Understanding and Improving Hidden Representation for Neural Machine Translation

Guanlin Li[ϵ], **Lemao Liu**[λ], **Xintong Li**[∞], **Conghui Zhu**[ϵ *], **Tiejun Zhao**[ϵ], **Shuming Shi**[λ]
[ϵ]Harbin Institute of Technology, [λ] Tencent AI Lab
[∞] The Chinese University of Hong Kong
{epsilonlee.green, znculee}@gmail.com, {chzhu, tjzhao}@hit.edu.cn,
{redmondliu, shumingshi}@tencent.com

Abstract

Multilayer architectures are currently the gold standard for large-scale neural machine translation. Existing works have explored some methods for understanding the hidden representations, however, they have not sought to improve the translation quality rationally according to their understanding. Towards understanding for performance improvement, we first artificially construct a sequence of nested relative tasks and measure the feature generalization ability of the learned hidden representation over these tasks. Based on our understanding, we then propose to regularize the layer-wise representations with all tree-induced tasks. To overcome the computational bottleneck resulting from the large number of regularization terms, we design efficient approximation methods by selecting a few coarse-to-fine tasks for regularization. Extensive experiments on two widely-used datasets demonstrate the proposed methods only lead to small extra overheads in training but no additional overheads in testing, and achieve consistent improvements (up to +1.3 BLEU) compared to the state-of-the-art translation model.

1 Introduction

Neural machine translation (NMT) has witnessed great successes in recent years (Bahdanau et al., 2014; Wu et al., 2016). Current state-of-the-art (SOTA) NMT models are mainly constructed by a stacked neural architecture consisting of multiple hidden layers from bottom-up, where a classifier is built upon the topmost layer to solve the target task of translation (Gehring et al., 2017; Vaswani et al., 2017). Most works tend to focus on the translation performance of the classifier defined on the topmost layer, however, they do not deeply understand the learned representations of hidden layers. Shi et al. (2016) and Belinkov et al. (2017) attempt to understand the hidden representations through the lens of a few linguistic tasks, while Ding et al. (2017) and Strobelt et al. (2018) propose appealing visualization approaches to understand NMT models including the representation of hidden layers. However, employing the analyses to motivate new methods for better translation, the ultimate goal of understanding NMT, is not achieved in these works.

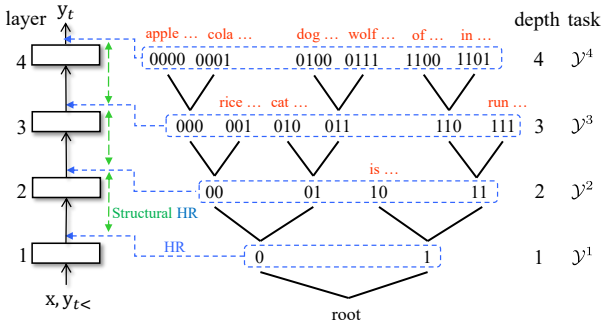


Figure 1: The structural hierarchical regularization framework. On the left is a 4-layer NMT decoder; on the right is a hierarchical clustering tree and the tree-induced relative tasks at every tree depth.

In our paper, we aim at understanding the hidden representation of NMT from an alternative viewpoint, and particularly we propose simple yet effective methods to improve the translation performance based on our understanding. We start from a fundamental question: what are the characteristics of the hidden representation for better translation modeling? Inspired by the lessons from transfer learning (Yosinski et al., 2014), we propose to empirically verify the argument: good hidden representation for a target task should be able to generalize well across *any* similar tasks. Unlike Shi et al. (2016) and Belinkov et al. (2017) who employ one or two linguistic tasks involving human annotated data to evaluate the feature generalization ability of the hidden representation,

* Conghui Zhu is the corresponding author.

Proceedings of NAACL-HLT 2019, pages 466–477
Minneapolis, Minnesota, June 2 - June 7, 2019.

which might make understanding bias to a specific task, we instead construct a nested sequence of *many* relative tasks with entailment structure induced by a hierarchical clustering tree over the output label space (target vocabulary). Each task is defined as predicting the cluster of the next token according to a given source sentence and its translation prefix. Similar to Yu et al. (2018), Zamir et al. (2018) and Belinkov et al. (2017), we measure the feature generalization ability of the hidden representation regarding each task. Our observations are (§2):

- The hidden representations learned by NMT indeed has decent feature generalization ability for the tree-induced relative tasks compared to the randomly initialized NMT model and a strong baseline with lexical features.
- The hidden representations from the higher layers generalize better across tasks than those from the lower layers. And more similar tasks have closer performances.

Based on the above findings, we decide to regularize and improve the hidden representations of NMT for better predictive performances regarding those relative tasks, in hope of achieving improved performance in terms of the target translation task. One natural solution is to feed all relative tasks to every hidden layer of the NMT decoder under the framework of multi-task learning. This may make the full coverage of the potential regularization effect. Unfortunately, this vanilla method is inefficient in training because there are more than one hundred task-layer combinations. [1] Based on the second finding, to approximate the vanilla method, we instead feed a single relative task to each hidden layer as a regularization auxiliary in a coarse-to-fine manner (§3.1). Furthermore, we design another regularization criterion to encourage predictive decision consistency between a pair of adjacent hidden layers, which leads to better approximated regularization effect (§3.2). Our method is simple to implement and efficient for training and testing. Figure 1 illustrates the representation regularization framework.

To summarize, our contributions are as follows:

- We propose an approach to understand hidden representation of multilayer NMT by measuring their feature generalization ability across relative tasks constructed by a hierarchical clustering tree.
- We propose two simple yet effective methods to regularize the hidden representation. These two methods serve as trade-offs between regularization coverage and efficiency with respect to the tree-induced tasks.
- We conduct experiments on two widely used datasets and obtain consistent improvements (up to +1.3 BLEU) over the current SOTA Transformer (Vaswani et al., 2017) model.

[1] There are about 22 tasks that we have constructed and 6 layers in SOTA NMT models (Vaswani et al., 2017).

2 Understanding Hidden Representation

In this section, we first introduce some background knowledge and notations of the multilayer NMT model. Then, we present a simple approach to better understand hidden representation through transfer learning. By analyzing the feature generalization ability, we draw some constructive conclusions which are used for designing regularization methods in Section 3.

2.1 Background and Notations

Suppose $\mathrm{x} = \langle \mathrm{x}_1, \cdots, \mathrm{x}_{|\mathrm{x}|} \rangle$ is a source sentence, i.e. a sequence of source tokens, and a target sentence $\mathrm{y} = \langle \mathrm{y}_1, \cdots, \mathrm{y}_{|\mathrm{y}|} \rangle$ is a translation of x, where each y_t in y belongs to $\mathcal{Y}$, the target vocabulary. A translation model minimizes the following chain-rule factorized negative log-likelihood loss:

$$\begin{aligned} \ell_{mle} &= -\log P(\mathrm{y} \mid \mathrm{x}; \theta) \\ &= -\sum_t \log P(\mathrm{y}_t \mid \mathrm{x}, \mathrm{y}_{<t}; \theta), \end{aligned} \tag{1}$$

where θ denotes the overall parameter of the translation model. According to Eq.(1), an alternative view of the translation problem can be cast to token-level stepwise **classification** (Daumé et al., 2009): predict the target token y_t given a context consisting of x and $\mathrm{y}_{<t} = \langle \mathrm{y}_1, \cdots, \mathrm{y}_{t-1} \rangle$ corresponding to each factor $P(\mathrm{y}_t \mid \mathrm{x}, \mathrm{y}_{<t}; \theta)$.

The SOTA multilayer NMT models parameterize $P(\mathrm{y}_t \mid \mathrm{x}, \mathrm{y}_{<t}; \theta)$ via powerful multilayer encoder and stacked layers of feature transformations $\langle h^1, \cdots, h^L \rangle$ at the decoder side:

$$P(\mathrm{y}_t \mid \mathrm{x}, \mathrm{y}_{<t}; \theta) = P(\mathrm{y}_t \mid \mathrm{x}, \mathrm{y}_{<t}, h^L; \theta), \tag{2}$$

where $h^l(\mathrm{x}, \mathrm{y}_{<t}) = \phi^l\big(\mathrm{x}, \mathrm{y}_{<t}; h^{l-1}; \theta\big)$ is the l^{th} hidden layer recursively defined by ϕ^l on h^{l-1}.

We also use $h^l(\mathrm{x}, \mathrm{y}_{<t})$ to represent the output hidden representation of layer l for a specific context. Note that, ϕ^l bears several types of instantiation and is an active area of research (Wu et al., 2016; Gehring et al., 2017; Vaswani et al., 2017).

2.2 Feature Generalization Ability of Hidden Representations

Inspired by feature transfer learning (Yosinski et al., 2014), we attempt to understand hidden representations of NMT by evaluating their generalization abilities across any tasks that are related to translation. There are some researchers who study hidden representations of NMT by using linguistic tasks such as morphology, named entity, part-or-speech or syntax (Shi et al., 2016; Belinkov et al., 2017, 2018). They typically rely on human annotated resources to train a model for each linguistic task, so their methods can not be used for languages which lack human annotations. Moreover, their considered tasks are too few to have a good coverage over task space for measuring transferability (Yu et al., 2018; Zamir et al., 2018), and their understanding results may bias to a specific task. As a result, to evaluate the feature generalization ability of hidden representation, we artificially construct plenty of relative tasks which do not employ any human annotation. This makes our evaluation approach more general.

Definition of the relative tasks Suppose $\mathcal{Y}^k$ denotes any partition (or clustering) regarding the output label space (target vocabulary) $\mathcal{Y}$. That is, $\mathcal{Y}^k$ is a set of subsets $\mathcal{Y}^k_i \subset \mathcal{Y}$ where $i = 1...|\mathcal{Y}^k|$, such that $\forall i, j, \mathcal{Y}^k_i \cap \mathcal{Y}^k_j = \emptyset$ and $\cup_i \mathcal{Y}^k_i = \mathcal{Y}$. We define the following relative task: given a context $\langle \mathrm{x}, \mathrm{y}_{<t} \rangle$, predict the subset or the cluster to which the t^{th} token y_t belongs in $\mathcal{Y}^k$, denoted as $\mathcal{Y}^k(\mathrm{y}_t)$. To simplify notation, we regard $\mathcal{Y}^k$ both as a relative task and as a partition.

It is clear that the above type of tasks are similar to the task of translation according to the description in Section §2.1. Furthermore, different k represents different relative task and thus we actually obtain a great many relative tasks in total. However, it is impossible to evaluate the hidden representation on all those tasks; moreover, due to relationship between tokens (Hu et al., 2016) in $\mathcal{Y}$, not all partitions are reasonable. As a consequence, motivated by the analysis of VC Dimension (Vapnik, 1995), we construct a sequence of nested partitions with an entailment structure: [2] $\mathcal{Y}^1 \preceq \cdots \preceq \mathcal{Y}^K$. The benefit is that a spectrum of task hardness can be constructed due to the increased partition or task cardinalities.

As a matter of fact, we instantiate the above nested partitions through brown clustering (Brown et al., 1992; Stratos et al., 2014) over $\mathcal{Y}$ to get a hierarchical clustering tree and then consider each tree depth along the tree as a partition representing a relative task $\mathcal{Y}^k$ (as shown in Figure 1). In the following experiments, we run brown clustering algorithm over a Ch⇒En dataset (§4) and construct a tree of English with depth 21. Without loss of generality, we regard the task $\mathcal{Y}^{22}$ at a virtual 22 depth of the tree as equivalent to the translation task $\mathcal{Y}$. Actually, $\mathcal{Y}$ and $\mathcal{Y}^{22}$ have the same cardinality but are different in definition. [3]

Evaluating generalization We use multi-class logistic regression to fit the layer-wise hidden representation learned by a well-trained 6-layer Transformer (Vaswani et al., 2017) over each training instance $\langle \mathrm{x}, \mathrm{y}_{<t} \rangle$. Specifically, given a context $\langle \mathrm{x}, \mathrm{y}_{<t} \rangle$, for each task $\mathcal{Y}^k$ and a hidden representation $h^l(\mathrm{x}, \mathrm{y}_{<t})$ of this context, which is fixed as constant, we predict the cluster $\mathcal{Y}^k(\mathrm{y}_t)$ according to the following probability:

$$P\big(\mathcal{Y}^k(\mathrm{y}_t) \mid h^l(\mathrm{x}, \mathrm{y}_{<t}); \theta^l_{\mathcal{Y}^k}\big), \tag{3}$$

where $\theta^l_{\mathcal{Y}^k}$ is the parameter of the logistic regression model for task $\mathcal{Y}^k$ at l^{th} layer. The difference between Eq.(3) and Eq.(2) is that the former is the linear model parameterized by $\theta^l_{\mathcal{Y}^k}$ while the latter is the NMT model parameterized by θ.

Since there are $L = 6$ layers in Transformer's decoder and $K = 22$ relative tasks, we have more than one hundred such linear models defined with Eq. (3) in total. Therefore, it is costly to train them independently. Since the loss for each linear model is convex, joint training leads to exactly the same optimum as independent training and thus we employ mini-batch stochastic gradient descent to minimize the joint loss as follows:

$$-\sum_k \sum_l \sum_t \log P\big(\mathcal{Y}^k(\mathrm{y}_t) \mid h^l(\mathrm{x}, \mathrm{y}_{<t}); \theta^l_{\mathcal{Y}^k}\big). \tag{4}$$

[2] Here what we mean entailment relation ($\preceq$) between two partitions $\mathcal{Y}^k$ and $\mathcal{Y}^{k+1}$ is: $\forall \mathcal{Y}^{k+1}_i, \exists! \mathcal{Y}^k_j, s.t. \mathcal{Y}^{k+1}_i \subseteq \mathcal{Y}^k_j$.

[3] Please refer to Appendix A for detailed preprocessing of the tree to get nested partitions.

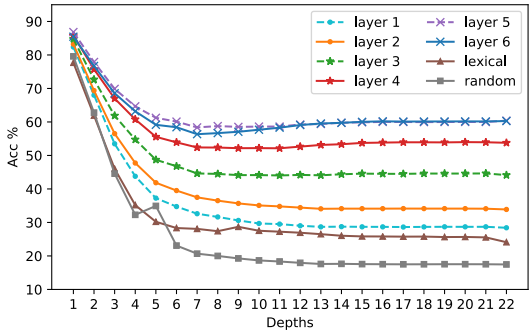


Figure 2: The transfer learning performances.

After training, we fix each $\theta^l_{\mathcal{Y}^k}$ and then measure the feature generalization ability of each h^l by validating on the task $\mathcal{Y}^k$ regarding a heldout dataset, following Yu et al. (2018). For validation, we report accuracy on a heldout dataset through the strategy of maximum a posteriori (MAP). [4]

Analysis To figure out how good the learned hidden representations are, we consider two **baselines** to extract features regarding each context $\langle \mathrm{x}, \mathrm{y}_{<t} \rangle$ to train logistic regression models for comparison. For the first baseline, the features of the context are the hidden representations from the last layer of a randomly initialized Transformer; for the second, the features are derived by lexical feature templates, which include the source-side bag-of-words (BOW) features and target-side BOW features indexed by relative positions of y_t's previous with up to m (Markov length) tokens. [5]

As shown in Figure 2, the lexical baseline delivers comparable accuracies for fine-grained tasks with respect to well learned Transformer's first layer, thanks to its discriminant ability with abundant lexical features. For example, its accuracy reaches about 26% for the task with cardinality $|\mathcal{Y}^{21}|$. The random baseline performs worse for tasks with cardinality $|\mathcal{Y}^{8}|$, which indicates that random representations in NMT have limited generalization abilities to fine-grained tasks as expected. The well-trained low-layer hidden representations yield much higher accuracies than the random baseline and are even better than the lexical baseline. This shows that the hidden representations from a well-trained NMT have good generalization abilities across relative tasks. In addition, as the layer goes up, the performance of hidden representations increase significantly over different relative tasks, which clearly demonstrates that more complex neural architecture leads to stronger expressibility. This provides a quantitative evidence to support the statement in Bengio et al. (2009), Goodfellow et al. (2016).

[4] The accuracy is measured by whether $\arg\max_{z \in \mathcal{Y}^k} P(z \mid h^l(\mathrm{x}, \mathrm{y}_{<t}); \theta^l_{\mathcal{Y}^k}) = \mathcal{Y}^k(\mathrm{y}_t)$.

[5] Please refer to Appendix B for more details are.

3 Structural Hierarchical Regularization

In this section, we propose two simple methods, which respect the above findings, to enhance the hidden representations in NMT such that they generalize well across those relative tasks.

3.1 Hierarchical Regularization

A natural method to improve feature generalization of hidden representation is to jointly train the target task with all relative tasks for all hidden layers, which we call full-coverage method. As mentioned in Section §2.2, this method will lead to training more than one hundred tasks ($K \times L$) in total, where K denotes the depth of the hierarchical clustering tree (aka. the number of tasks) and L the number of hidden layers. Unfortunately, since each task involves a softmax operation which may be the computation bottleneck for the task $\mathcal{Y}^k$ with large cardinality, this method is inefficient for training.

As a solution to approximate the potential regularization effect of the full-coverage method, we confine each hidden layer to engage in a single relative task. Motivated by the observation that representations from higher layers have better expressibility than lower layers, as claimed in §2.2, we instead employ a coarse-to-fine strategy to select one task for each layer: finer-grained tasks for higher layers while coarser-grained task for lower layers. Specifically, suppose $1 \leq s(l) \leq K$ is the selected index regarding task $\mathcal{Y}^{s(l)}$ for the l^{th} layer, then it subjects to $s(l) < s(l+1)$ for each l. In addition, to encourage the diversity among the selected L tasks, we require $s(l+1) - s(l)$ to be large enough for all l. Formally, the loss of the hierarchical regularization (HR) method is:

$$\ell_{hr} = -\sum_{l}\sum_{t} \log P(\mathcal{Y}^{s(l)}(\mathrm{y}_t) \mid \mathrm{x}, \mathrm{y}_{<t}, h^l; \theta, \theta^l_{\mathcal{Y}^{s(l)}}), \quad (5)$$

where $P(\mathcal{Y}^{s(l)}(\mathrm{y}_t) \mid \mathrm{x}^j, \mathrm{y}^j_{<t}, h^l; \theta, \theta^l_{\mathcal{Y}^{s(l)}})$ is similar to Eq. 3 except that it treats the parameters θ in NMT as parameters besides $\theta^l_{\mathcal{Y}^{s(l)}}$. Compared to Eq. 4, it includes fewer terms for summation.

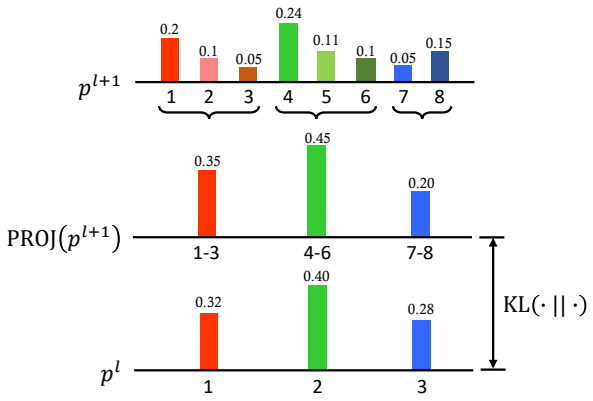


Figure 3: The conceptual graph of the KL consistency loss in SHR. Here, the multinomial probability vector p^l is calculated through $P(\cdot \mid \mathrm{x}, \mathrm{y}_{<t}, h^l; \theta, \theta^l_{\mathcal{Y}^{s(l)}})$.

3.2 Structural Hierarchical Regularization

The HR method is very simple and computationally efficient, however, using one task to regularize a layer may not be a good approximation of the full-coverage method, since HR method might lead to inconsistent decisions for two different layers, which is formalized through the following entailment structure as introduced in Section 2.2:

$$\arg\max_{\mathrm{z}\in\mathcal{Y}^{s(l_1)}} P(\mathrm{z} \mid \mathrm{x}, \mathrm{y}_{<t}, h^{l_1}; \theta, \theta^{l_1}_{\mathcal{Y}^{s(l_1)}}) \supset \arg\max_{\mathrm{z}\in\mathcal{Y}^{s(l_2)}} P(\mathrm{z} \mid \mathrm{x}, \mathrm{y}_{<t}, h^{l_2}; \theta, \theta^{l_2}_{\mathcal{Y}^{s(l_2)}}), \quad (6)$$

where $s(l)$ is the selected task for the l^{th} layer by HR, $1 \le l_1 < l_2 \le L$ and $P(\mathrm{z}|\mathrm{x}, \mathrm{y}_{<t}, h^l; \theta, \theta^l_{\mathcal{Y}^{s(l)}})$ is similar to Eq. (3) for the task $\mathcal{Y}^{s(l)}$ and l^{th} layer except that it does not treat the NMT parameters θ as constant. However, it always occurs on the training data that $\mathcal{Y}^{s(l_1)}(\mathrm{y}_t) \subset \mathcal{Y}^{s(l_2)}(\mathrm{y}_t)$.

To alleviate this inconsistency issue for better approximating the full-coverage method, we leverage the above structural property by adding another regularization term. Firstly, we project the distribution $P(\cdot|\mathrm{x}, \mathrm{y}_{<t}, h^l; \theta, \theta^l_{\mathcal{Y}^{s(l)}})$ into the domain of $\mathcal{Y}^{s(l-1)}$. Then we calculate KL divergence between the projected distribution and $P(\cdot|\mathrm{x}, \mathrm{y}_{<t}, h^{l-1}; \theta, \theta^{l-1}_{\mathcal{Y}^{s(l-1)}})$. Figure 3 illustrates the idea. Since it is inefficient to consider all pairs of l_1 and l_2, so we instead consider the consistency between all adjacent layers. Formally, we obtain

Method	# Param.		Speed	
	Zh⇒En	En⇒De	Train	Test
Baseline	90.2M	88.0M	1.82	1.33
FHR	+60.9M	+63.2M	0.72	
HR	+3.2M	+3.1M	1.65	
SHR	+3.2M	+3.1M	1.40	

Table 1: Train and test efficiency comparison, measured by steps/s (second) and sentences/s respectively.

the following loss function:

$$\ell_{shr} = \ell_{hr} + \frac{1}{L-1}\sum_l \mathrm{KL}\Big(P(\cdot|\mathrm{x}, \mathrm{y}_{<t}, h^l; \theta, \theta^l_{\mathcal{Y}^{s(l)}}) \,||\, \mathrm{PROJ}\big[P(\cdot|\mathrm{x}, \mathrm{y}_{<t}, h^{l+1}; \theta, \theta^{l+1}_{\mathcal{Y}^{s(l+1)}})\big]\Big), \quad (7)$$

where PROJ is the projection defined in Figure 3, and other notations are defined as before.

We call the above regularization as structural hierarchical regularization (SHR) since it takes advantage of the structure of the tree. In our experiments, we add HR (Eq.(5)) and SHR (Eq.(7)) losses respectively into the negative log-likelihood regarding Eq. (1) for training all parameters θ and $\theta^l_{\mathcal{Y}^{s(l)}}$. One of our advantage is that we only use θ for testing and thus our testing is as efficient as that for the baseline NMT model.

4 Experiments

4.1 Setup

We conduct experiments on two widely-used corpora. We choose from the LDC corpora about 1.8M sentence pairs for Zh⇒En translation with word-level vocabulary of 30k for both languages. We use the WMT14 En⇒De task which consists 4.5M sentence pairs and the vocabulary is built by joint BPE with 32k merging operations. Besides the baseline, we also conduct experiments on 3 regularization variants:

- **Baseline**: the Transformer base model proposed in Vaswani et al. (2017).
- **FHR**: fine-grained HR based Transformer, which adopts the original label space as task for all selected layers for regularization. This variant is used to demonstrate that low layers which are weak in expressibility can mess up hard tasks which are unsuitable to learn.
- **HR** and **SHR**: as proposed in Section 3.

Method	MT02	MT03	MT04	MT05	MT06	MT08	Avg.
Zhao et al. (2018)	N/A	44.98	45.51	43.93	43.95	33.33	42.34
Baseline	46.08	44.09	46.50	44.45	45.26	37.10	43.48
FHR	45.46	43.56	47.51	44.00	45.45	37.22	43.58
HR	46.28	44.04	47.80	44.56	45.56	38.17	44.08
SHR	**47.05**	**44.80**	**48.15**	**45.55**	**46.30**	**39.02**	**44.78**

Table 2: BLEU comparison on the LDC dataset.

Choice of relative tasks Based on the heuristics in Section 3.1, we first choose the task with the largest cardinality from the hierarchical clustering tree without the virtual depth, because this task is most related to translation (close cardinalities). Then we balance task diversity through a 5 times cardinality difference between tasks from the previous chosen task. As a result, we can obtain 4 tasks with $s(l) = 5, 8, 11, 20$ for the Zh⇒En task and $s(l) = 5, 7, 10, 21$ for the En⇒De task, where $l = 2, 3, 4, 5$ of the 6-layer decoder. [6]

4.2 Efficiency Comparison

Table 1 summarizes the total number of parameters for the baseline and 3 regularization variants. As in Eq. (5), HR introduces extra parameters compared with the baseline. Besides, calculating the second term in Eq. (7) requires modest overheads. Therefore, training our SHR is slower than training the baseline. Although the proposed HR and SHR introduce extra parameters during training, they do not involve them during testing and thus testing is as efficient as the baseline.

4.3 Translation Quality on Zh⇒En Dataset

Table 2 shows the evaluation results of the baseline and 3 regularization variants on the Zh⇒En dataset. Since there are no recent work reporting Transformer's performance on this dataset, we choose a recurrent SOTA model to show that our baseline is already better than it, which is a common knowledge that Transformer can outperform recurrent NMT models. Our HR method surpasses the baseline 0.6 BLEU point, while the SHR method can improve upon HR by about a further 0.8 point, namely about 1.4 points over the baseline. Interestingly, the FHR method only performs on par with baseline, which indicates that forcing low layers to learn fine-grained tasks will not lead to beneficial intermediate representations since they struggle to learn a well-structured representation space. This matches the finding in Section 2: low layers may not be expressible enough to perform well on tasks with large cardinalities.

[6] Please refer to Appendix C for detailed information.

4.4 Analyses on Zh⇒En Dataset

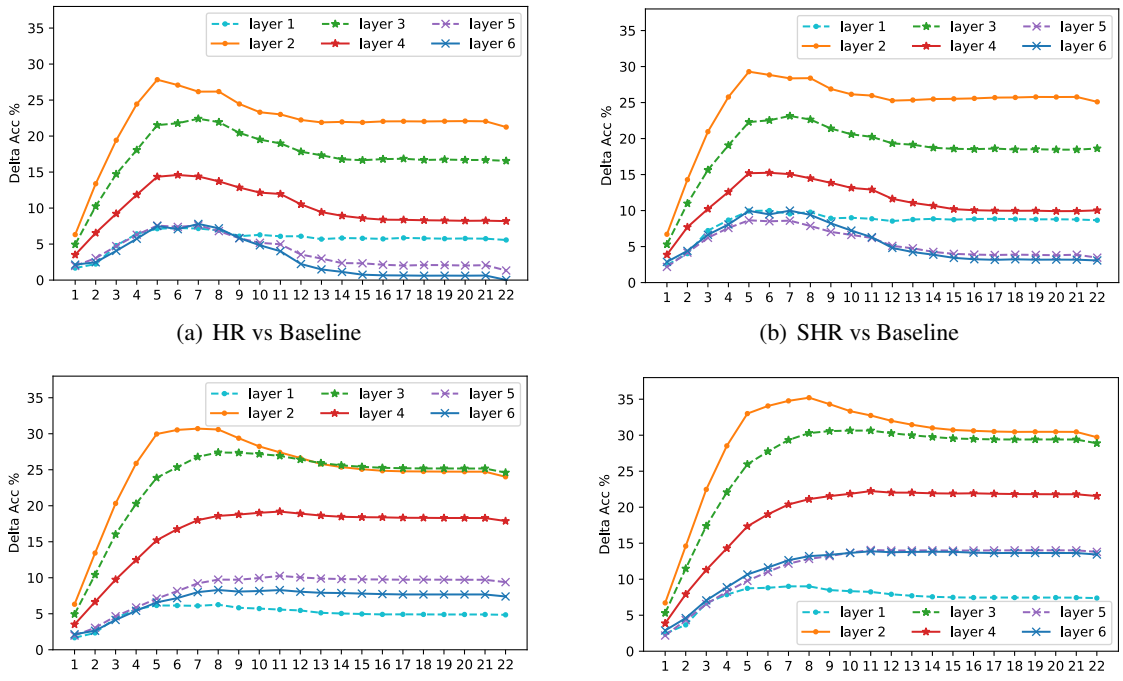
In the following, we conduct several quantitative experiments to demonstrate the advantages of our proposed two regularization methods over the baseline. Note that, since we need to guarantee that the decoded sequence has the same length with the reference for one-by-one token comparison, the following experiments are all conducted with teacher forcing and greedy decoding.

4.4.1 Better Feature Generalization Ability

In the same manner as Section 2, we learn softmax weights for all relative tasks by fixing model weights learned by HR and SHR methods. Figure 4(a), (b) show the Δ feature generalization ability (absolute accuracy difference) of HR and SHR over baseline. Since layer 1 is not selected as the regularized layer, no significant gap is observed. However, since layer 1 is close to the loss directly imposed on layer 2, improvements about 5% and 8% are obtained. Since in the baseline, layer 5, 6 are already close or with the ultimate fine-grained loss, HR method shows very small gain. But our SHR method can still improve about 4% absolute points. Except for layer 1, it is also evident to see larger gaps (more than 20%) at lower layers than higher layers due to the fact that lower layers, which are distant from the topmost loss in the baseline, require more supervision signals to shape their latent representation space.

4.4.2 Improved Decision Consistency

We measure decision consistency for a specific layer and decision consistency between a pair of layers using two metrics. The first metric is measured by conditional accuracy, which is the possibilities of the classifier parameterized by $\theta^l_{\mathcal{Y}^k}$ correctly predicting $\mathcal{Y}^k(\mathrm{y}_t)$ if the classifier parameterized by $\theta^l_{\mathcal{Y}^{k'}}$ correctly predicts $\mathcal{Y}^{k'}(\mathrm{y}_t)$ for any



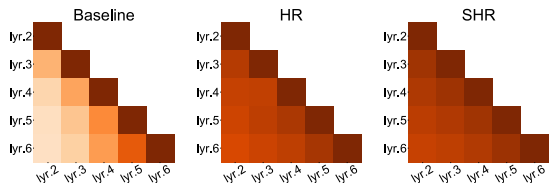
(a) HR vs Baseline (b) SHR vs Baseline

(c) HR vs Baseline (d) SHR vs Baseline

Figure 4: (a), (b) are the Δ feature generalization ability (absolute accuracy difference) of HR and SHR method compared to baseline; (c), (d) are the conditional absolute accuracy difference of HR and SHR over baseline.

$k' < k$. The second metric is measured by the counts of consistent decision pairs between any pair of regularized layers as defined in Eq. (6).

Figure 4(c), (d) shows the absolute conditional accuracy difference of our HR and SHR over baseline. In accordance with the observations in previous subsection, except for layer 1, other layers show significant gains (HR more than 7%, SHR more than 10%) over baseline. Decision consistency for each layer proves the well-shaped layer-wise representation and potentially paves the way for better inter-layer decision consistency.



(a) (b) (c)

Figure 5: The consistency correlation between different regularized layers (lyr.2 to lyr.6) of the baseline and our two regularization methods.

Figure 5 illustrates the consistency counts between any regularized layer pairs, including those without KL-based regularization. Deeper color represents more consistency counts. It is evident that the baseline has a very poor consistency between any layers. Our HR method is almost 2 times better, and the SHR obtains further improvement. A better decision consistency can couple the decision between relative tasks, so that by reaching a high accuracy on easier tasks can benefit the harder ones. Another interesting observation is that non-adjacent layers without KL loss also obtain significant improvements on decision consistency, because the KL term is actually transitive between layers where the predictive distributions are in accordance with the tree structure.

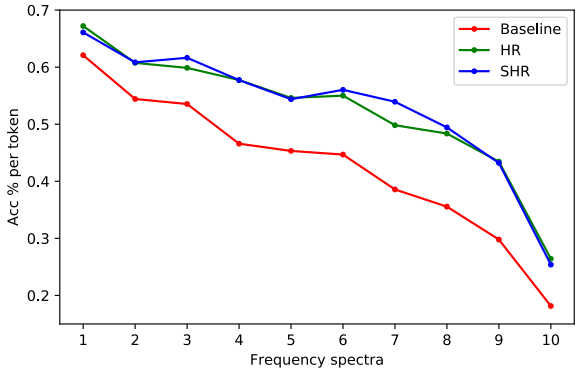


Figure 6: Development set accuracy among the baseline and our proposed regularization variants over different word frequency bins.

4.4.3 Promoted Low-Frequency Word Performance

In this subsection, we clarify that the coarse-to-fine regularized representations can also benefit low-frequency words. We divide the vocabulary into ten equally-sized bins, and summarize token accuracy for each bin over the development set. As shown in Figure 6, the x-axis represents the frequency spectra, that is, we sort the bins by word frequency from rank 1 (the most frequent words) to 10 (the rare words). We can see that both HR and SHR methods demonstrate a gradually increased gap over the baseline as the word frequency decreases, which means our methods become better for less frequent word bins. However the gap shrinks at the 10^{th} bin. This may be the fact that for those words that appears with less than 50 counts, both methods are helpless.

For baseline, it is hard to train well-shaped hidden representations for low-frequent words; in addition, due to the distance between the loss and the low layers, it is also hard to train weights due to the unstable gradient signal. By adding our regularization terms, every level of the multilayer decoder will receive supervision signals directly and lower layers will receive coarser grained thus higher frequency signals to shape their representations.

4.5 Translation Quality on En⇒De Dataset

Table 3 shows the evaluation results of the baseline and the 3 regularization variants on the En⇒De dataset. Notice that we use the base model while Chen et al. (2018) and Ott et al. (2018) use big models. The FHR method still does not show significant improvement over the baseline (less than 0.2 BLEU point), which verifies the hypothesis that we make by analyzing the Zh⇒En results. Our HR method is already stronger than Chen et al. (2018) which uses a multilayer RNN as decoder. Compared to the current state-of-the-art in Ott et al. (2018) who utilize huge batch size and over 100 GPUs on the Transformer big model, our SHR method can be on par with them. This comparison indicates that better regularized hidden representations can be potentially powerful than increasing model capacity when using the same optimization method.

Method	MT13	MT14
Chen et al. (2018)	N/A	28.49
Ott et al. (2018)	**26.70**	**29.30**
Baseline	25.99	27.75
FHR	26.10	27.91
HR	26.48	28.87
SHR	**26.64**	**29.18**

Table 3: BLEU comparison on the WMT14 dataset. Here MT13 and MT14 denote newstest2013 and newstest2014, which are used as development and test set respectively.

5 Related Work

Since the dawn of NMT, many works have proposed for understanding what has been encoded in the learned hidden representations. Shi et al. (2016) are the first to investigate source syntax encoded in source hidden representations. Similarly, Belinkov et al. (2017) and Belinkov et al. (2018) give detailed analyses of both encoder and decoder's learned knowledge about part-of-speech and semantic tags at different layers. Unlike those works that employ one or two linguistic tasks, we instead construct plenty of artificial tasks without any human annotations to analyze the hidden representations. This makes our approach more general and may potentially lead to less biased conclusions.

Based on our understanding of the hidden representations, we further develop simple methods to improve NMT through representation regularization. Many works regularize NMT with lexical knowledge such as BOW (Weng et al., 2017) and morphology (Niehues and Cho, 2017; Zaremoodi et al., 2018), or syntactic knowledge (Kiperwasser and Ballesteros, 2018; Eriguchi et al., 2017). One significant difference is that we take into account the structure among plenty of artificial tasks and design a well motivated regularization term to encourage the structural consistency of tasks, which further improves NMT performance. In addition, our coarse-to-fine way to select tasks for regularization is also inspired by recent works using a coarse-to-fine mechanism for learning better word embeddings in NMT (Zhang et al., 2018) and predicting intermediate solutions for semantic parsing (Dong and Lapata, 2018).

6 Conclusion

In this work, we present a simple approach for better understanding NMT learned layer-wise representations with transfer learning over plenty of artificially constructed relative tasks. This approach

is general as it requires no human annotated data, only demanding target monolingual corpus. Based on our understanding, we propose two efficient yet effective methods for representation regularization which further pushes forward the SOTA NMT performances. In the future, we want to dig deeply into the subspace regularities of the learned representations for more fine-grained understanding.

Acknowledgements

The authors would like to first thank all the anonymous reviewers for their critical suggestion and valuable experimental advice. The authors would also like to thank Yong Jiang for discussion; Shangchen Zhou for better figure design; Haozhe Xie, Chaoqun Duan, Xin Li, Ziyi Dou and Mengzhou Xia for proof reading. Tiejun Zhao is supported by National Key RD Program of China Project 2017YFB1002102.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? *arXiv preprint arXiv:1704.03471*.

Yonatan Belinkov, Lluís Màrquez, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2018. Evaluating layers of representation in neural machine translation on part-of-speech and semantic tagging tasks. *arXiv preprint arXiv:1801.07772*.

Yoshua Bengio et al. 2009. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Niki Parmar, Mike Schuster, Zhifeng Chen, et al. 2018. The best of both worlds: Combining recent advances in neural machine translation. *arXiv preprint arXiv:1804.09849*.

Hal Daumé, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Machine learning*, 75(3):297–325.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2018. Universal transformers. *arXiv preprint arXiv:1807.03819*.

Yanzhuo Ding, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1159. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. *arXiv preprint arXiv:1805.04793*.

Akiko Eriguchi, Yoshimasa Tsuruoka, and Kyunghyun Cho. 2017. Learning to parse and translate improves neural machine translation. *arXiv preprint arXiv:1702.03525*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*, volume 1. MIT press Cambridge.

Hexiang Hu, Guang-Tong Zhou, Zhiwei Deng, Zicheng Liao, and Greg Mori. 2016. Learning structured inference neural networks with label relations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2960–2968.

Stanisław Jastrzebski, Devansh Arpit, Nicolas Ballas, Vikas Verma, Tong Che, and Yoshua Bengio. 2017. Residual connections encourage iterative inference. *arXiv preprint arXiv:1710.04773*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Eliyahu Kiperwasser and Miguel Ballesteros. 2018. Scheduled multi-task learning: From syntax to translation. *arXiv preprint arXiv:1804.08915*.

Marc T Law, Jake Snell, Amir-massoud Farahmand, Raquel Urtasun, and Richard S Zemel. 2018. Dimensionality reduction for representing the knowledge of probabilistic models.

Jan Niehues and Eunah Cho. 2017. Exploiting linguistic resources for neural machine translation using multi-task learning. *arXiv preprint arXiv:1708.00993*.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation*, pages 1–9, Belgium, Brussels. Association for Computational Linguistics.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural mt learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534.

Karl Stratos, Do-kyum Kim, Michael Collins, and Daniel J Hsu. 2014. A spectral algorithm for learning class-based n-gram models of natural language. In *UAI*, pages 762–771. Citeseer.

H. Strobelt, S. Gehrmann, M. Behrisch, A. Perer, H. Pfister, and A. M. Rush. 2018. Seq2Seq-Vis: A Visual Debugging Tool for Sequence-to-Sequence Models. *ArXiv e-prints*.

Vladimir N. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Heidelberg.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Rongxiang Weng, Shujian Huang, Zaixiang Zheng, Xinyu Dai, and Jiajun Chen. 2017. Neural machine translation with word predictions. *arXiv preprint arXiv:1708.01771*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1206–1215. Association for Computational Linguistics.

Amir R Zamir, Alexander Sax, , William B Shen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. 2018. Taskonomy: Disentangling task transfer learning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Poorya Zaremoodi, Wray Buntine, and Gholamreza Haffari. 2018. Adaptive knowledge sharing in multi-task learning: Improving low-resource neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 656–661.

Zhirui Zhang, Shujie Liu, Mu Li, Ming Zhou, and Enhong Chen. 2018. Coarse-to-fine learning for neural machine translation. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 316–328. Springer.

Yang Zhao, Jiajun Zhang, Zhongjun He, Chengqing Zong, and Hua Wu. 2018. Addressing troublesome words in neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 391–400. Association for Computational Linguistics.

A Hierarchical Clustering Tree Induced Relative Tasks

In this appendix, we demonstrate a more detailed introduction to the partition of $\mathcal{Y}$, the hierarchical clustering tree constructed by $\mathcal{Y}$ and the tree-induced relative tasks, which are introduced in Section 2.2, through an example.

Suppose our vocabulary $\mathcal{Y}$ only consists of a few words, that is, $\mathcal{Y} = \{\text{cat, dog, run, jump, is}\}$. Any partition of $\mathcal{Y}$ denoted as $\mathcal{Y}^k$ is a set of subsets of $\mathcal{Y}$. As shown in Figure 7, this partition is actually $\{ \{ \text{ cat, dog } \}, \{ \text{ jump, run } \}, \{ \text{ is } \} \}$.

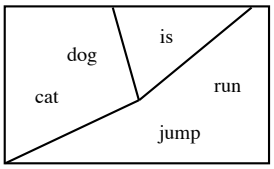


Figure 7: One example partition of the vocabulary $\mathcal{Y}$.

Then suppose the constructed hierarchical clustering tree looks like the tree in Figure 8(a). In this tree, not all the leaves are at the same depth, so we left-branches the leaves with {cat, dog} at depth 1 to stretch to depth 2. Then by adding a virtual level at depth 3, we can construct a relative task with the same cardinality as the fine-grained translation task $\mathcal{Y}$, as shown in Figure 9(b).

B Lexical Feature-based Baseline

In this appendix, we describe the lexical feature-based baseline that we use in Section 2.2 to compare with the layer-wise representations learned by Transformer. There are two types of feature template: a) the source-side bag-of-word (BOW) features; and b) the target-side order-aware BOW features. Specifically, given a context $\langle \mathrm{x}, \mathrm{y}_{<t} \rangle$, we extract features according to the above templates:

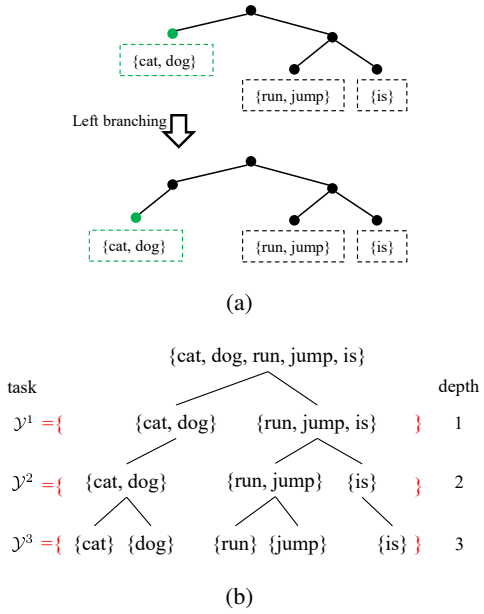
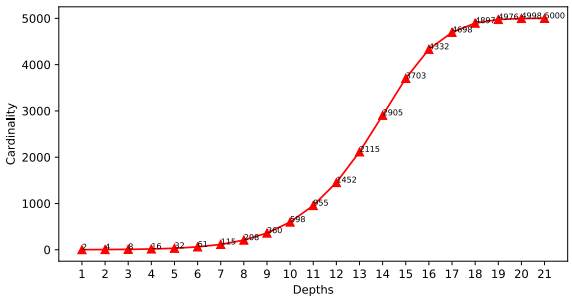
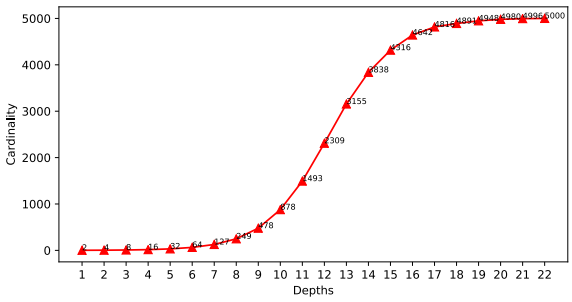


Figure 8: (a) The original hierarchical clustering tree, with left branching to have all leaves at the same tree depth; (b) The hierarchical clustering tree with depth 2, and a virtual task is constructed at depth 3.



(a) Task cardinalities for English (Zh⇒En).



(b) Task cardinalities for German (En⇒De).

Figure 9: Task cardinalities.

- the BOW representation of the source sentence x: that is, if the source vocabulary is $\mathcal{X}$, the source BOW feature vector has length of $|\mathcal{X}|$, with each entry the appearances of that token in x.
- the order-aware BOW representation within k-Markov dependency chain: that is, we extract feature from $\mathrm{y}_{t-k:t-1}$ by considering both the token identity and the relative distance of that token to the predicted one y_t. This feature template constructs a feature vector of $k \times |\mathcal{Y}|$ entries ($\mathcal{Y}$ the target vocabulary), with each entry set to 1 when appears in the chain, otherwise 0.
- the order-unaware BOW representation outside the k-Markov dependency chain, that is, we extract feature from the $\mathrm{y}_{0:t-k-1}$ with the same philosophy of the source-side BOW feature and obtain a feature vector of size $\mathcal{Y}$.

So in total, the feature vector extracted from the context $\langle \mathrm{x}, \mathrm{y}_{<t} \rangle$ has a size $(|\mathcal{X}| + k \times |\mathcal{Y}|)$, which will be around 300k if the vocabulary size is around 30k and the Markov order $k = 10$.

C Detailed Experiment Information

In this appendix, we describe the detailed experiment information: the construction of the hierarchical clustering tree (Brown et al., 1992; Stratos et al., 2014), the model configuration, and the training details.

We construct two hierarchical clustering trees for the two target languages, English (Zh⇒En) and German (En⇒De) in our experiments, with Percy Liang's Brown Clustering algorithm implementation. [7] In both languages, we set the number of clusters to 5000 (a hyperparameter in the algorithm, $c = 5000$), that is, we will obtain trees with 5000 leaves. We set c to 5000 since the total vocabulary of the two languages are around 30k, and 5000 clusters will get a token coverage of 6 (30k/5000=6) for each leave if the clusters are balanced. By using left-branching introduced in Appendix A, we can finally get two trees with every depths as a relative task. The statistics of each relative task's cardinality for each tree are shown in Figure 9(a) and (b) respectively. For selecting the depths in a coarse-to-fine manner, we follow the heuristics mentioned in Section 3.1, that is, we select depths which are diverse enough so as to have better coverage of all the relative tasks. Specifically, we follow a quotient between two adjacent selected tasks of 5, and select from the task which has the cardinality of 5000, then we select tasks of cardinalities around 1000, 200, 40 respectively.

[7] https://github.com/percyliang/brown-cluster.

Method	MT02	MT03	MT04	MT05	MT06	MT08	Avg.
Baseline	46.08	44.09	46.50	44.45	45.26	37.10	43.48
L2	46.67	44.50	47.35	45.02	46.20	38.43	44.30
L3	46.40	44.65	46.90	45.02	45.95	37.92	44.08
L4	46.35	44.30	46.97	45.10	46.06	37.31	43.95
L5	46.29	44.57	46.97	44.75	45.45	37.74	43.89
HR	46.28	44.04	47.80	44.56	45.56	38.17	44.08
SHR	**47.05**	**44.80**	**48.15**	**45.55**	**46.30**	**39.02**	**44.78**

Table 4: BLEU comparison on the LDC dataset with independently regularized layers.

For the Zh⇒En dataset, we select tasks at depths 5, 8, 11, 20 with cardinalities 32, 208, 955, 5000; for the En⇒De dataset, we select tasks at depths 5, 7, 10, 21 with cardinalities 32, 127, 878, 5000.

The model configuration strictly follows that of the base model in Vaswani et al. (2017) with word embedding size of 512, hidden size of 512, feed-forward projection size of 2048, layer and head number of 6 and 8. The dropout rates of the embedding, attention and residual block are all set to 0.1. All architectural settings are in accordance with the base model of Vaswani et al. (2017).

We use Adam (Kingma and Ba, 2014) as the default optimizer with an initial learning rate of 0.0001. Training batch size is set to 8192×4 tokens per batch, that is, we use data parallelism with 4 P40 or M40 GPUs and 8192 tokens per GPU. We train the Zh⇒En models from scratch for about 240k iterations about 2 weeks on 4 M40 GPUs for both the baseline and the 3 regularization variants. For the En⇒De models, we first attempt to train all the methods from scratch up to 200k iterations, but do not see significant improvement on BLEU score (around 0.6 points on test). So we use the pretrained baseline to initialize our proposed methods, and further train them for about 200k iterations, which results in the reported improvement in Section 4.5.

D Experiments of Independent Layer Regularization on Zh⇒En

As suggested by the reviewers, we conduct independent layer regularization by imposing a relative task with cardinality of 5000 on layer 2 to 5 with the six layer Transformer. The performances are demonstrated in Table 4. It seems that independent layer regularization can as well brings about descent improvements over the baseline. And regularizing layer 2 of the decoder performs better than our HR method. It is surprising that lower layers are more urgent to be regularized than higher layers. This phenomenon may raise the question that how on earth the intermediate layer representations help with final prediction. One hypothesis may be drawn from our paper is that: in baseline, the coherence among the layer-wise representations may be weak so that some non-linear transformations from lower layer may not lead to essential predictive power of the final layer representation. And by using KL divergence to externally constrain their decision consistency may take better advantage of lower layers. As one of the reviewer pointed out, Universal Transformer (Dehghani et al., 2018) with base model's hyperparameters achieves 28.90 on WMT14 En⇒De newstest14, with 1/6 enc-dec parameters (not considering embeddings). Its architectural inductive bias is motivated from iterative refinement of the layer-wise representation so the decoder at each time step builds an RNN like reasoning process to refine upon previous layer's representation. We think this might be a more effective inductive bias in ResNet architecture which is adopted by Transformer, since Jastrzebski et al. (2017) provides evidence that ResNet does iterative representation inference (residual as refined quantity). Future directions may include relating the dimension reduced representations (Law et al., 2018) to the coarse-to-fine structural bias, or experiments on Universal Transformer architecture to probe its learned representations.