inductive bias (Li et al., 2019; Korrel et al., 2019; Baan et al., 2019; Chen et al., 2020a; Gordon et al., 2020; Herzig and Berant, 2021), meta-learning (Lake, 2019; Conklin et al., 2021), and data augmentation (Andreas, 2020; Akyürek et al., 2021). Recently, Ontañón et al. (2021) and Csordás et al. (2021) show that the Transformer architecture can perform better on compositional generalization with some modifications. Although these methods have demonstrated better generalization or interpretability, most of them are limited small vocabulary and limited samples semantic parsing datasets. In the context of machine translation, Lake and Baroni (2018) construct a small dataset where the training data contains a word $\text{daxy}$ along with its parallel sentences of a single pattern (e.g., $\text{I am daxy, je suis daxist}$) while the test set contains novel patterns (e.g., $\text{He is daxy}$). However, the experiment is limited in that the test set only consists of 8 samples. Different from existing work, Li et al. (2021) propose a large dataset (CoGnition) and construct a large-scale test set that contains newly constructed constituents as novel compounds, so that generalization ability can be evaluated directly based on compound translation error rate. We proposed a method enhancing compositional generalization on the dedicated dataset of Li et al. (2021), while at the same time gives improvements to the machine
She chose a cute style.
Taylor chose to get flowers.
Taylor chose the color she wanted.

He chose a toy car.
One day, he walked to his car.
It was his favorite toy.

The boy played with it all day.

The Transformer encoder (Vaswani et al., 2017) first maps \( X \) to embeddings, packing them as a matrix \( H^0 \), and then takes \( H^0 \) as input and outputs a contextualized sequence representation \( H^L \in \mathbb{R}^{d \times T} \), where \( d \) and \( L \) denote dimension size and the number of layers respectively.

**Attention.** Formally, given a set of packed query, key, and value matrices \( Q, K, \) and \( V \), the dot product attention mechanism are defined as

\[
\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{Q^T K}{\sqrt{d}}\right)V,
\]

where \( d \) is the dimension of the key vector.

A typical extension of the above is multi-head attention (MHA), where multiple linear projections are executed in parallel, and the outputs of all heads are concatenated:

\[
\text{MHA}(Q, K, V) = W^O[\text{head}_1; \ldots; \text{head}_h],
\]

where \( W^O, W^K, W^V \) are model parameters.

**Layer Structure.** The Transformer encoder has \( L \) identical layers, each of which is composed of two sublayers (i.e., self-attention and feed-forward networks). In the \( l \)-th self-attention layer, the query, key, and value matrices are all the hidden states from the previous layer \( H^{l-1} \):

\[
H^l_a = \text{MHA}(H^{l-1}, H^{l-1}, H^{l-1}).
\]

The feed-forward sublayer is a two-layer transformation with ReLU activation:

\[
H^l = W^2\text{ReLU}(W^1H^l_a + b^l_1) + b^l_2,
\]
where $W_1^l, b_1^l, W_2^l,$ and $b_2^l$ are trainable parameters. The layer normalization and residual connection are omitted for brevity.

### 3.2 Proto-Transformer

Our proposed Proto-Transformer extends the base Transformer by introducing a prototype-attention module on top of the self-attention module in each encoder layer, which aggregates token-level prototype representations for each token.

#### 3.2.1 Prototype-attention

Assuming that each token $x_t$ in the input sequence is assigned a set of prototype vectors packed as a matrix $C^x_t \in \mathbb{R}^{d \times k}$ where $k$ is the number of prototypes. The prototype attention aggregates the global prototype information and refines the contextualized representations for each token by the multi-head attention mechanism:

$$H^l_p = \text{MHA}(H^l_x, C, C),$$

where $C$ is all the prototype representations of the sequence (i.e., $\{C^{x_1}, C^{x_2}, ..., C^{x_T}\}$), and the locality of prototype attention is implemented by mask mechanisms. The output $H^l_p$ is fed into the feed-forward network (Eq. 5).

The difference between the Proto-Transformer encoder and the Transformer encoder is illustrated by the dotted box in Figure 2. By stacking multiple self-attention layers with prototype-attention, the encoder is able to exploit less-sparse prototype representations, preventing the model from over-memorizing the local context.

#### 3.2.2 Training

Proto-Transformer takes a two-stage training process, which is summarized in Algorithm 1. Here, $\mathcal{D}$ denotes a training corpus, $\mathcal{V}$ denotes the vocabulary of $\mathcal{D}$, $\theta^{(0)}$ denotes initial parameters of a Transformer model, $\theta_+$ denotes parameters of the prototype-attention module, and $k$ and $N$ denote the number of prototypes and training epochs of stage 1 respectively.

**Stage 1.** We first train a base Transformer model for $N$ epochs until it is able to generate reasonable translations. Given the training corpus $\mathcal{D} = \{(X, Y)\}$, where $X$ and $Y$ denote a source sentence and target translation, respectively, the model is optimized by minimizing cross-entropy loss. After training for $N$ epochs we obtain a model $\theta^{(N)}$, which has acquired some translation knowledge.

**Stage 2.** We cluster the contextualized representations of each token to obtain its prototype representations. In particular, using the model $\theta^{(N)}$, we build a token-representation lookup table $\mathcal{P}$. We iterate through the whole training corpus and calculate contextualized representations $\{H^{v_1}_x, ..., H^{v_i}_x\}$ of each source token $v_i$ in the vocabulary except punctuation, where $R(v_i)$ is the number of contextualized representations of token $v_i$. Here we omit the superscript $L$ for simplicity. Next, for each token $v_i$, we use K-Means (Lloyd, 1982) \(^1\) to cluster the contextualized representations due to its efficiency for large number of samples in high dimensions:

$$C^{v_i} = \text{K-Means}(H^{v_i}_1, ..., H^{v_i}_{R(v_i)}).$$

All token-prototype pairs are saved in a lookup table $\mathcal{P}$. Finally, we introduce the prototype-attention modules with parameters $\theta_+$ to the Transformer encoder, and for each source sentence we retrieve the prototype representations from table $\mathcal{P}$. We continue to optimize the whole parameter set $\{\theta^{(N)} \cup \theta_+\}$ to obtain a final model $\theta^*$.

As mentioned earlier in the introduction, we compare the above one-pass method to a two-pass method, where we train the base model to convergence instead of for $N$ iterations, before training

---

\(^1\)https://scikit-learn.org/stable/modules/classes.html
a Proto-Transformer from scratch which uses clustered prototypes from the base model. The relative advantage is that the prototypes are taken from a fully trained NMT model, but the disadvantage is that the training time doubles. In addition, it remains an empirical question whether fully-trained NMT models give prototypes that are more suitable for guiding a final model. We discuss this in the Section 5.1.

It is worth noting that a limitation of K-Means is the pre-specification of cluster numbers, which can be different for different tokens in practice. Non-parametric clustering algorithms such as DBSCAN (Ester et al., 1996) can potentially solve the problem. However, their runtime complexity is typically \(O(n^2)\) or worse, which prevents us from choosing them for large-scale NMT data.

4 Experiments

4.1 Experimental Settings

Datasets. We use CoGnition (Li et al., 2021) to systematically evaluate compositional generalization in MT scenarios, an English→Chinese (En→Zh) translation dataset. It consists of a training set of 196,246 sentence pairs, a valid set and a test set of 10,000 samples. In particular, it has a dedicated test set (i.e., CG-test set) consisting of 10,800 sentences containing novel compounds, so that the model’s ability of compositional generalization can be measured by the ratio of compounds that are correctly translated. In addition, we choose 9 machine translation tasks from IWSLT, WMT and JRC-Acquis to verify the general effectiveness of our methods. The dataset statistics are shown in Appendix B.

Setup. We use Transformer (Vaswani et al., 2017) as our baseline models implemented using the Fairseq toolkit (Ott et al., 2019). For CoGnition and IWSLT, we use the Transformer iwslt_de_en setting while for the others we use the transformer_base setting. Following previous work, for IWSLT and JRC-Acquis En↔Es, we use beam search with width 5 and length penalty 0.6 for inference, whereas for the other datasets we set the beam width as 4. For CoGnition, \(N\) and \(k\) are set as 8 and 3 based on the validation set, and \(N\) and \(k\) for the other datasets are shown in Appendix B. Empirically, \(N\) is recommended to chosen from 10% to 25% of the total training epochs. We conduct categorization for all tokens except punctuation and low-frequent words, and the token on CoGnition is word and is subword on other datasets. For reproducibility and stability, we train 6 models with seeds provided in (Li et al., 2021) for each method and report the average performance.

Evaluation Metrics. We use compound translation error rate (CTER; (Li et al., 2021)) to measure model performance on CoGnition. Specifically, instance-level CTER denotes the ratio of samples where the novel compounds are translated incorrectly, while aggregate-level CTER denotes the ratio of compounds that suffer at least one incorrect translation in corresponding contexts. To calculate CTER, Li et al. (2021) provide a manually collected dictionary for all the atoms based on the training set, since each word may have different translations. We also conduct human evaluation (Appendix A) as a supplement. We conduct evaluation using BLEU (Papineni et al., 2002) for the other datasets.

4.2 Baseline Methods

We compare our method with following baselines: (1) Transformer (Vaswani et al., 2017), which uses the same settings as Li et al. (2021); (2) Transformer-Small, a more compact Transformer model with 4 layers and 256 hidden size; (3) Transformer-Deep, which increases the number of encoder layers to 8 to take the same parameters as Proto-Transformer; (4) Transformer-Rela, which replaces absolute positional encoding with a relative one, an important component for compositional generalization demonstrated in (Csordás et al., 2021; Ontañón et al., 2021); (5) Bow (Raunak et al., 2019): which uses bag-of-words loss to regularize the encoder based on the observation that the encoder representations are much weaker on unseen composition; and (6) SeqMix (Guo et al., 2020): which synthesizes examples by interpolating embeddings\(^2\).

4.3 Results on CoGnition

The main results on CoGnition are shown in Table 1. Transformer gives instance-level and aggregate-level CTERs of 28.42% and 62.88%, respectively. In comparison, Proto-Transformer gives a score of 21.69% and 51.84%, respectively, with a significant improvement of 6.73% and 11.04% accordingly. The performance is the best when the hyper-parameters of Beta distribution are set to 1.0, which is consistent with (Guo et al., 2020).

\(^2\)The performance is the best when the hyper-parameters of Beta distribution are set to 1.0, which is consistent with (Guo et al., 2020).
Table 1: Compound translation error rate (CTER) on CoGnition. We report instance-level and aggregate-level CTERs, separated by “/”. NP, VP, and PP denote noun phrases, verb phrases and positional phrases, respectively, three compound types in the compositional generalization test set (CG-test set).

Moreover, Proto-Transformer outperforms all baseline systems significantly, indicating that categorization on the contextualized representations is more beneficial to compositional generalization. We also calculate the BLEU scores, and Proto-Transformer and Transformer obtain 60.1 and 59.5, respectively. The BLEU scores are all relatively high since the sentences on the CG test set are similar to the sentences in training data except the novel compounds. For CoGnition, CTER is more accurate and suitable than BLEU because the translation dictionary processes multiple accurate translations of each compound better.

Since Proto-Transformer brings some extra parameters (approximately 6M), we investigate whether the performance improvement is derived from the increase of model parameters. As can be seen, Transformer-Deep performs poorly on the CG-test set, indicating that only increasing model capacity may be harmful since it leads to worse over-fitting to sequence-level distributions. Besides, the more compact model Transformer-Small yields better results than Transformer and Transformer-Deep but lags far behind Proto-Transformer. This shows that model size is useful but not sufficient for solving compositional generalization.

Proto-Transformer performs better than Bow, indicating that the encoder representations refined by categorization are more adequate than the regularization technique. Compared to SeqMix, the improvement of Proto-Transformer is more significant (2.31% vs 11.04% aggregate-level CTER). SeqMix reduces representation sparsity via linear interpolation in the input embedding space, and we conjecture that the stochastically synthetic samples may be unreasonable and harmful to model training. Relative positional embedding (Shaw et al., 2018) is demonstrated to be important for compositional generalization. Specifically, Transformer-Rela reduces CTERs by 0.92% and 2.85% but is inferior to ours, indicating that the prototypes bring more than positional information.

4.4 Results on General MT Datasets

We conduct experiments on several general MT benchmarks, where the translation compound error rate (CTER) cannot be calculated directly. The performances on IWSLT, WMT and JRC-Acquis are presented in Table 2. Compared with Transformer, our model achieves consistent improvement (0.67 BLEU), demonstrating its effectiveness under general evaluation settings. Proto-Transformer outperforms Transformer by 0.91 and 0.78 BLEU scores on WMT’16 En→Ro and Ro→En respectively. For JRC-Acquis, our model achieves an average improvement of 0.62 BLEU score on En→Es and 0.56 BLEU score on Es→En. On the largest dataset WMT’16 En→De, our model also performs better than Transformer by 0.54 BLEU score. In addition, the large datasets possibly benefit from the additional parameters (about 6M). We run an experiment of a model with 8 encoder layers on EN-DE,
Table 3: CTERs against different number of prototypes on the CG-test set. “0” denotes to the Transformer baseline.

<table>
<thead>
<tr>
<th>$k$</th>
<th>CTER</th>
<th>$k$</th>
<th>CTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>28.32%/62.88%</td>
<td>3</td>
<td>21.09%/51.84%</td>
</tr>
<tr>
<td>1</td>
<td>23.60%/54.33%</td>
<td>4</td>
<td>23.04%/52.70%</td>
</tr>
<tr>
<td>2</td>
<td>22.58%/53.19%</td>
<td>5</td>
<td>24.98%/54.71%</td>
</tr>
</tbody>
</table>

Table 4: Word translation variations induced from Transformer (TF) and Proto-Transformer (PT). The columns PT and TF denote the number of aligned Romanian tokens, and the bold characters display the tokens induced from Proto-Transformer.

<table>
<thead>
<tr>
<th>Tgt tokens</th>
<th>PT</th>
<th>TF</th>
<th>Src tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>yesterday</td>
<td>1</td>
<td>3</td>
<td>ieri, psd, victor</td>
</tr>
<tr>
<td>limited</td>
<td>2</td>
<td>5</td>
<td>limitat, limitat, comert, apel, ...</td>
</tr>
<tr>
<td>six</td>
<td>1</td>
<td>6</td>
<td>sase, procente, saptamanu, din, ...</td>
</tr>
<tr>
<td>agriculture</td>
<td>1</td>
<td>4</td>
<td>agricultura, plasa, ...</td>
</tr>
<tr>
<td>culture</td>
<td>2</td>
<td>5</td>
<td>cultura, culturii, profesion...</td>
</tr>
<tr>
<td>republic</td>
<td>1</td>
<td>3</td>
<td>república, liderului, ...</td>
</tr>
<tr>
<td>simply</td>
<td>1</td>
<td>3</td>
<td>simplu, nu, individuala, ...</td>
</tr>
<tr>
<td>november</td>
<td>1</td>
<td>6</td>
<td>noiembrie, propaganda, 2008, ...</td>
</tr>
<tr>
<td>tomorrow</td>
<td>1</td>
<td>3</td>
<td>maine, dimineata, amiaza, ...</td>
</tr>
<tr>
<td>saturday</td>
<td>1</td>
<td>3</td>
<td>sambata, seara, dimineata, ...</td>
</tr>
</tbody>
</table>

Figure 3: CTERs of Proto-Transformer and Transformer over different compound and context lengths.

5 Analysis

5.1 Prototypes

Effect on the Number of Prototypes. The number of prototypes $k$ controls the granularity of context used to refine the representation learning, and is determined based on the model loss on the development set. In this experiment, we investigate its influence on generalization performance. As shown in (Li et al., 2021), the model mis-translates “He became sick from eating all of the peanut butter on the ball” into “He became sick from eating all of the peanut butter on the field”. With a minor mistake on the compound “on the ball”, the model achieves a sentence-level BLEU of 61.4, but the full sentence meaning is largely affected.

As can be seen in Table 4, Proto-Transformer attends to much fewer token types than Transformer. This shows that Proto-Transformer’s translations are less prone to context changes, which indicates the sparsity reduction of input sample space. The finding also partly explains the improvement on the Ro→En dataset. The above observations verify our intuition in the introduction that adding prototypes to the source representation enhances the model’s knowledge on token-level translation consistency, which leads to better compositional generalization.

It is noteworthy that increased translation consistency may increase translationese effects (Vanmassenhove et al., 2021). We argue that there can be a trade-off between the consistency modeling and translationese. Our approach does not force the model to generate highly consistent translation, but aims to alleviate the vulnerability to context changes by introducing several prototypes.

One-pass vs Two-pass. The possible advantage of the two-pass training procedure (Two-Pass) mentioned in Introduction is that the quality of the prototype representations generated by a converged model may be better. Specifically, Two-Pass achieves 24.36% instance-level and 55.69% aggregate-level CTER, outperforming Transformer...
by 4.06% and 7.19%, respectively, but is noticeably inferior to Proto-Transformer (21.69% and 51.84%). This observation may relate to the observation that neural models learn more generalizable features in the early phase before memorization (Arpit et al., 2017; Stephenson et al., 2021). The one-pass approach can not only reduce the training cost (11,103 seconds vs 6,286 seconds) but also leverage more generalizable features to better improve models’ compositional generalization. In addition, the prototypes can be initialized randomly and trained along with the model, and this variant obtains 26.24% and 57.94% CTERs. It performs much worse than Proto-Transformer (21.69% and 51.84%) though they have identical number of parameters and architecture, indicating the effectiveness of explicit categorization during training.

5.2 Effects on Compositional Generalization

**Composition Length.** Longer compounds are harder to generalize as they contain richer semantic information (Li et al., 2021). We classify the test samples by compound length and context length, and calculate the instance-level CTER. In Figure 3, we can observe that the advantage of Proto-Transformer grows larger in generalizing longer compounds and context. In particular, Proto-Transformer gives a lower CTER by 12.80% over samples with context longer than 13 tokens. The underlying reason can be that longer compounds or contexts are more sparse in input space, and Proto-Transformer alleviates sparseness by putting token distributions into representations via prototypes.

**Modifier.** One challenging type of novel compounds in CoGnition is the postpositive modifier atom (MOD), which is constructed to enrich the information of its preceding word (e.g., *he liked* in the sentence *he bought the car he liked*). The difficulty of translating compounds with MOD lies in word reordering from English to Chinese. We divide the test samples into two groups according to compounds with or without MOD (Figure 4). Proto-Transformer demonstrates larger advantage in translating the compounds with MOD, showing its superiority in processing complex semantic composition. To further understand the effectiveness, we choose the token *liked*, a core part of the MOD atom *he liked*, and visualize its representations with t-SNE (van der Maaten and Hinton, 2008). In Figure 5, we can see that the representations of *he liked* serving as MOD are concentrated at the leftmost prototype, while the representations of *he liked* not serving as MOD scatter on the other prototypes. Through representation categorization, Proto-Transformer finds a specialized prototype which abstracts the knowledge of *liked* serving as a part of MOD, and refers to the MOD prototype when processing compounds with MOD.

**Case Study** We present 3 source samples containing a novel compound *woke the silly boyfriend up* and 5 atoms, i.e., *woke*, *the*, *silly*, *boyfriend* and *up*, and their translations in Table 5. For all samples, correct translations should contain complete semantic meaning of the compound. Proto-Transformer correctly translates the compound along with different contexts across all samples, while Transformer suffers various mistakes. In the first sample, the translation of *something terrible* is omitted by Transformer. In the second sample, Transformer omits *silly boyfriend* and mistranslates *woke the silly boyfriend up* into *a silly bird woke up*. Similar mistake can be observed in the third example, where Transformer mistranslates *his dog woke the silly boyfriend up* into *his dog woke up*. Transformer overfits contexts and makes errors on unseen compositions, while our model is more stable.
6 Conclusion

We investigated a conceptually simple method for enhancing compositional generalization of NMT models, proposing a two-stage training framework to fuse prototype representations into the encoding process of Transformer. Experiments on CoGnition show the effectiveness of our method on compositional generalization, and extensive results over 9 translation tasks verify the generality of our method. To our knowledge, we are the first to propose token-level categorization for NMT, achieving promising performance on both a large-scale compositional generalization dataset and general datasets.

Acknowledgements

The work is funded by the Zhejiang Province Key Project 2022SDXHDX0003. We would like to thank all of the anonymous reviewers and Jiali Zeng for the helpful comments. Yue Zhang is the corresponding author.

References


A Human evaluation

<table>
<thead>
<tr>
<th>Model</th>
<th>Adequacy</th>
<th>Fluency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformer</td>
<td>4.32</td>
<td>4.27</td>
</tr>
<tr>
<td>Proto-Transformer</td>
<td>4.51</td>
<td>4.54</td>
</tr>
</tbody>
</table>

Table 6: Human evaluation on adequacy and fluency.

We conduct the human evaluation for translations in terms of adequacy and fluency. We randomly sample 100 sentences from the CG-test set of CoGnition, and invite three annotators to evaluate the translation adequacy and fluency ranging from one to five. The five point scale for adequacy indicates how much of the meaning expressed in the reference translation is also expressed in a hypothesis translation: 5 = All, 4 = Most, 3 = Much, 2 = Little, and 1 = None. The five point scale for fluency indicates how fluent the translation is: 5 = Flawless, 4 = Good, 3 = Non-native, 2 = Disfluent, and 1 = Incomprehensible.

The average of the scores from the three annotators is taken as the final score, and the results of the baseline and our model are shown in Table 6. Compared with Transformer, Proto-Transformer improves adequacy and fluency by 0.19 and 0.27, respectively. Proto-Transformer achieves more accurate translation of the novel compounds by alleviating the problem of compositional generalization, and possibly make the other part of the sentence to be translated better.

B Dataset Statistics and Hyper-parameters

We list the statistics and the introduced hyper-parameters of all the datasets in Table 7.

For IWSLT’14 English↔German (En↔De), IWSLT’14 English→Italian (En→It), and IWSLT’15 English→Vietnamese (En→Vi), we use Moses tokenizer\(^3\) and apply joint BPE (Sennrich et al., 2016) with 10,000 merge operations. For WMT’16 English↔Romanian (En↔Ro), we use the processed data from Lee et al. (2018). For WMT’16 English→German (En→De) and WMT’17 English→Finnish (En→Fi), we apply joint BPE with 37,000 and 32,000 merge operations, respectively. For JRC-Acquis English↔Spanish (En↔Es) we use the datasets processed by Gu et al. (2018). For WMT’17 En-Fi, we use the concatenation of newstest2015, newstest2016 and newstestB2016 as the development set, and the newstest2017 as the test set.

C Target-side Prototypes.

Target-side prototypes possibly contain more bilingual translation knowledge since the decoder processes target sentences based on source representations. We are interested in whether target-side prototypes can be used to enhance compositional generalization and/or further improve Proto-Transformer. To answer this question, we first extract prototypes from the decoder and incorporate them back to the decoder using the same mechanism of Proto-Transformer. The model can reduce CTERs to 25.93%/58.21%, largely inferior to using source-side prototypes (21.69%/51.84%). We also try to incorporate target-side prototypes to the decoder based on Proto-Transformer but it gives no noticeable improvement, achieving 21.65% and 51.71% CTERs, respectively. The underlying reason is connected a recent finding that translation already happens in the source encoding and the representations from encoders contain sufficient translation knowledge (Kasai et al., 2021; Xu et al., 2021a).

D Computational Cost

Our framework adds some computational overheads including extracting contextualized representations and conducting the clustering algorithm. The former only requires a single forward pass over the training set, merely amounting to a fraction of the cost of training for one epoch. Thanks to pytorch implementation of K-means algorithms which utilize GPU for faster matrix computations, the clustering is friendly for the large-scale datasets and it can be much faster with parallelization and more powerful hardware. Since the number of prototypes is a constant \(k\), the complexity of prototype-attention is \(O(kT)\), linear with respect to sequence length \(T\).

\(^3\)https://github.com/moses-smt/newsdev2015,
<table>
<thead>
<tr>
<th></th>
<th>CoGnition</th>
<th>IWSLT'14</th>
<th>IWSLT'15</th>
<th>WMT'16</th>
<th>WMT'17</th>
<th>JRC-Acquis</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>En-Zh</td>
<td>En-De</td>
<td>En-It</td>
<td>En-Vi</td>
<td>En-Ro</td>
<td>En-De</td>
</tr>
<tr>
<td>#Train</td>
<td>190k</td>
<td>157k</td>
<td>175k</td>
<td>133k</td>
<td>608k</td>
<td>608k</td>
</tr>
<tr>
<td>#Valid</td>
<td>10k</td>
<td>7k</td>
<td>1k</td>
<td>1.6k</td>
<td>2k</td>
<td>2k</td>
</tr>
<tr>
<td>#Test</td>
<td>10k</td>
<td>7k</td>
<td>0.9k</td>
<td>1.3k</td>
<td>2k</td>
<td>2k</td>
</tr>
<tr>
<td>N</td>
<td>8</td>
<td>12</td>
<td>10</td>
<td>8</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>k</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 7: Dataset statistics and hyper-parameters.