Neuron Interaction Based Representation Composition for Neural Machine Translation

Jian Li,[1,2] Xing Wang,[3] Baosong Yang,[4] Shuming Shi,[3] Michael R. Lyu,[1,2] Zhaopeng Tu[3*]
[1]Department of Computer Science and Engineering, The Chinese University of Hong Kong
[2]Shenzhen Research Institute, The Chinese University of Hong Kong
{jianli, lyu}@cse.cuhk.edu.hk
[3]Tencent AI Lab
{brightxwang, shumingshi,zptu}@tencent.com
[4]University of Macau
nlp2ct.baosong@gmail.com

Abstract

Recent NLP studies reveal that substantial linguistic information can be attributed to single neurons, i.e., individual dimensions of the representation vectors. We hypothesize that modeling strong interactions among neurons helps to better capture complex information by composing the linguistic properties embedded in individual neurons. Starting from this intuition, we propose a novel approach to compose representations learned by different components in neural machine translation (e.g., multi-layer networks or multi-head attention), based on modeling strong interactions among neurons in the representation vectors. Specifically, we leverage bilinear pooling to model pairwise multiplicative interactions among individual neurons, and a low-rank approximation to make the model computationally feasible. We further propose extended bilinear pooling to incorporate first-order representations. Experiments on WMT14 English⇒German and English⇒French translation tasks show that our model consistently improves performances over the SOTA TRANSFORMER baseline. Further analyses demonstrate that our approach indeed captures more syntactic and semantic information as expected.

Introduction

Deep neural networks (DNNs) have advanced the state of the art in various natural language processing (NLP) tasks, such as machine translation (Vaswani et al. 2017), semantic role labeling (Strubell et al. 2018), and language representations (Devlin et al. 2019). The strength of DNNs lies in their ability to capture different linguistic properties of the input by different layers (Shi, Padhi, and Knight 2016; Raganato and Tiedemann 2018), and composing (i.e. aggregating) these layer representations can further improve performances by providing more comprehensive linguistic information of the input (Peters et al. 2018; Dou et al. 2018).

Recent NLP studies show that single neurons in neural models which are defined as individual dimensions of the representation vectors, carry distinct linguistic information (Bau et al. 2019). A follow-up work further reveals that simple properties such as coordinating conjunction (e.g., "but/and") or determiner (e.g., "the") can be attributed to individual neurons, while complex linguistic phenomena such as syntax (e.g., part-of-speech tag) and semantics (e.g., semantic entity type) are distributed across neurons (Dalvi et al. 2019). These observations are consistent with recent findings in neuroscience, which show that task-relevant information can be decoded from a group of neurons interacting with each other (Morcos and Harvey 2016). One question naturally arises: *can we better capture complex linguistic phenomena by composing/grouping the linguistic properties embedded in individual neurons?*

The starting point of our approach is an observation in neuroscience: *stronger neuron interactions* – directly exchanging signals between neurons, enable more information processing in the nervous system (Koch, Poggio, and Torre 1983). We believe that simulating the neuron interactions in nervous system would be an appealing alternative to representation composition, which can potentially better learn the compositionality of natural language with subtle operations at a smaller granularity. Concretely, we employ bilinear pooling (Lin, RoyChowdhury, and Maji 2015), which executes pairwise multiplicative interactions among individual representation elements, to achieve *strong* neuron interactions. We also introduce a low-rank approximation to make the original bilinear models computationally feasible (Kim et al. 2017). Furthermore, as bilinear pooling only encodes multiplicative second-order features, we propose *extended bilinear pooling* to incorporate first-order representations, which can capture more comprehensive information of the input sentences.

We validate the proposed neuron interaction based (NI-based) representation composition on top of multi-layer multi-head self-attention networks (MLMHSANS). The reason is two-fold. First, MLMHSANS are critical components of various SOTA DNNS models, such as TRANSFORMER (Vaswani et al. 2017), BERT (Devlin et al. 2019), and LISA (Strubell et al. 2018). Second, MLMHSANS involve in compositions of both multi-layer representations and multi-head representations, which can investigate the universality of NI-based composition. Specifically,

- First, we conduct experiments on the machine translation

[*] Corresponding author: Zhaopeng Tu. Work was partially done when Jian Li and Baosong Yang were interning at Tencent AI Lab.

task, a benchmark to evaluate the performance of neural models. Experimental results on the widely-used WMT14 English⇒German and English⇒French data show that the NI-based composition consistently improves performance over TRANSFORMER across language pairs. Compared with existing representation composition strategies (Peters et al. 2018; Dou et al. 2018), our approach shows its superiority in efficacy and efficiency.

- Second, we carry out linguistic analysis (Conneau et al. 2018) on the learned representations from NMT encoder, and find that NI-based composition indeed captures more syntactic and semantic information as expected. These results provide support for our hypothesis that modeling strong neuron interactions helps to better capture complex linguistic information via advanced composition functions, which is essential for downstream NLP tasks.

This paper is an early step in exploring neuron interactions for representation composition in NLP tasks, which we hope will be a long and fruitful journey. We make the following contributions:

- Our study demonstrates the necessity of modeling neuron interactions for representation composition in deep NLP tasks. We employ bilinear pooling to simulate the strong neuron interactions.
- We propose *extended bilinear pooling* to incorporate first-order representations, which produces a more comprehensive representation.
- Experimental results show that representation composition benefits the widely-employed MLMHSANs by aggregating information learned by multi-layer and/or multi-head attention components.

Background

Multi-Layer Multi-Head Self-Attention

In the past two years, MLMHSANs based models establish the SOTA performances across different NLP tasks. The main strength of MLMHSANs lies in the powerful representation learning capacity provided by the multi-layer and multi-head architectures. MLMHSANs perform a series of nonlinear transformations from the input sequences to final output sequences.

Specifically, MLMHSANs are composed of a stack of L identical layers (*multi-layer*), each of which is calculated as

$$\mathbf{H}^l \quad = \quad \textsc{Self-Att}(\mathbf{H}^{l-1}) + \mathbf{H}^{l-1}, \tag{1}$$

where a residual connection is employed around each of two layers (He et al. 2016). $\textsc{Self-Att}(\cdot)$ is a self-attention model, which captures dependencies among hidden states in $\mathbf{H}^{l-1}$:

$$\textsc{Self-Att}(\mathbf{H}^{l-1}) = \textsc{Att}(\mathbf{Q}^l, \mathbf{K}^{l-1})\ \mathbf{V}^{l-1}, \tag{2}$$

where $\{\mathbf{Q}^l, \mathbf{K}^{l-1}, \mathbf{V}^{l-1}\}$ are the query, key and value vectors that are transformed from the lower layer $\mathbf{H}^{l-1}$, respectively.

Instead of performing a single attention function, Vaswani et al. (2017) found it is beneficial to capture different context features with multiple individual attention functions (*multi-head*). Concretely, multi-head attention model first transforms $\{\mathbf{Q}, \mathbf{K}, \mathbf{V}\}$ into H subspaces with different, learnable linear projections:[1]

$$\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h = \mathbf{Q}\mathbf{W}_h^Q, \mathbf{K}\mathbf{W}_h^K, \mathbf{V}\mathbf{W}_h^V, \tag{3}$$

where $\{\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h\}$ are respectively the query, key, and value representations of the h-th head. $\{\mathbf{W}_h^Q, \mathbf{W}_h^K, \mathbf{W}_h^V\}$ denote parameter matrices associated with the h-th head. H self-attention functions (Equation 2) are applied in parallel to produce the output states $\{\mathbf{O}_1, \dots, \mathbf{O}_H\}$. Finally, the H outputs are concatenated and linearly transformed to produce a final representation:

$$\mathbf{H} = [\mathbf{O}_1, \dots, \mathbf{O}_H]\ \mathbf{W}^O, \tag{4}$$

where $\mathbf{W}^O \in \mathbb{R}^{d \times d}$ is a trainable matrix.

Representation Composition

Composing (i.e. aggregating) representations learned by different layers or attention heads has been shown beneficial for MLMHSANs (Dou et al. 2018; Ahmed, Keskar, and Socher 2018). Without loss of generality, from here on, we refer to $\{\mathbf{r}_1, \dots, \mathbf{r}_N\} \in \mathbb{R}^d$ for the representations to compose, where $\mathbf{r}_i$ can be a layer representation ($\mathbf{H}^l$, Equation 1) or head representation ($\mathbf{O}_h$, Equation 4). The composition is expressed as

$$\widetilde{\mathbf{H}} = \textsc{Compose}(\mathbf{r}_1, \dots, \mathbf{r}_N), \tag{5}$$

where $\textsc{Compose}(\cdot)$ can be arbitrary functions, such as linear combination[2] (Peters et al. 2018; Ahmed, Keskar, and Socher 2018) and hierarchical aggregation (Dou et al. 2018). Although effective to some extent, these approaches do not model neuron interactions among the representation vectors, which we believe is valuable for representation composition in deep NLP models.

Approach

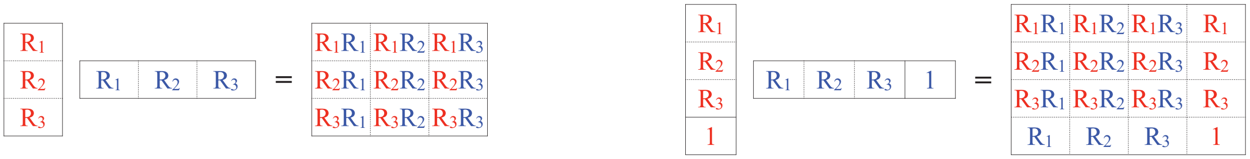
Motivation

Different types of neurons in the nervous system carry distinct signals (Cohen et al. 2012). Similarly, neurons in deep NLP models – individual dimensions of representation vectors, carry distinct linguistic information (Bau et al. 2019; Dalvi et al. 2019). Studies in neuroscience reveal that stronger neuron interactions bring more information processing capability (Koch, Poggio, and Torre 1983), which we believe also applies to deep NLP models.

In this work, we explore the strong neuron interactions provided by bilinear pooling for representation composition. Bilinear pooling (Lin, RoyChowdhury, and Maji 2015) is a recently proposed feature fusion approach in the vision field. Instead of linearly combining all representations, bilinear pooling executes pairwise multiplicative interactions among

[1] Here we skip the layer index for simplification.

[2] The linear composition of multi-head representations (Equation 4) can be rewritten in the format of weighted sum: $\mathbf{O} = \sum_{h=1}^{H} \mathbf{O}_h \mathbf{W}_h^O$ with $\mathbf{W}_h^O \in \mathbb{R}^{\frac{d}{H} \times d}$.



(a) Bilinear Pooling

(b) Extended Bilinear Pooling

Figure 1: Illustration of (a) *bilinear pooling* that models fully neuron-wise multiplicative interaction, and (b) *extended bilinear pooling* that captures both second- and first-order neuron interactions.

individual representations, to model *full* neuron interactions as shown in Figure 1(a).

Note that there are many possible ways to implement the neuron interactions. The aim of this paper is not to explore this whole space but simply to show that one fairly straightforward implementation works well on a strong benchmark.

Bilinear Pooling for Neuron Interaction

Bilinear Pooling Bilinear pooling (Tenenbaum and Freeman 2000) is defined as an *outer product* of two representation vectors followed by a linear projection. As illustrated in Figure 1(a), all elements of the two vectors have direct multiplicative interactions with each other. However, in the scenario of multi-layer and multi-head composition, we generally have more than two representation vectors to compose (i.e., L layers and H attention heads). To utilize the full second-order (i.e. multiplicative) interactions in bilinear pooling, we concatenate all the representation vectors and feed the concatenated vector twice to the bilinear pooling. Concretely, we have:

$$\mathbf{R} = |\widehat{\mathbf{R}}\widehat{\mathbf{R}}^\top|\mathbf{W}^B, \tag{6}$$

$$\widehat{\mathbf{R}} = [\mathbf{r}_1, \dots, \mathbf{r}_N], \tag{7}$$

where $|\widehat{\mathbf{R}}\widehat{\mathbf{R}}^\top| \in \mathbb{R}^{Nd \times Nd}$ is the outer product of the concatenated representation $\widehat{\mathbf{R}}$, $|\cdot|$ denotes serializing the matrix into a vector with dimensionality $(Nd)^2$. In this way, all elements in the partial representations are able to interact with each other in a multiplicative way.

However, the parameter matrix $\mathbf{W}^B \in \mathbb{R}^{(Nd)^2 \times d}$ and computing cost cubically increases with dimensionality d, which becomes problematic when training or decoding on a GPU with limited memory[3]. There have been a few attempts to reduce the computational complexity of the original bilinear pooling. Gao et al. (2016) propose *compact bilinear pooling* to reduce the quadratic expansion of dimensionality for image classification. Kim et al. (2017) and Kong and Fowlkes (2017) propose *low-rank bilinear pooling* for visual question answering and image classification respectively, which further reduces the parameters to be learned and achieves comparable effectiveness with full bilinear pooling. In this work, we focus on the low-rank approximation for its efficiency, and generalize from the original model for deep representations.

Low-Rank Approximation In the full bilinear models, each output element $R_i \in \mathbb{R}^1$ can be expressed as

$$\begin{aligned} R_i &= \sum_{j=1}^{Nd}\sum_{k=1}^{Nd} w^B_{jk,i}\widehat{R}_j\widehat{R}_k^\top \\ &= \widehat{\mathbf{R}}^\top\mathbf{W}_i^B\widehat{\mathbf{R}}, \end{aligned} \tag{8}$$

where $\mathbf{W}_i^B \in \mathbb{R}^{Nd \times Nd}$ is a weight matrix to produce output element R_i. The low-rank approximation enforces the rank of $\mathbf{W}_i^B$ to be low-rank $r \leq Nd$ (Pirsiavash, Ramanan, and Fowlkes 2009), which is then factorized as $\mathbf{U}_i\mathbf{V}_i^\top$ with $\mathbf{U}_i \in \mathbb{R}^{Nd \times r}$ and $\mathbf{V}_i \in \mathbb{R}^{Nd \times r}$. Accordingly, Equation 8 can be rewritten as

$$\begin{aligned} R_i &= \widehat{\mathbf{R}}^\top\mathbf{U}_i\mathbf{V}_i^\top\widehat{\mathbf{R}} \\ &= (\widehat{\mathbf{R}}^\top\mathbf{U}_i \odot \widehat{\mathbf{R}}^\top\mathbf{V}_i)\mathbb{1}_r, \end{aligned} \tag{9}$$

where $\mathbb{1}_r$ is a r-dimensional vector of ones, $\odot$ represents element-wise product. By replacing $\mathbb{1}_r$ with $\mathbf{P} \in \mathbb{R}^{r \times d}$, and redefining $\mathbf{U} \in \mathbb{R}^{Nd \times r}$ and $\mathbf{V} \in \mathbb{R}^{Nd \times r}$, the low-rank approximation can be defined as

$$\mathbf{R} = (\widehat{\mathbf{R}}^\top\mathbf{U} \odot \widehat{\mathbf{R}}^\top\mathbf{V})\mathbf{P}. \tag{10}$$

In this way, the computation complexity is reduced from $O(d^3)$ to $O(d^2)$. And the parameter matrices $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{P}$ are now feasible to fit in GPU memory.

Extended Bilinear Pooling with First-Order Representation Previous work in information theory has proven that second-order and first-order representations encode different types of information (Goudreau et al. 1994), which we believe also holds on NLP tasks. As bilinear pooling only encodes second-order (i.e., multiplicative) interactions among individual neurons, we propose the *extended bilinear pooling* to inherit the advantages of first-order representations and form a more comprehensive representation.

Specifically, we append $\mathbf{1}$s to the representation vectors. As illustrated in Figure 1(b), we respectively append $\mathbf{1}$ to the two $\mathbf{R}$ vectors, then the outer product of them produces both second-order and first-order interactions among the elements. According to Equation 10, the final representation is revised as:

$$\mathbf{R_f} = (\begin{bmatrix}\widehat{\mathbf{R}} \\ 1\end{bmatrix}^\top \mathbf{U} \odot \begin{bmatrix}\widehat{\mathbf{R}} \\ 1\end{bmatrix}^\top \mathbf{V})\,\mathbf{P}, \tag{11}$$

where $\widehat{\mathbf{R}}$ is the concatenated representation as in Equation 7. As a result, the final representation $\mathbf{R_f}$ preserves both multiplicative bilinear features (as in Equation 10) and first-order linear features (as in Equation 4).

[3] For example, a regular TRANSFORMER model requires a huge amount of 36 billion ($(Nd)^2 \times d$) parameters for $d = 1000$ and $N = 6$.

#	Model	# Para.	Train	Decode	BLEU
1	TRANSFORMER-BASE	88.0M	2.02	1.50	27.31
	Existing representation composition				
2	+ Multi-Layer: Linear Combination	+3.1M	1.98	1.46	27.77
3	+ Multi-Layer: Hierarchical Aggregation	+23.1M	1.62	1.36	28.32[4]
4	+ Multi-Head: Hierarchical Aggregation	+13.6M	1.74	1.38	28.13
5	+ Both (3+4)	+36.7M	1.42	1.25	28.42
	This work: neuron-interaction based representation composition				
6	+ Multi-Layer: *NI-based Composition*	+16.8M	1.93	1.44	28.31
7	+ Multi-Head: *NI-based Composition*	+14.1M	1.92	1.43	28.29
8	+ Both (6+7)	+30.9M	1.87	1.40	**28.54**

Table 1: Translation performance on WMT14 English⇒German translation task. "# Para." denotes the number of parameters, and "Train" and "Decode" respectively denote the training speed (steps/second) and decoding speed (sentences/second). We compare our model with linear combination (Peters et al. 2018) and hierarchical aggregation (Dou et al. 2018).

Applying to TRANSFORMER TRANSFORMER (Vaswani et al. 2017) consists of an encoder and a decoder, each of which is stacked in 6 layers where we can apply multi-layer composition (excluding the embedding layer) to produce the final representations of the encoder and decoder. Besides, each layer has one (in encoder) or two (in decoder) multi-head attention component with H heads, to which we can apply multi-head composition to substitute Equation 4. The two sorts of representation composition can be used individually, while combining them is expected to further improve the performance.

Experiments

Setup

Dataset We conduct experiments on the WMT2014 English⇒German (En⇒De) and English⇒French (En⇒Fr) translation tasks. The En⇒De dataset consists of about 4.56 million sentence pairs. We use newstest2013 as the development set and newstest2014 as the test set. The En⇒Fr dataset consists of 35.52 million sentence pairs. We use the concatenation of newstest2012 and newstest2013 as the development set and newstest2014 as the test set. We employ BPE (Sennrich, Haddow, and Birch 2016) with 32K merge operations for both language pairs. We adopt the case-sensitive 4-gram NIST BLEU score (Papineni et al. 2002) as our evaluation metric and bootstrap resampling (Koehn 2004) for statistical significance test.

Models We evaluate the proposed approaches on the advanced TRANSFORMER model (Vaswani et al. 2017), and implement on top of an open-source toolkit – THUMT (Zhang et al. 2017). We follow Vaswani et al. (2017) to set the configurations and have reproduced their reported results on the En⇒De task. The parameters of the proposed models are initialized by the pre-trained TRANSFORMER model. We have tested both *Base* and *Big* models, which differ at hidden size (512 vs. 1024) and number of attention heads (8 vs. 16). Concerning the low-rank parameter (Equation 9), we set low-rank dimensionality r to 512 and 1024 in *Base* and *Big* models respectively. All models are trained on eight NVIDIA P40 GPUs where each is allocated with a batch size of 4096 tokens. In consideration of computation cost, we study model variations with *Base* model on the En⇒De task, and evaluate overall performance with *Big* model on both En⇒De and En⇒Fr tasks.

Comparison to Existing Approaches

In this section, we evaluate the impacts of different representation composition strategies on the En⇒De translation task with TRANSFORMER-BASE, as listed in Table 1.

Existing Representation Composition (Rows 1-5) For the conventional TRANSFORMER model, it adopts multi-head composition with linear combination but only uses top-layer representation as its default setting. Accordingly, we keep the linear multi-head composition (Row 1) unchanged, and choose two representative multi-layer composition strategies (Rows 2 and 3): the widely-used linear combination (Peters et al. 2018) and the effective hierarchical aggregation (Dou et al. 2018). The hierarchical aggregation merges states of different layers through a CNN-like tree structure with the filter size being two, to hierarchically preserve and combine feature channels.

As seen, linearly combining all layers (Row 2) achieves +0.46 BLEU improvement over TRANSFORMER-BASE with almost the same training and decoding speeds. Hierarchical aggregation for multi-layer composition (Row 3) yields larger improvement in terms of BLEU score, but at the cost of considerable speed decrease. To make a fair comparison, we also implement hierarchical aggregation for multi-head composition (Rows 4 and 5), which consistently improves performances at the cost of introducing more parameters and slower speeds.

The Proposed Approach (Rows 6-8) Firstly, we apply our NI-based composition, i.e. *extended bilinear pooling*, for multi-layer composition with the default linear multi-head

[4] The original result in (Dou et al. 2018) is 28.63, which is *case-insensitive*. As we report case-sensitive BLEU scores, we have requested Dou et al. to get this result.

Architecture	EN⇒DE			EN⇒FR		
	# Para.	Train	BLEU	# Para.	Train	BLEU
Existing NMT systems: (Vaswani et al. 2017)						
Transformer-Base	65M	n/a	27.3	n/a	n/a	38.1
Transformer-Big	213M	n/a	28.4	n/a	n/a	41.8
Our NMT systems						
Transformer-Base	88M	2.02	27.31	95M	2.01	39.28
+ NI-Based Composition	118M	1.87	28.54$^{\Uparrow}$	125M	1.85	40.15$^{\Uparrow}$
Transformer-Big	264M	0.85	28.58	278M	0.84	41.41
+ NI-Based Composition	387M	0.61	29.17$^{\Uparrow}$	401M	0.59	42.10$^{\Uparrow}$

Table 2: Comparing with existing NMT systems on WMT14 English⇒German ("EN⇒DE") and English⇒French ("EN⇒FR") translation tasks. "⇑": significantly better than the baseline ($p < 0.01$) using bootstrap resampling (Koehn 2004).

composition (Row 6). We find that the approach achieves almost the same translation performance as hierarchical aggregation (Row 3), while keeps the training and decoding speeds as *efficient* as linear combination. Then, we apply the NI-based approach for multi-head composition with the default top layer exploitation (Row 7). We can see that our approach gains +0.98 BLEU point over Transformer-Base and achieves more improvement than hierarchical aggregation (Row 4). The two results demonstrate that our NI-based approach can be effectively applied to different representation composition scenarios.

At last, we simultaneously apply the NI-based approach to the multi-layer and multi-head composition (Row 8). Our model achieves further improvement over individual models and the hierarchical aggregation (Row 5), showing that Transformer can benefit from the complementary composition from multiple heads and historical layers. In the following experiments, we adopt NI-based composition for both the multi-layer and multi-head compositions as the default strategy.

Main Results on Machine Translation

In this section, we validate the proposed NI-based representation composition on both WMT14 En⇒De and En⇒Fr translation tasks. Experimental results are listed in Table 2. The performances of our implemented Transformer match the results on both language pairs reported in previous work (Vaswani et al. 2017), which we believe makes the evaluation convincing.

Incorporating NI-based composition consistently and significantly improves translation performance for both base and big Transformer models across language pairs, demonstrating the effectiveness and universality of the proposed NI-based representation composition. It is encouraging to see that Transformer-Base with NI-based composition even achieves competitive performance as that of Transformer-Big in the En⇒De task, with only half fewer parameters and the training speed is twice faster. This further demonstrates that our performance gains are not simply brought by additional parameters. Note that the improvement on En⇒De task is larger than En⇒Fr task, which can be attributed to the size of training data (4M vs. 35M).

	Task	Base	Ours	△
Surface	SeLen	92.20	92.11	-0.1%
	WC	63.00	63.50	+0.8%
	Ave.	77.60	77.81	+0.3%
Syntactic	TrDep	44.74	44.96	+0.5%
	ToCo	79.02	81.31	**+2.9%**
	BShif	71.24	72.44	**+1.7%**
	Ave.	65.00	66.24	**+1.9%**
Semantic	Tense	89.24	89.26	+0.0%
	SubNm	84.69	87.05	**+2.8%**
	ObjNm	84.53	86.91	**+2.8%**
	SOMO	52.13	52.52	+0.7%
	CoIn	62.47	64.93	**+3.9%**
	Ave.	74.61	76.13	**+2.0%**

Table 3: Classification accuracies on 10 probing tasks of evaluating the linguistic properties ("Surface", "Syntactic", and "Semantic"). "Ave." denotes the averaged accuracy in each category. "△" denotes the relative improvement, and we highlight the numbers $\geq 1\%$.

Analysis

In this section, we conduct extensive analysis to deeply understand the proposed models in terms of 1) investigating the linguistic properties learned by the NMT encoder; 2) the influences of first-order representation and low-rank constraint; and 3) the translation performances on sentences of varying lengths.

Targeted Linguistic Evaluation on NMT Encoder Machine translation is a complex task, which consists of both the understanding of input sentence (encoder) and the generation of output conditioned on such understanding (decoder). In this probing experiment, we evaluate the understanding part using Transformer encoders that are trained on the EN⇒DE NMT data, and are fixed in the probing tasks with only MLP classifiers being trained on probing data.

Recently, Conneau et al. (2018) designed 10 probing tasks to study what linguistic properties are captured by representations from sentence encoders. A probing task is a classification problem that focuses on simple linguistic properties of input sentences, including surface information, syntactic information, and semantic information. For example, "WC" tests whether it is possible to recover information about the original words given its sentence embedding. "Bshif" checks whether two consecutive tokens have been inverted. "SubNm" focuses on the number of the subject of the main clause. For more detailed description about the 10 tasks, interested readers can refer to the original paper (Conneau et al. 2018). We conduct probing tasks to examine whether the NI-based representation composition can benefit the TRANSFORMER encoder to produce more informative representation.

Table 3 lists the results. The NI-based composition outperforms that by the baseline in most probing tasks, proving that our composition strategy indeed helps TRANSFORMER encoder generate more informative representation, especially at the syntactic and semantic level. The averaged gains in syntactic and semantic tasks are significant, showing that our strategy makes SAN capture more high-level linguistic properties. Note that the lower values in surface tasks (e.g., SeLen), are consistent with the conclusion in (Conneau et al. 2018): as model captures deeper linguistic properties, it will tend to forget about these superficial features.

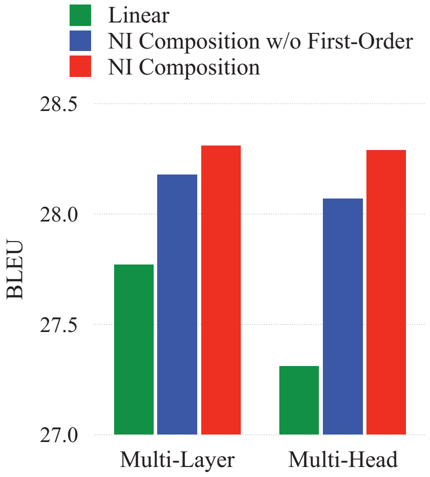


Figure 2: Effect of first-order representation on WMT14 En⇒De translation task.

Effect of First-Order Representation As aforementioned, we extend the conventional bilinear pooling by appending 1s to the representation vectors thus incorporate first-order representations (i.e. linear combination), and capture both multiplicative bilinear features and additive linear features. Here we conduct ablation study to validate the effectiveness of each component. We respectively experiment on multi-layer and multi-head representation composition, and the results are shown in Figure 2.

Several observations can be made. First, we notice that by replacing linear combination with mere bilinear pooling ("NI-based composition w/o first-order" in Figure 2), the translation performance significantly improves both in multi-layer and multi-head composition, demonstrating the effectiveness of full neuron interaction and second-order features. We further observe that it is indeed beneficial to extend bilinear pooling with linear combination ("NI composition" in Figure 2) which captures the complementary information among them and forms a more comprehensive representation of the input.

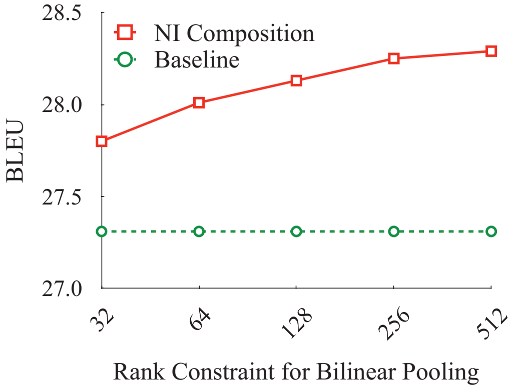


Figure 3: BLEU scores on the En⇒De test set with different rank constraints for bilinear pooling. "Baseline" denotes TRANSFORMER-BASE.

Effect of Low-Rank Constraint In this experiment, we study the impact of low-rank constraint r (Equation 9) on bilinear pooling, as shown in Figure 3. It is interesting to investigate whether the model with a smaller setting of r can also achieve considerable results. We examine groups of multi-head composition models with different r on the En⇒De translation task. From Figure 3, we can see that the translation performance increases with larger r value and the model with $r = 512$ achieves best performance[5]. Note that even when the dimensionality r is reduced to 32, our model can still consistently outperform the baseline with only 0.9M parameters added (not shown in the figure). This reconfirms our claim that the improvements on the BLEU score could not be simply attributed to the additional parameters.

Length Analysis We group sentences of similar lengths together and compute the BLEU score for each group, as shown in Figure 4. Generally, the performance of TRANSFORMER goes up with the increase of input sentence lengths, which is different from the results on single-layer RNNSearch models (i.e., performance decreases on longer sentences) as shown in (Tu et al. 2016). We attribute this phenomenon to the advanced TRANSFORMER architecture including multiple layers, multi-head attention and feed-forward networks.

Clearly, our NI-based approaches outperform the baseline TRANSFORMER in all length segments, including only using multi-layer composition or multi-head composition, which

[5] The maximum value of r is 512 since the rank of a matrix $\mathbf{W} \in \mathbb{R}^{Nd \times Nd}$ is bounded by Nd.

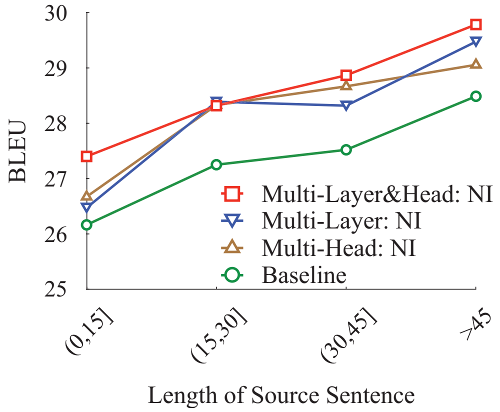


Figure 4: BLEU scores on the En⇒De test set with respect to various input sentence lengths. "Baseline" denotes TRANSFORMER-BASE.

verifies our contribution that representation composition indeed benefits SANs. Moreover, multi-layer composition and multi-head composition are complementary to each other regarding different length segments, and simultaneously applying them achieves further performance gain.

Related Work

Bilinear Pooling Bilinear pooling has been well-studied in the computer vision community, which is first introduced by Tenenbaum and Freeman (2000) to separate style and content. Bilinear pooling has since then been considered to replace fully-connected layers in neural networks by introducing second-order statistics, and applied to fine grained recognition (Lin, RoyChowdhury, and Maji 2015). While bilinear models provide richer representations than linear models (Goudreau et al. 1994), bilinear pooling produces a high-dimensional feature of quadratic expansion, which may constrain model structures and computational resources. To address this challenge, Gao et al. (2016) propose compact bilinear pooling through random projections for image classification, which is further applied to visual question answering (Fukui et al. 2016). Kim et al. (2017) and Kong and Fowlkes (2017) independently propose low-rank approximation on the transformation matrix of bilinear pooling, which aims to reduce the model size and corresponding computational burden. Their models are applied to visual question answering and fine-grained image classification, respectively.

While most work focus on computer vision tasks, our work is among the few studies (Dozat and Manning 2017; Delbrouck and Dupont 2017), which prove the idea of bilinear pooling can have promising applications on NLP tasks. Our approach differs at: 1) we apply bilinear pooling to representation composition in NMT, while they apply to the attention model in either parsing or multimodal NMT; and 2) we extend the original bilinear pooling to incorporate first-order representations, which consistently improves translation performance in different scenarios (Figure 2).

Multi-Layer Representation Composition Exploiting multi-layer representations has been well studied in the NLP community. Peters et al. (2018) have found that linearly combining different layers is helpful and improves their performances on various NLP tasks. In the context of NMT, several neural network based approaches to fuse information across historical layers have been proposed, such as dense information flow (Shen et al. 2018), iterative and hierarchical aggregation (Dou et al. 2018), routing-by-agreement (Dou et al. 2019), and transparent attention (Bapna et al. 2018).

In this work, we consider representation composition from a novel perspective of *modeling neuron interactions*, which we prove is a promising and effective direction. Besides, we generalize layer aggregation to representation composition in SANs by also considering multi-head composition, and we propose an unified NI-based approach to aggregate both types of representation.

Multi-Head Self-Attention Multi-head attention has shown promising results in many NLP tasks, such as machine translation (Vaswani et al. 2017) and semantic role labeling (Strubell et al. 2018). The strength of multi-head attention lies in the rich expressiveness by using multiple attention functions in different representation subspaces. Previous work show that multi-head attention can be further enhanced by encouraging individual attention heads to extract distinct information. For example, Li et al. (2018) propose disagreement regularizations to encourage different attention heads to encode distinct features, and Strubell et al. (2018) employ different attention heads to capture different linguistic features. Li et al. (2019) is a pioneering work on empirically validating the importance of information aggregation for multi-head attention. Along the same direction, we apply the NI-based approach to compose the representations learned by different attention heads (as well as different layers), and empirically reconfirm their findings.

Conclusion

In this work, we propose NI-based representation composition for MLMHSANs, by modeling strong neuron interactions in the representation vectors generated by different layers and attention heads. Specifically, we employ bilinear pooling to capture pairwise multiplicative interactions among individual neurons, and propose *extended bilinear pooling* to further incorporate first-order representations. Experiments on machine translation tasks show that our approach effectively and efficiently improves translation performance over the TRANSFORMER model, and multi-head composition and multi-layer composition are complementary to each other. Further analyses reveal that our model makes the encoder of TRANSFORMER capture more syntactic and semantic properties of input sentences.

Future work includes exploring more neuron interaction based approaches for representation composition other than the bilinear pooling, and applying our model to a variety of network architectures such as BERT (Devlin et al. 2019) and LISA (Strubell et al. 2018).

Acknowledgement

The work described in this paper was partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (No. CUHK 14210717 of the General Research Fund), and Microsoft Research Asia (2018 Microsoft Research Asia Collaborative Research Award). We thank the anonymous reviewers for their comments and suggestions.

References

Ahmed, K.; Keskar, N. S.; and Socher, R. 2018. Weighted Transformer Network for Machine Translation. *arXiv*.

Bapna, A.; Chen, M.; Firat, O.; Cao, Y.; and Wu, Y. 2018. Training deeper neural machine translation models with transparent attention. In *EMNLP*.

Bau, A.; Belinkov, Y.; Sajjad, H.; Durrani, N.; Dalvi, F.; and Glass, J. 2019. Identifying and controlling important neurons in neural machine translation. In *ICLR*.

Cohen, J. Y.; Haesler, S.; Vong, L.; Lowell, B. B.; and Uchida, N. 2012. Neuron-type-specific signals for reward and punishment in the ventral tegmental area. *Nature* 482(7383):85.

Conneau, A.; Kruszewski, G.; Lample, G.; Barrault, L.; and Baroni, M. 2018. What You Can Cram into A Single $&!#∗ Vector: Probing Sentence Embeddings for Linguistic Properties. In *ACL*.

Dalvi, F.; Durrani, N.; Sajjad, H.; Belinkov, Y.; Bau, A.; and Glass, J. 2019. What is one grain of sand in the desert? analyzing individual neurons in deep nlp models. In *AAAI*.

Delbrouck, J.-B., and Dupont, S. 2017. Multimodal compact bilinear pooling for multimodal neural machine translation. *arXiv*.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*.

Dou, Z.; Tu, Z.; Wang, X.; Shi, S.; and Zhang, T. 2018. Exploiting deep representations for neural machine translation. In *EMNLP*.

Dou, Z.; Tu, Z.; Wang, X.; Wang, L.; Shi, S.; and Zhang, T. 2019. Dynamic layer aggregation for neural machine translation with routing-by-agreement. In *AAAI*.

Dozat, T., and Manning, C. D. 2017. Deep biaffine attention for neural dependency parsing. In *ICLR*.

Fukui, A.; Park, D. H.; Yang, D.; Rohrbach, A.; Darrell, T.; and Rohrbach, M. 2016. Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding. In *EMNLP*.

Gao, Y.; Beijbom, O.; Zhang, N.; and Darrell, T. 2016. Compact bilinear pooling. In *CVPR*.

Goudreau, M. W.; Giles, C. L.; Chakradhar, S. T.; and Chen, D. 1994. First-order versus second-order single-layer recurrent neural networks. *IEEE Transactions on Neural Networks* 5(3):511–513.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Kim, J.-H.; On, K.-W.; Lim, W.; Kim, J.; Ha, J.-W.; and Zhang, B.-T. 2017. Hadamard product for low-rank bilinear pooling. In *ICLR*.

Koch, C.; Poggio, T.; and Torre, V. 1983. Nonlinear interactions in a dendritic tree: localization, timing, and role in information processing. *Proceedings of the National Academy of Sciences* 80(9):2799–2802.

Koehn, P. 2004. Statistical Significance Tests for Machine Translation Evaluation. In *EMNLP*.

Kong, S., and Fowlkes, C. 2017. Low-rank bilinear pooling for fine-grained classification. In *CVPR*.

Li, J.; Tu, Z.; Yang, B.; Lyu, M. R.; and Zhang, T. 2018. Multi-Head Attention with Disagreement Regularization. In *EMNLP*.

Li, J.; Yang, B.; Dou, Z.-Y.; Wang, X.; Lyu, M. R.; and Tu, Z. 2019. Information aggregation for multi-head attention with routing-by-agreement. In *NAACL*.

Lin, T.-Y.; RoyChowdhury, A.; and Maji, S. 2015. Bilinear cnn models for fine-grained visual recognition. In *ICCV*.

Morcos, A. S., and Harvey, C. D. 2016. History-dependent variability in population dynamics during evidence accumulation in cortex. *Nature neuroscience* 19(12):1672.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. BLEU: A method for Automatic Evaluation of Machine Translation. In *ACL*.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep Contextualized Word Representations. In *NAACL*.

Pirsiavash, H.; Ramanan, D.; and Fowlkes, C. C. 2009. Bilinear classifiers for visual recognition. In *NIPS*.

Raganato, A., and Tiedemann, J. 2018. An Analysis of Encoder Representations in Transformer-Based Machine Translation. In *EMNLP BlackboxNLP Workshop*.

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Neural Machine Translation of Rare Words with Subword Units. *ACL*.

Shen, Y.; He, D.; Qin, T.; and Liu, T.-Y. 2018. Dense information flow for neural machine translation. *NAACL*.

Shi, X.; Padhi, I.; and Knight, K. 2016. Does string-based neural mt learn source syntax? In *EMNLP*.

Strubell, E.; Verga, P.; Andor, D.; Weiss, D.; and McCallum, A. 2018. Linguistically-Informed Self-Attention for Semantic Role Labeling. In *EMNLP*.

Tenenbaum, J. B., and Freeman, W. T. 2000. Separating style and content with bilinear models. *Neural Computation* 12(6):1247–1283.

Tu, Z.; Lu, Z.; Liu, Y.; Liu, X.; and Li, H. 2016. Modeling coverage for neural machine translation. In *ACL 2016*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention Is All You Need. In *NIPS*.

Zhang, J.; Ding, Y.; Shen, S.; Cheng, Y.; Sun, M.; Luan, H.; and Liu, Y. 2017. THUMT: An Open Source Toolkit for Neural Machine Translation. *arXiv*.