Unified Transformer for Sequence-to-Sequence Learning

Guangxiang Zhao, Jingjing Xu, Zhiyuan Zhang, Xuancheng Ren, Liangcheng Luo, Xu Sun

{*zhaoguangxiang, jingjingxu, zzy1210, renxc, luolc, xusun* }@*pku.edu.cn*

MOE Key Lab of Computational Linguistics, School of EECS, Peking University

Peking University, No.5 Yiheyuan Road, Haidian District, Beijing, P.R.China 100871

Abstract

As a sequence-to-sequence learning model, Transformer builds the network via stacking self-attention, cross-attention, and feedforward neural network (FFN) modules. Although it achieves record-breaking results on machine translation, we have two observations on Transformer: first, Transformer stacks complex modules of different nature sequentially, which does not comply with the cell assembly theory in neuroscience; second, Transformer relies on solely the attention mechanism to aggregate contextual representations globally, which is unfavorable for transforming long sequences. We propose a simple network architecture, the Unified Transformer, which simplifies the Transformer structure to only one basic building unit. Each basic building unit contains multiple complementary neural systems, including the FFN and the two attention blocks, whose neurons gather and activate together, forming a cell assembly. In addition to the FFN and the attention blocks, we also unify the convolution mechanism to capture features with richer granularity to better transforming long sequences. In experiments, we show that on large-scale machine translation tasks, the Unified Transformer, built by simply stacking the proposed unified unit, is superior in translation quality over strong non-unified Transformer, especially on transforming long sequences. We also show that it generalizes well to models and datasets of different sizes. Moreover, the unified unit has the potential for accelerating inference due to its parallelism.

Keywords: Transformer; Attention Mechanism; Convolution Mechanism; Cell Assembly Theory; Sequence-to-sequence Learning; Natural Language Processing.

1. Introduction

Sequence-to-sequence learning aims to learn a model that converts a source input sequence to a target output sequence, where each sequence comprises several tokens. Its typical application is the machine translation task. In machine translation, we need to translate a sequence of words into a sequence with the same meaning in another language. Transformer, which is a stack of self-attention blocks, cross-attention blocks, and feedforward neural networks (FFN), achieves record-breaking results on machine translation tasks [1, 2].

We show the Transformer in subfigure (a) of Figure 1 and discuss two shortcomings of the Transformer rooted in how it is structured.

Motivation 1: Issues of the sequential mode in signal collecting and transforming. Transformer has a complex form and does not comply with the cell assembly theory in neuroscience. Transformer uses multiple different neural systems to process signals sequentially in both the Encoder and Decoder. For example, in the Decoder, information has to be sequentially processed in three different neural networks. However, recent researches point out that in this sequential mode, the order of these modules should be carefully determined, and the optimal orders could be wildly different according to datasets [3]. Moreover, this sequential mode is not beneficial for parallel computation. From the perspective of neuroscience, the cell assembly theory indicates that two neural systems that are active simultaneously

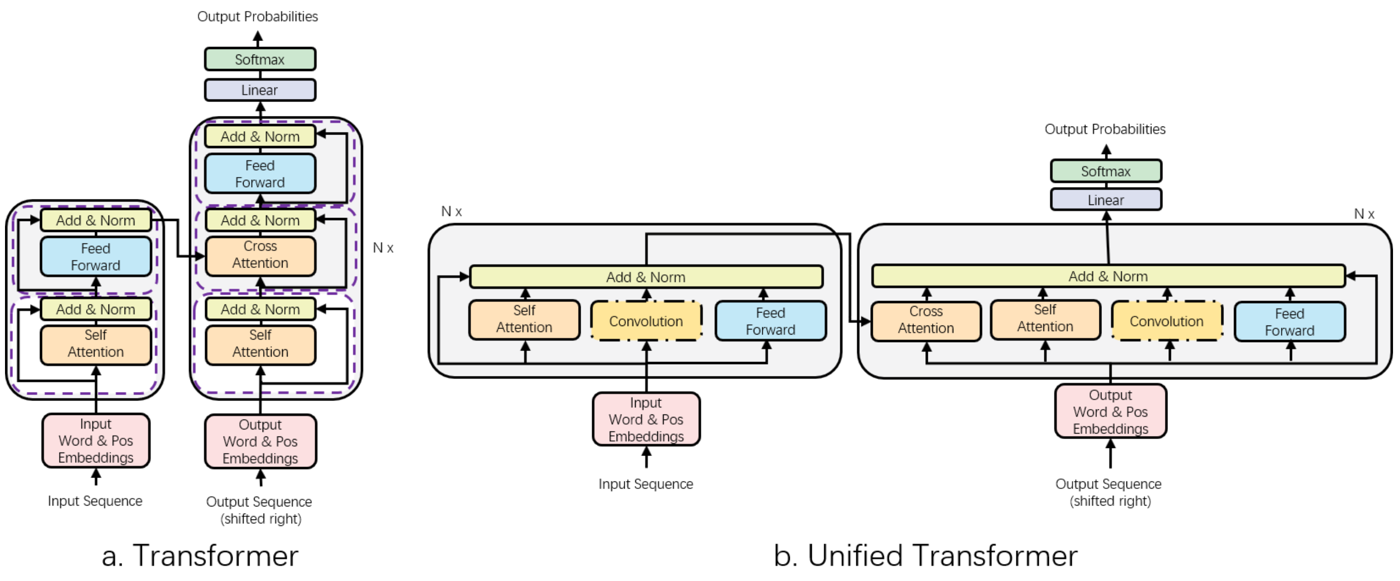

Figure 1: **(a): Transformer** uses multiple different neural systems (highlighted in purple dotted boxes) to process signals sequentially in both the Encoder and Decoder, without stimulating and collecting them at the same time. **(b): Unified Transformer** is built by simply stacking one neural assembly, the *P*a*R*allel *I*ntersected *M*ulti-scale Att*E*ntion (Prime) module. In contrast to the Transformer, where modules of different structures are activated sequentially, the Unified Transformer repeats the Prime module of the same structure, and sub-modules of different structures are activated concurrently, forming a neural assembly. The parallel structure of Prime is simple, in that it avoids the need to optimize the sequential order of the modules, and extensive, in that it facilitates the introduction of sub-modules with new functionalities (e.g., the convolution for contextual representations with finer-granularity).

tend to become "associated" so that activity in one facilitates activity in the other. The Transformer does not comply with this point. Systems of different structures in the Transformer are not activated simultaneously, so they are not effective in improving each other.

Motivation 2: Disadvantages of the pure self-attention for modeling contextual representations. Transformer relies solely on the self-attention mechanism to model intra-sequence token relations. Although self-attention updates token representations based on a weighted sum of all token representations in the same sequence, we find it unfavorable to transform long sequences. Previous studies empirically show that Transformer has surprising shortcomings in modeling long sequences precisely because of its use of self-attention on syntactic tasks [4] . We revisit this question and study the Transformer's ability to generate long sequences on machine translation and conduct preliminary experiments on the translation dataset of IWSLT De-En. As shown in Figure 2, longer sequences get significantly worse results. We hypothesize that attention maps can be over-concentrated and dispersed in long sequences, and visualizations of attention maps verify our hypothesis. We give an example in Figure 3, and this phenomenon is widespread in the last two layers of the Transformer. It may work fine for shorter sequences, but it brings difficulty for the model to comprehend the source information for longer sequences. In the latest work, local attention that constrains the attention to focus on only part of the sequences [5, 6, 7] helps address this problem. However, it costs self-attention to capture long-range dependencies and does not demonstrate effectiveness in sequence-to-sequence learning tasks.

Solution: Unifying complementary neural systems with the Unified Transformer. We propose a new simple network architecture, the Unified Transformer, to solve the two problems in Transformer. We plot the architecture in subfigure (b) of Figure 1. The Unified Transformer simplifies the Transformer architecture and is built by simply stacking one neural assembly – the PaRallel Intersected Multi-scale attEntion (Prime) module, in which multiple neural systems fire together and wire together. In

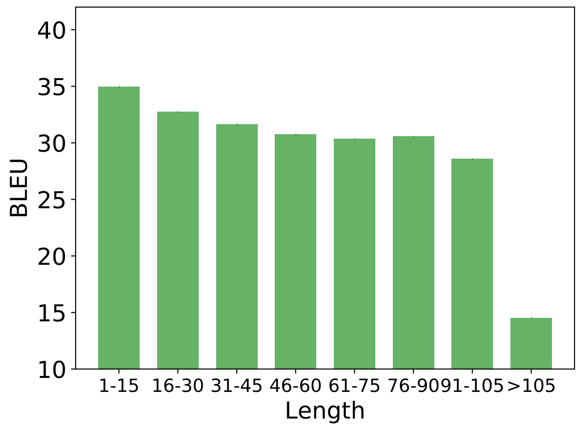

Figure 2: The performance of the Transformer drops significantly with the increase of sentence length in machine translation.

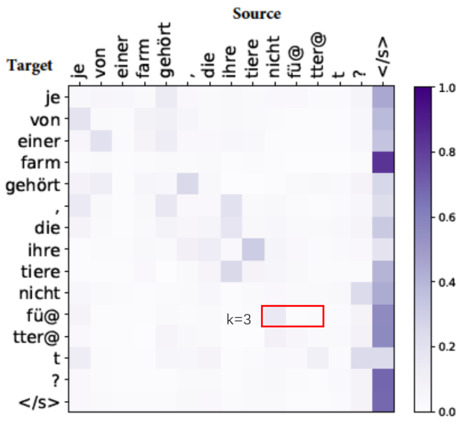

Figure 3: Illustration of an attention map from the 3rd Encoder layer. As we can see, the attention is too dispersed to capture sufficient information. Especially, "[EOS]", contributing little to word alignment, is surprisingly over attended. Simultaneous modeling of local relations with convolution can make up for the issue of dispersed attention in self-attention, e.g., for token "fu@", the window of convolution can be the red window with a kernel size of $k = 3$.

contrast to the original Transformer, where modules of different structures are processed sequentially, the Unified Transformer adopts the same Prime module, which in turn activates its components concurrently. This design solves the *first issue* of the Transformer and implements the basic idea of Unified Transformer. For a comparable structure in terms of processing abilities, we introduce Prime-simple, which is directly transformed from the Transformer. In the Encoder (left in Figure 1), FFN and self-attention are activated and collected at the same time. In the Decoder (right in Figure 1), Prime-simple also unifies the cross attention to collect features from the Encoder at the same time. The idea of unifying neural systems results in a more extensible model. To solve the *second issue* of Transformer, we extend Prime-simple to Prime, which unifies convolution that models local dependencies to make up for the shortcomings of pure attention for learning contextual representations. The granularity of convolution [8, 9] is between the FFN of learning context-independent representations [10] and the self-attention [11, 12, 1] of aggregating the global context. To model context of various granularity, Prime adaptively determines the kernel sizes of the convolutions. We also explore unifying multiple convolutions with different window sizes to capture richer context.

Challenges in implementing the solution. In implementation, we solve two challenges in realizing the solution. First, converting the sequential structure to the concurrent structure sparsely increases the width and reduces the depth (i.e., the number) of residual blocks, so it is not easy to use the original optimized training settings. We mitigate this problem by approximating the configuration of the original Transformer. For example, we increase the number of layers to reach a similar number of residual blocks and reduce the width of each neural system. Second, convolution simultaneously learns channel-level features and sequence-level fusion, and attention models them successively. Directly combining them leads to understanding contextual representations from different semantic spaces and harms the contextual feature fusion. Therefore, we adopt the depthwise separable convolution [13], and propose Shared Projection that first separates the channel-level feature learning and sequence-level feature fusion, and then fuses sequence-level features from different scales in the unified semantic space. Please refer to Section 4.2 for details and justifications.

Results. We show that the Unified Transformer built by simply stacking the unified unit, i.e., the Prime module, to be superior in translation quality over strong non-unified transformer, especially on transforming long sequences through experiments on large-scale machine translation tasks. Moreover, Unified Transformer generalizes well to the base model and small datasets.

We also show the unified unit has the potential for accelerating inference due to its parallelism.

Our main contributions are as follows:

- We first propose a unified version of the Transformer, which simplifies the Transformer to only one basic building unit; in that unit, multiple neural systems with complementary functions activate and gather together. Unified Transformer makes the Transformer that stacks complex modules of different nature sequentially comply with the cell assembly theory and does not suffer from orders of neural systems.
- We find that the operation of convolution and self-attention should be performed in the same semantic space when unifying them, and first successfully unify convolution and self-attention in sequence-to-sequence learning by overcoming the challenges with the proposed Shared Projection.
- Unified Transformer outperforms Transformer and other comparable, achieving state-of-the-art BLEU scores on three main machine translation tasks of various scales, including WMT14 En-Fr, WMT14 En-De, and IWSLT14 De-En. In addition, because the understanding of the semantics of words and phrases is not affected by the divergent global attention, Unified Transformer has dramatically improved the translation quality in long-sequence machine translation.

2. Unified Transformer with Parallel Intersected Multi-Scale Attention (Prime)

As discussed in the introduction, we introduce the Unified Transformer as the solution to the two issues of Transformer. We plot the overall architecture in Figure 1. Unified Transformer also adopts an Encoder-Decoder framework [14, 15, 16] to understanding the input sequence and generating the output sequence. The Encoder takes a sequence of word and position embeddings $(x_1, \cdots, x_n)$ as input where n is the length of input. It encodes word embeddings to build a contextual sequence representations $\boldsymbol{z} = (z_1, \cdots, z_n)$. Then, given $\boldsymbol{z}$, the Decoder predicts the words in the target sequence $(y_1, \cdots, y_m)$ sequentially. Unified Transformer is built by simply stacking one neural assembly. The PaRallel Intersected Multi-scale attEntion (Prime) module, in which multiple neural systems fire together and wire together. The Encoder is a stack of N Prime modules. Residual connection [17] and layer normalization [18] are used to connect two adjacent residual blocks. The Decoder is similar to the Encoder, except that each Prime module in the Decoder also performs attention over the Encoder stack's output through additional cross attention in parallel. In the following subsections, we omit the formalization of Prime in the Decoder and the bias term in linear projections for simplicity.

2.1. Prime-simple for Simultaneously Neural Computation

To build a unified Transformer architecture and let one neural system facilitate activate in the other. We introduce Prime-simple, in which two neural systems are active simultaneously and are aggregated simultaneously. We directly adjust Prime-simple from the sequential form of the Transformer without adding any new modules. In the Encoder (left in Figure 1), FFN and self-attention are activated and collected at the same time. In the Decoder (right in Figure 1), Prime-simple also unifies the cross attention to collect features from the Encoder at the same time. Prime-simple takes the output of $(i-1)$ layer as input and generates the output representation with the "Add & Norm" operation:

$$X_i = \mathrm{LN}(X_{i-1} + \mathrm{Attention}(X_{i-1}) + \mathrm{FFN}(X_{i-1})) \quad (1)$$

Where "Attention" refers to self-attention in Transformer, "FFN" refers to a point-wise feed-forward network in Transformer. The followings list the details of each part, "LN" refers to layer normalization.

Self-Attention network for learning global context representations. Self-attention is responsible for learning representations of the global context. For a given input sequence X, it first projects X into three representations, key K, query Q, and

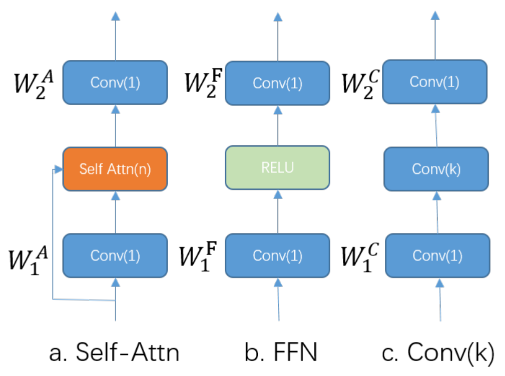

Figure 4: From left to right are self-attention (a), pointwise feedforward net (b), and convolution (c), respectively. The number in "()" is the kernel size. These modules all consist of an input projection, an output projection, and an intermediate operation. Transformer Encoder consists of the self-attention (a) and the pointwise feedforward net (b).

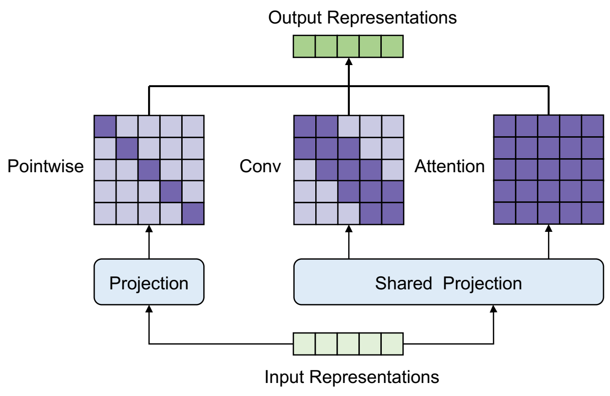

Figure 5: Prime, which is the building unit of the Unified Transformer, unifies point-wise feed forward neural nets, convolution, and self-attention to exploit their inductive biases, and simultaneously learn multi-scale sequence representations. Notably, we project convolution and self-attention into the same semantic space (by sharing W_1^A and W_1^C in Figure 4) to simultaneously learn contextual representations at the sequence-level.

value of attention V_A. Then, it uses the attention mechanism to get the contextual representations:

$$\begin{aligned} \text{Attention}(X) &= \text{SelfAttention}(V_A)W_2^A \\ \text{SelfAttention}(V_A) &= AV_A \\ A &= \text{softmax}(\frac{QK^T}{\sqrt{d_k}}) \\ Q = XW^Q, K = XW^K, V_A &= XW_1^A \end{aligned} \tag{2}$$

where W_1^A, W^Q, W^K, and W_2^A are projection parameters. The projection operation W_1^A can be viewed as 1-d convolution with kernel size 1, it is context-independent, and the aim is to get channel-level representations. Since W^Q and W^K are only used to calculating attention scores for global feature fusion at the sequence level, we view W_1^A as the input projections of the self-attention and view W_2^A as the output projections of self-attention. The intermediate operation is attention with parameters W^Q and W^K. We plot the process of the self-attention network in the left of Figure 4.

Pointwise feed-forward network for learning context-independent representations. To learn token-level representations, Prime-simple concatenates a self-attention network with a position-wise feed-forward network at each layer. Since the linear transformations are the same across different positions, the pointwise feed-forward network learns context-independent representations.

$$\text{Pointwise}(X) = \max(0, XW_1^F)W_2^F \tag{3}$$

W_1^F, W_2^F, are learnable parameters and X is the previous layer's output. We view W_1^F as the input projections of FFN and view W_2^F as the output projections of FFN. The intermediate operation is ReLU (Rectified Linear Units) [19] activation. We plot the process of the FFN in the middle of Figure 4.

Challenges in implementing Prime-simple. Since this parallelization method sparsely increases the width and reduces the number of residual blocks, so it is not easy to use the original optimization settings. We solve this problem by approximating the configuration of the original Transformer. Take the configuration of the base model as an example. We increase the number of layers from 6 to 12, so the total number of residual blocks of Encoder is still 12. We also reduce the width of each neural system in Prime from 512 to 384. But the initialization variance is inversely proportional to the square root of the width, so we change the initialization from the larger Xavier initialization to the initialization that is the default in PyTorch and has a minor variance. We also discuss the challenge and the solution in the section of Analysis.

2.2. *Prime for Assembling More Neural Systems*

Prime is an extended version of Prime-simple. We depict the Prime module in Figure 5. Prime in the Encoder of United Transformer additionally includes the convolution compared to Prime-simple. Prime contains three parts: self-attention for capturing global features, depth-wise separable convolution for capturing local features, and a position-wise feed-forward network for learning context-independent representations.

$$\begin{aligned} X_i = \mathrm{LN}(&X_{i-1} + \mathrm{Attention}(X_{i-1}) \\ &+ \mathrm{Conv}(X_{i-1}) + \mathrm{FFN}(X_{i-1})) \end{aligned} \tag{4}$$

where the "Conv" refers to depthwise-separable convolution [13]. The convolution compensates for the insufficient use of local information, while the self-attention focuses on capturing the global dependencies.

Convolution network for local context modeling. We introduce convolution operations into Prime to the local context. To learn contextual sequence representations in the same hidden space, we choose depth-wise convolution [13, 20] (we denote it as DepthConv in the experiments) as the convolution operation because it includes two separate transformations, namely, context-independent pointwise projection transformation and contextual transformation. That original convolution operator is not separable, but DepthConv can share the same pointwise projection transformation with the self-attention and enables parallel feature fusion at different context lengths in a unified space. We choose dynamic convolution [21], the best variant of DepthConv, as our implementation.

Each convolution sub-module contains multiple cells with different kernel sizes. We use them for learning different-range features. The output of the convolution cell with kernel size k is:

$$\begin{aligned} V_C &= XW_1^C \\ \mathrm{Conv}_k(X) &= \mathrm{DepthConv}_k(V_C){W_2}^C \end{aligned} \tag{5}$$

where W_1^C and ${W_2}^C$ are input and output pointwise projections with learn-able parameters. The DepthConv refers to depth

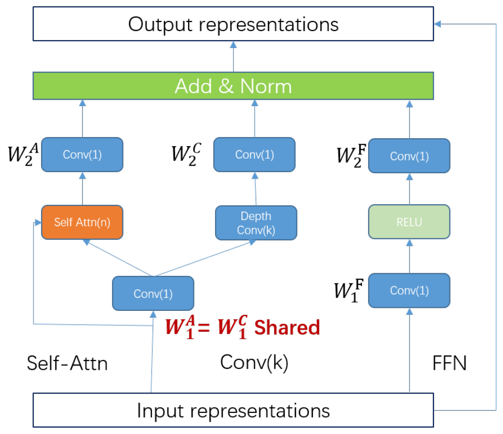

Figure 6: Prime structure in more detail. Prime has a shared projection between the convolution branch and the self-attention branch.

convolution in the work of [21]. For an input sequence X, the output D is computed as:

$$\begin{aligned} D_{i,c} &= \mathrm{DepthConv}_k(X) \\ &= \sum_{j=1}^{k} \big(\mathrm{softmax}(\sum_{c=1}^{d} W_{j,c}^C X_{i,c}) \cdot X_{i+j-\lceil \frac{k+1}{2} \rceil, c}\big) \end{aligned} \tag{6}$$

where d is the hidden size, W^C is the inherent parameter in the intermediate operation. We plot the process of the FFN in the middle of Figure 4, the "Conv(k)" is $\mathrm{DepthConv}_k$ in Prime, we discuss choices in experiments.

Unifying self-attention and convolution with the Shared Projection. The challenge in building Prime is that the depthwise-separable convolution and the self-attention learn contextual representations from different semantic space. To simultaneously learn contextual sequence representations at multiple scales in a unified semantic space, we exploit the **Shared Projection** to project the channel-wise features into the same hidden space. In Shared Projection, the projection in the self-attention mechanism $V_A = XW_1^A$ and that in the convolution mechanism $V_C = XW_1^C$ is shared, that is $W_1^C = W_1^A$, we plot it in Figure 6. It is opposite to the **Separate Projection** with two independent projections: $V_A = XW_1^A$, $V_C = XW_1^C$, where W_1^A and W_1^C are two different parameter matrices. We analyze the necessity of Shared Projection in Section 4.2.

Adaptive kernel size. We also explore more convolutions with various window sizes to capture richer context and let the model decide whether it needs a large kernel size during training. The kernel size determines the length of local context features. Previous works [20, 21] regard it as a hyper-parameter and grow the kernel size from the bottom layer to the top layer. To automatically learn the length of context, we introduce multiple convolution cells and a gate between different convolution cells.

$$\mathrm{Conv}(X) = \sum_{i=1}^{n} \frac{\exp(\alpha_i)}{\sum\limits_{j=1}^{n} \exp(\alpha_j)} \mathrm{Conv}_{k_i}(X) \tag{7}$$

where α_i is a learn-able parameter, n is the number of cells; we adopt kernel sizes of 3 and 15 in this setting without trying other kernels because they take into account the short-range (size=3) and the long-range (size=15) feature fusion. The context length of these convolutions is between the pointwise FFN that learns context-independent representations and the self-attention that learns representations from global context.

3. Experiment

In this section, we empirically verify the benefits of the Unified Transformer and justify our choices. We organize this section as follows. First, we elaborate on details of the experimental setup for reproduction. Second, we show that the Unified Transformer with the Prime module achieves state-of-the-art results on translation datasets. Third, we show that the unified schema is favorable to parallel computation. It also brings significant improvements when generating long sequences and improves translation quality in practice. Lastly, we analyze the two challenges and corresponding solutions encountered when building the United Transformer.

3.1. Experiment Setup

This subsection describes the model configuration, the datasets, and experimental settings for training and evaluation. The hyper-parameters in the experiment setting follow the best-tuned Transformer [22] and Dynamic Convolution [21] without any change. For experiments on large datasets, we adopt their pre-processing (Including BPE tokenization [23], vocab size) and training settings (batch size, number of steps, optimizer.). For experiments on small datasets. We tune the best hyper-parameters for Transformer on the valid set and directly adopt these hyper-parameters for our model. Our implementation is based on `fairseq`[1], which is the codebase that contains the best tuned Transformer [22] and a competitive model Dynamic-Convolution [21]. We also use pre-processing tool `process` as well as the evaluation method `fairseq-score`. For reproducibility, we describe the experimental details for training and evaluation below.

3.1.1. Datasets

The typical application of sequence-to-sequence learning is the machine translation task. Canonical models including Seq2seq [16], Transformer [1] are all tested on the machine translation task. In machine translation, we need to translate a sequence of words into a sequence with the same meaning and another language form. We test and compare our model on two large-scale datasets and two small datasets.

WMT14 En-Fr and En-De datasets. The WMT 2014 English-French translation dataset, consisting of $36M$ sentence pairs, is adopted as the extremely large-scale machine dataset to test our model at the large size (200–250M). We use the standard split of development set and test set. We use *newstest2014* as the test set and use *newstest2012* +*newstest2013* as the development set. Following [24]. For the large dataset, we borrow the setup of Transformer [1] and adopt the WMT 2014 English-German translation dataset, which consists of $4.5M$ sentence pairs, the BPE size in this dataset is $32K$. The test and validation datasets we used are the same as Transformer [1].[2]

IWSLT De-En and En-Vi datasets. Besides, we perform experiments on two small IWSLT datasets to compare the small version of Unified Transformer with other models at the base size (40–50M). The IWSLT 2014[3] German-English translation dataset consists of $160K$ sentence pairs. The IWSLT 2015[4]

[1] `https://github.com/pytorch/fairseq`

[2] Please refer to `http://www.statmt.org/wmt14/` for details of these two datasets in WMT 2014.

[3] Link to IWSLT2014: `https://workshop2014.iwslt.org/`

[4] Link to IWSLT2015: `https://workshop2015.iwslt.org/`

English-Vietnamese translation dataset consists of $133K$ training sentence pairs. For the En-Vi task, we build a dictionary including all source and target words. The vocabulary size for English is $17.2K$, and the vocabulary size for Vietnamese is $6.8K$.

3.1.2. Evaluation

We adopt beam search with a beam size of 5 for De-En, En-Fr, and En-Vi translation tasks during inference. Length penalty is set to 0.8 for En-Fr according to the validation results, and 1 for the two small datasets following the default setting [25]. The BLEU[5] metric [26] is adopted to evaluate the model performance during evaluation.

3.1.3. Model Configuration

We only compare models reported with the comparable model size (40–50M for the base model, 200–250M for the big model) and the same training data for fair comparisons. We do not compare [27] because it is an ensemble method. We build a base and a big version of United Transformer. Their parameter sizes are comparable to Transformer-base and Transformer-big, respectively. As to the depth, United Transformer consists of 12 residual blocks for both the Encoder and the Decoder.

Since CUDA and the fairseq code base only allow dimension that is integer times of 64, we choose the dimension of 384 and 768 for the base and the big model, respectively, which results in a parameter size that closes to the Transformer. The base model of Unified Transformer with Prime-simple and Prime has 44M and 49M parameters, respectively, and the big version has 233M and 243M parameters, respectively. On the other hand, Transformer big and Transformer base parameter sizes are 210M and 41M, respectively. For completeness of introduction, the proposal can also beat a 2x larger model of Transformer with more layers, a better initialization, and optimization. For example, Transformer can grow to 92M and 560M parameters to get 35.6 and 29.6 BLEU scores on IWSLT De-En and WMT14 En-De, respectively, but the increased capacity (2x parameter size) does bring much fewer gains than ours [28]. Nonetheless, comparing with much larger models is not the main pursuit of this paper.

We adopt multi-head attention [1] as the implementation of self-attention in Prime modules. The number of attention heads are the same as that in the Transformer. As for convolution, we adopt dynamic convolution as the variant of depth-wise separable convolution. We tune the kernel size on the validation set. For convolution with a single kernel, we try kernel size in $\{3,7,15,31\}$ and use the kernel size of 7 for all layers. For dynamically selected kernels, the kernel size is 3 for the small kernel and 15 for the large kernel for all layers. We build Prime-simple similarly.

3.1.4. Training

Models with the big size are trained on 4 Titan RTX GPUs, while base models are trained on a single NVIDIA RTX 2080Ti GPU. In addition, the batch size mentioned below is at the token level.

Training for the big Unified Transformer. For training big Unified Transformer, the hyper-parameters are the same as that in [21]. Details are as follows. Parameters are updated every 32 step. We train the model for $80K$ updates with a batch size of 5120 for En-Fr, and train the model for $30K$ updates with a batch size of 3584 for En-De. The dropout rate is 0.1 for En-Fr and 0.3 for En-De. We use the cosine learning rate schedule with 10000 warmup steps. The max learning rate is 0.001 on En-De translation and 0.0007 on En-Fr translation. For checkpoint averaging, following [21], we tune the average checkpoints for En-De translation tasks. For En-Fr translation, we do not average checkpoint but use the final single checkpoint. It takes 2–3 days for training on WMT En-De and ten days for training on WMT En-Fr. The expected best validation PPL of WMT En-De and WMT En-Fr is 3.75 and 3.09, respectively.

Training for the base Unified Transformer. For models at the size of "base" (40–50M), we tune the best hyper-parameters for

[5]https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl

Model	En-De	En-Fr
ConvSeq2seq [24]	25.2	40.5
SliceNet [20]	26.1	-
Convolutional Self-attention [29]	28.7	-
Reformer [30]	29.1	-
DynamicConv* [21]	29.4	43.0
Transformer [1]	28.4	41.0
Transformer (relative position embedding) [31]	29.2	41.5
Transformer* (well-tuned the optimization) [2]	29.1	42.9
Transformer (with fixup initialization) [32]	29.3	-
Layer reordering [33]	28.7	-
Weighted Transformer [34]	28.9	41.4
Multi-Granularity Self-Attention [35]	29.0	-
Layer-wise Coordination [36]	29.1	-
Evolved Transformer [37]	29.8	41.3
Unified Transformer (Prime-simple)	29.8	43.2
Unified Transformer (Prime)	**29.9**	**43.5**

Table 1: BLEU scores of Unified Transformer and previous models of the big size on WMT14 En-De and WMT14 En-Fr datasets. * refers to our re-implementation. The proposal outperforms all these models.

the Transformer baseline and directly use these hyper-parameters from our model. We train and test Prime-base on two small datasets, IWSLT 2014 De-En translation, and IWSLT2015 En-Vi translation. Following the Transformer [1], we use Adam optimizer with a learning rate of 0.001. We use the warmup mechanism and invert the learning rate decay with warmup updates of $4K$. For the De-En dataset, we train the model for $20K$ steps with a batch size of $4K$. The parameters are updated every 4 step. The dropout rate is 0.4. For the En-Vi dataset, we train the model for $10K$ step with a batch size of $4K$. The parameters are also updated every 4 step. The dropout rate is 0.3. We save checkpoints every epoch and average the last 10 checkpoints for inference. It takes 2–3 hours for training on IWSLT De-En and an hour for training on IWSLT En-Vi. The expected best validation PPL of IWSLT De-En is around 4.6.

3.2. *Main Results*

We report results on large translation datasets and small translation datasets, respectively.

Results on Large-scale Datasets. We summarize the comparisons between the Unified Transformer and the baselines displayed in Table 1 as follows.

- As depicted in the first group, previous studies add the local context constraint to Transformer- [24, 20, 29, 30, 21]. However, they do not unify other neural systems and our Unified Transformer surpasses them.

- As depicted in the second group, some researchers try to keep the architecture of Transformer [1] unchanged, but add relative positional encoding [31] or tune the settings for training Transformer[22], or modify the initialization [32]. Unified Transformer significantly outperforms these advanced Transformer models.

- As depicted in the third group, some studies are also interested in maximizing the exploitation of information in each layer. First, Layer Reordering [33] proposes to adjust the order of self-attention and FFN, but they can only get +0.1 BLEU scores compared to their Transformer baseline. In addition, the order in this sequential mode should be carefully manually crafted and inconsistent among datasets. In comparison, the Unified Transformer consistently achieves significant improvements on various translation datasets. Second, Weighted Transformer [34] builds a multi-branch model, and each branch is a Transformer model. Third, Multi-Granularity Self-Attention [35] exploits multi-granularity through syntactic trees. Lastly, Layer-wise Coordination[36] unifies the Encoder and Decoder. Unified Transformer also outperforms these three models.

- As depicted in the fourth group, compared to the Evolved Transformer [37], which is constructed by network architecture search, and mixes convolutions of different kernel sizes, search for the best unification in the sequence-level,

Model	En-Vi	De-En
NBMT [38]	28.1	30.1
SACT [39]	29.1	-
NP2MT [40]	30.6	31.7
Transformer (fixup initialization) [32]	-	34.5
Mixed Multi-Head Self-Attention [41]	-	35.4
Transformer* [1]	30.6	35.3
DynamicConv* [21]	30.7	35.7
Unified Transformer (Prime-simple)	30.7	35.8
Unified Transformer (Prime)	**31.3**	**36.3**

Table 2: Comparisons between the Unified Transformer of the base size and SOTA models on IWSLT De-En and En-Vi, * refers to our re-implementation.

Unified Transformer achieves +2.2 BLEU gains in En-Fr translation.

These results show that the Unified Transformer, in which complementary neural systems activate and collect at the same time facilitate each other, improves the overall performance. Since the variation of these results under three different initialization is less than 0.2 (Wu et al. also reported a similar value [21]), the improvement of our model over previous work is significant.

Results on Small Datasets and the base model. Unified Transformer can also scale to base models and small datasets, as depicted in Table 2. For example, the base model of Unified Transformer pushes the state-of-the-art from 35.7 to 36.3 on the IWSLT De-En translation dataset. Mixed Multi-Head Self-Attention [41] also mix local and global attention patterns, but they do not change the sequential stack of FFN and self-attention, and the diversity of their mix is not as high as ours (We mix convolution and pointwise FFN). Hence, Mixed Multi-Head Self-Attention gets worse results than the Unified Transformer.

Results of the Unified Transformer with Prime-simple. Table 1 and Table 2 show that Prime-simple, which contains the basic idea of Unified Transformer, also achieves good performance.

Model	Inference Speed (tokens/s)
Transformer	132
Unified Transformer	173
Acceleration	+31%

Table 3: The comparison between the inference speed of Unified Transformer and Transformer.

3.3. Benefits of the Unified Transformer

In this subsection, we show that the unified schema is favorable to parallel computation. It also brings significant improvements when generating long sequences and improves translation quality in practice.

3.3.1. Parallel Design Brings Time Efficiency on GPUs

The underlying parallel structure (compared to the sequential design in Transformer) of the Unified Transformer's Prime unit allows GPUs to compute it efficiently. For example, we can combine small matrices into large matrices, and while it does not reduce the number of actual operations, it can be better paralleled by GPUs to speed up computation. For each Prime unit, we first concentrate W^Q, W^K, W_1^A of self-attention and W_1 of point feed-forward transformation into a single Encoder matrix W^{Enc}. And then perform transformation such as the operation of dot-product attention, depth-separable convolution, and nonlinear transformation ReLU in parallel to learn multi-scale representations in the hidden layer. W_2^A, W_2, W_2^C can also be combined as a single Decoder matrix W^{Dec}. We implement the Decoder similarly.

In Table 3, we conduct comparisons to show the speed gains with the implementation above, and the batch size is set to one sample per batch to simulate the online inference environment. Under the settings where the numbers of parameters are similar for Prime and Transformer, we obtain about a +31% increase in inference speed. The experiments use Unified Transformer with 6 Prime-simple modules and Transformer with 6 base blocks, and the hidden size is 512. Under this setting, we adopt the initialization that is tuned on the valid set for the Unified Trans-

Case 1	
Source	wenn sie denken, dass die auf der linken seite jazz ist und die, **auf der rechten seite swing ist, dann klatschen sie bitte.**
Target	if you think the one on the left is jazz **and the one on the right is swing, clap your hands.**
Transformer	if you think it's jazz on the left, **and those on the right side of the swing are clapping, please.**
Unified Transformer	if you think the one on the left is jazz, **and the one on the right is swing, please clap.**
Case 2	
Source	wir sind an der schwelle zu erstaunlichen, erstaunlichen ereignissen auf vielen gebieten. und doch denke ich wirklich, dass wir hunderte, 300 jahre vor die aufklärung zurück gehen müssten, um eine zeit zu finden, in der wir fortschritt bekämpft haben, in der wir über diese dinge **heftiger getritten haben, an mehr fronten als jetzt.**
Target	we're on the verge of amazing , amazing events in many fields , and yet i actually think we'd have to go back hundreds, 300 years, before the enlightenment, to find a time when we battled progress, when we fought about these things **more vigorously, on more fronts, than we do now.**
Transformer	we're at the threshold of amazing, amazing events in many areas, and yet i really think we have to go back hundreds and 300 years before the enlightenment to find a time when we fought progress in which we've driven over these things **more recently than now.**
Unified Transformer	we're at the threshold of amazing , amazing events in many areas, and yet i really think that we have to go back hundreds and 300 years before the enlightenment to find a time when we've been fighting progress, where we've driven these things **more fiercely, on more frontiers than we are now.**

Table 4: Case study on the De-En translation dataset. The orange bolded words denote the **source** sentences. The red bolded words denote the **wrong** translation and blue bolded words denote the **correct** translation. In case 1, Transformer, without simultaneously modeling local context when modeling global context, fails to capture the relationship between some words and their neighbors, such as "right" and "clap", but Unified Transformer can deal with this issue. In case 2, Transformer with the global self-attention results in the lack the semantic modeling of the word itself. Transformer translates "fronten" into "recently" and lost the word "vigorously". In addition, Transformer also poorly models the local semantic information of phrases. Therefore, Transformer generates an unreasonable phrase "more recently than now". Unified Transformer simultaneously models word semantics, and phrase semantics with FFN and convolution respectively, solves these problems.

former and get a BLEU score of 35.3, which is the same as that in the Transformer.

3.3.2. *Quantitative Result: Better Generation for Long Sequences*

As demonstrated in Figure 7, Unified Transformer with the Prime unit generates better sequences of various lengths than self-attention, but it is remarkably adept at generating long sequences, e.g., for sequences longer than 100, the proposed model is two times better.

3.3.3. *Case Study: Improved Translation Quality in Practice*

We conduct the case study on the De-En dataset and show the cases in Table 4 (at the end of this paper). They show that it generates more reasonable sentences and increases translation quality. In case 1, although the baseline transformer translates many correct words according to the source sentence, the translated sentence is not fluent. Moreover, it indicates that the Transformer does not capture the relationship between them and their neighbors, such as "right" and "clap". By contrast, Unified Transformer captures them well by combining local convolution

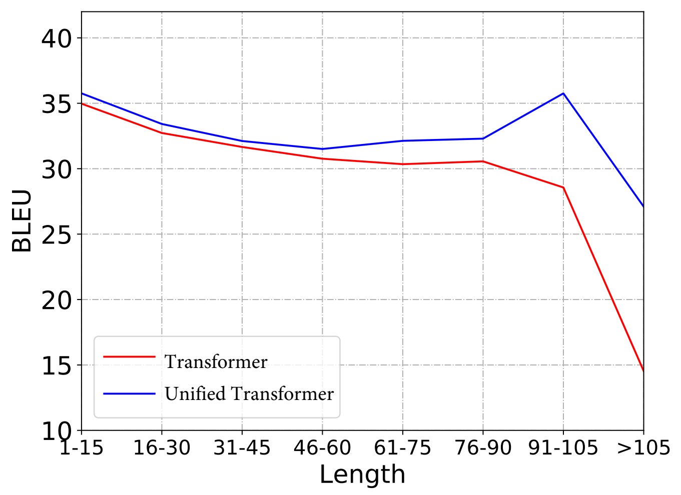

Figure 7: BLEU scores of models on different groups at different lengths of source sentences. The experiments are conducted on the IWSLT De-En dataset. Unified Transformer with the Prime unit improves over Transformer, especially on long sentences.

with global self-attention. In case 2, Transformer with the global self-attention results in the lack the semantic modeling of the word itself. Thus, for example, Transformer translates ”fronten” into ”recently” and lost the word ”vigorously.” In addition, Transformer also poorly models the local semantic information of phrases. Therefore, Transformer generates an unreasonable phrase ”more recently than now.” Unified Transformer simultaneously models word semantics and phrase semantics with FFN, and convolution solves these problems. Nevertheless, they all have room for improvement in the syntactic structure. For example, they did not realize that the two ”in der” (where) are parallel, so they understand the latter sentence as ”driven the progress” (progress is is mentioned in the former).

4. Analysis

We analyze the challenges and corresponding solutions encountered when building the United Transformer.

4.1. Re-scaling the Model Size Solves the Challenge in Implementing the Prime-simple

Since this parallelization method sparsely increases the width and reduces the number of residual blocks, so it is not easy to use the original optimization settings. We solve this problem by

Model	Params	BLEU
Transformer (6 L, 30 Blocks, 512 Dim)	41M	35.3
Transformer (12 L, 60 Blocks, 512 Dim)	74M	35.0
Unified Transformer (6 L, 12 Blocks, 512 Dim)	41M	35.0
Unified Transformer (12 L, 24 Blocks, 384 Dim)	44M	35.8

Table 5: Results on model size calibrations. Blocks refer to the residual blocks. The dimension is calculated at the level of embedding, so the Unified Transformer increase the width in a sparse way. We re-scale the Unified Transformer with the Prime-simple unit to make its size close to that of the Transformer. Results are from base models on the IWSLT 2015 De-En Translation Task.

making the configuration of the Unified Transformer similar to the original Transformer. As shown in Table 5, we increase the number of layers from 6 to 12, which makes the total number of residual blocks is close to that of the Transformer. We also reduce the width of each neural system in Prime-simple from 512 to 384 so that the overall width will not be too large, e.g., it will be 768 in the Encoder. However, results in Table 5 also tell us that increasing the parameter size does not necessarily lead to improvement.

4.2. The Shared Projection Solves the Challenge in Unifying Convolution and Self-attention

Directly combining self-attention and convolution with separate projections does not work in the Unified Transformer.

Does concatenating self-attention with convolution certainly improve the model?. To bridge the gap between self-attention and a pointwise transformation that learns token-level representations and self-attention that learns global context representations, we introduce convolution to enhance our multi-scale attention. As we can see from the first experimental group of Table 6, convolution is important in parallel multi-scale attention. However, combining convolution and self-attention in one module is not easy to build better representations on sequence-to-sequence tasks. As shown in line1 and line 5 of group 2 of Table 6, simply learning local representations by using convolution or depth-wise separable convolution in parallel with self-attention harms the performance. Furthermore, combining depth-wise

	Model or Module Configuration	BLEU
	Unified Transformer (Prime, Unify self-attention and DepthConv with Shared Projection.)	36.3
	Unified Transformer (Prime-simple, without DepthConv)	35.8
	Transformer	35.3
Separate Proj.	Prime (Substitute DepthConv with Convolution (K=3))	35.2
	Prime (Substitute DepthConv with Convolution (K=5))	35.0
	Prime (Substitute DepthConv with Convolution (K=7))	34.5
	Long-short range attention (Unify self-attention and convolution without Shared Projection) [42]	35.1
	Prime (DepthConv(K=3) without Shared Projection)	34.9
Shared Proj.	Prime (DepthConv(K=3) with Shared Projection)	36.2
	Prime (DepthConv (K=7) with Shared Projection)	36.2
	Prime (DepthConv (K=15) with Shared Projection)	36.0
	Prime (DepthConv (K=31) with Shared Projection)	35.8
	Prime (DepthConv (Grow K among layers:3,7,15,31) with Shared Projection)	35.9
	Prime (Unify DepthConv (Adaptively selected kernel (k=3,15)) with Shared Projection)	36.3

Table 6: Comparisons between the Unified Transformer with different variants of Prime on the IWSLT 2014 De-En translation task. This table is to show the effectiveness of the Shared Projection and the adaptive kernel.

separable convolution (in this work, we choose its best variant dynamic convolution as implementation) is even worse than combining convolution. Long-short range attention (Line4, group2) is recent work for combining attention and DepthConv in Lite Transformer. We re-implement the comparable model of 50M parameter size (dimension is 720). But it also does not work with the separate projection.

Existing methods learn channel-level representations and sequence-level feature fusion, respectively, but this independence can not make them complement each other to learn real multi-scale feature fusion.

Why do we choose DepthConv and what is the importance of Shared Projection?. We conjecture that convolution and self-attention both learn contextual sequence representations, and they should share the pointwise transformation and perform the contextual transformation in the same hidden space. Thus, the model can learn multi-scale contextual feature fusion simultaneously.

In contrast to DepthConv, standard convolution is not separable and simultaneously learns the position features and channel features. We first project the input to a hidden representation and perform a variant of depth-wise convolution and self-attention transformations in parallel.

The comparison between line5 of group2 and line1 of group3 in Table 6 validates the utility of sharing projection in parallel multi-scale attention. Furthermore, the Shared Projection gains 1.4 BLEU scores over the Separate Projection and achieves the improvement of 0.5 BLEU scores over the Prime without DepthConv.

4.3. *Pointwise FFN is Essential to Learn Context-Independent Representations*

Below, we share the input projection between pointwise FFN and convolution to examine the role of pointwise FFN in the proposed parallel intersected multi-scale attention. In the experiment, we keep the self-attention path unchanged.

We can see from Table 7 that as the kernel size increases from 3 to 31, the model's BLEU score gradually decreases from

Kernel size	BLEU
Prime	36.3
Prime without DepthConv	35.8
Prime without FFN	33.5
Share input Projection of FFN with DepthConv (k=3)	35.2
Share input Projection of FFN with DepthConv (k=7)	35.1
Share input Projection of FFN with DepthConv (k=15)	34.7
Share input Projection of FFN with DepthConv (k=31)	34.2

Table 7: Evidence that FFN needs separate input projection. FFN that is also viewed as point-wise attention and DepthConv that learn contextual representation have different functionality, so they can not activate in the unified space.

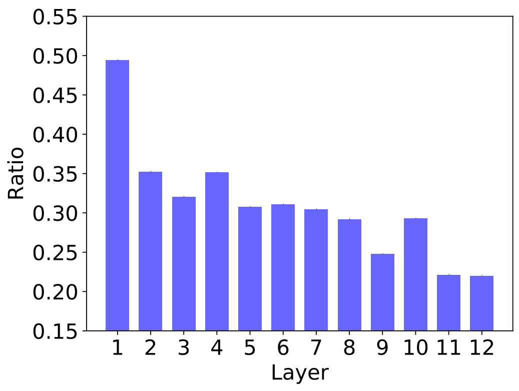

Figure 8: Adaptively selected kernels at each layer: The blue bars represent the ratio between the percentage of the convolution with smaller kernel sizes and the percentage of the convolution with large kernel sizes.

35.2 to 34.2. [6] This result further demonstrates the importance of pointwise FFN as a context-independent representation learner in the proposal. Therefore, it is better to do feature fusion among convolution and self-attention since they both learn contextual representations.

4.4. Adaptive Window Size in the Convolutions

Prime contains multiple dynamic convolution cells whose streams are unified by a gated mechanism. The weight for each dynamic cell is a scalar. Here we analyze the weight of different dynamic convolution cells in different layers. Figure 8 shows that as the layer depth increases, the weight of dynamic convolution cells with small kernel sizes gradually decreases. Furthermore, it demonstrates that lower layers prefer local features while higher layers prefer global features. It is also corresponding to the finding in [43].

5. Related Work

Models in sequence-to-sequence learning. Sequence-to-sequence learning is a benchmark task in natural language processing, which evolves understanding sequence and generating sequence. Machine translation is the touchstone of sequence-to-sequence learning. Traditional approaches usually adopt long-short term memory networks [16, 44] to learn representations of sequences auto-repressively. Recent studies explore convolutional neural networks (CNN) [24] or self-attention [1] to support high-parallel sequence modeling and do not require auto-regressive structure during encoding, thus bringing considerable efficiency improvements. In addition, they are strong at learning local or global dependencies.

Implementations of the cell assembly theory and deep learning. Assembly theory indicates that two neuron or two neural systems that are active simultaneously tend to become 'associated' so that activity in one facilitates activity in the other. At the level of the neural system, researchers have built state-of-the-art models in terms of convolution and depth separable convolution, respectively, where multiple neural systems are active at the same time, [45, 46]. They all have one thing in common: each layer is equivalent in form, and then multiple neuron systems within each layer starting from one vector and then merge into one vector and work in a parallel way. We are the first to introduce this idea to the Transformer. We let FFN [10], convolution [20, 21], self-attention [1], and cross attention [24, 1] originate from a vector and import into a vector in each layer.

Multi-branch and multi-scale neural networks. Multi-branch nets are general implementations of cell assembly theory. However, it does not require branches of each layer to start together and gather together. The Prime module in the Unified Transformer is related to successful multi-branch nets [45, 47, 46] but

[6] We report the results for ablations in the Encoder since the BLEU is much lower in the Encoder-Decoder case.

Prime focuses on learning sequence representations and incorporate self-attention. Recent studies have introduced the idea of multi-branch networks into sequence-to-sequence-learning [34, 48], but branches of their model are identical and can not simultaneously unify various branches with different inductive biases. Compared to other models which incorporate local attention to learn both global and local pattern [49, 41, 50, 35], Prime designs a parallel schema that can take advantage of convolution and positional feed-forward networks. The comparison between Prime and Mixed Multi-Head Self-Attention [41] in Table 2 also verifies the benefit of Prime over other multi-branch nets.

Unifying self-attention and convolution. Cordonnier et al. provide evidence that attention layers can perform convolution, and they often learn to do so. There are several studies on combining self-attention and convolution [3]. However, they do not surpass both convolution and self-attention mechanisms. Sukhbaatar et al. propose to augment convolution with self-attention by directly concentrating them on computer vision tasks [51]. However, as demonstrated in Table 6 their separate projection method does not work for the sequence-to-sequence learning task. Since the state-of-the-art models on question answering tasks still consist of self-attention and do no adapt ideas in QAnet [52]. Both self-attention [2] and convolution [21] outperform Evolved Transformer by near 2 BLEU scores on En-Fr translation. It seems that learning global and local context through stacking self-attention and convolution layers do not beat either self-attention or convolution models. In contrast, the proposed parallel multi-scale attention outperforms previous convolution or self-attention-based models on main translation tasks, showing its effectiveness for sequence-to-sequence learning. We demonstrate in section 4.2 that the long-short range attention [42] in the lite Transformer, which combines Depth-conv and self-attention but learns sequence-level feature fusion on separate semantic space, does not work at the base size.

6. Conclusion and Future work

In this work, we observe that Transformer that sequentially stacks multiple neural systems with different functions suffers from their orders, and it does not comply with the cell assembly theory. Also, Transformer relies solely on the attention mechanism to aggregate contextual representations globally, and attention is easy to disperse. This issue deteriorates the learning of words and phrases when transforming long sequences. We then present the Unified Transformer to solve these issues, and Unified Transformer simplifies the Transformer to only one building unit. Multiple complementary neural systems, including the FFN, convolution, and self-attention, are in that unit. These neural systems compensate each other, activate together, and wire together to form a cell assembly. We also deal with several challenges in building the Unified Transformer and get some interesting findings. For example, we should perform multi-scale contextual representations learning in a unified semantic space; FFN is an essential module for learning context-independent representations.

In the experiments, we show that on large-scale machine translation tasks, the Unified Transformer, built by simply stacking this unified unit – Prime module, is superior in translation quality over non-unified Transformer, especially on transforming long sequences. We also show that it generalizes well to the base model and small datasets. In addition, because the understanding of the semantics of words and phrases is not affected by the divergent global attention, Unified Transformer has dramatically improved the translation quality in long-sequence machine translation. Besides, the unified unit has the potential for accelerating inference due to its parallelism.

For future work, the parallel structure of the Unified Transformer is highly extensible and provides opportunities to improve these models. Moreover, the Prime module in the Unified Transformer may also shed light on search space design in NAS [53]. Finally, we are excited about the future of the Unified Transformer. We plan to apply this simple but effective idea to other sequence-to-sequence learning tasks, including summarization and speech recognition.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need (2017). arXiv:1706.03762.

[2] M. Ott, S. Edunov, D. Grangier, M. Auli, Scaling neural machine translation, arXiv preprint arXiv:1806.00187.

[3] J. Cordonnier, A. Loukas, M. Jaggi, On the relationship between self-attention and convolutional layers, CoRR abs/1911.03584. arXiv:1911.03584.
URL http://arxiv.org/abs/1911.03584

[4] G. Tang, M. Müller, A. Rios, R. Sennrich, Why self-attention? A targeted evaluation of neural machine translation architectures, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018, 2018, pp. 4263–4272.

[5] R. Child, S. Gray, A. Radford, I. Sutskever, Generating long sequences with sparse transformers (2019). arXiv:1904.10509.

[6] S. Sukhbaatar, E. Grave, P. Bojanowski, A. Joulin, Adaptive attention span in transformers, Proceedings of the 57th Annual Meeting of the Association for Computational Linguisticsdoi:10.18653/v1/p19-1032.
URL http://dx.doi.org/10.18653/v1/p19-1032

[7] B. Yang, L. Wang, D. F. Wong, L. S. Chao, Z. Tu, Convolutional self-attention networks, Proceedings of the 2019 Conference of the Northdoi:10.18653/v1/n19-1407.
URL http://dx.doi.org/10.18653/v1/n19-1407

[8] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al., Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.

[9] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 25, Curran Associates, Inc., 2012, pp. 1097–1105.
URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[10] J. Devlin, Sharp models on dull hardware: Fast and accurate neural machine translation decoding on the CPU, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Copenhagen, Denmark, 2017, pp. 2820–2825. doi:10.18653/v1/D17-1300.
URL https://www.aclweb.org/anthology/D17-1300

[11] J. Cheng, L. Dong, M. Lapata, Long short-term memory-networks for machine reading, Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processingdoi:10.18653/v1/d16-1053.
URL http://dx.doi.org/10.18653/v1/D16-1053

[12] Z. Lin, M. Feng, C. N. dos Santos, M. Yu, B. Xiang, B. Zhou, Y. Bengio, A structured self-attentive sentence embedding (2017). arXiv:1703.03130.

[13] F. Chollet, Xception: Deep learning with depthwise separable convolutions, 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)doi:10.1109/cvpr.2017.195.
URL http://dx.doi.org/10.1109/CVPR.2017.195

[14] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder–decoder for statistical machine translation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, 2014, pp. 1724–1734. doi:10.3115/v1/D14-1179.
URL https://www.aclweb.org/anthology/D14-1179

[15] N. Kalchbrenner, P. Blunsom, Recurrent continuous translation models, in: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Seattle, Washington, USA, 2013, pp. 1700–1709.
URL https://www.aclweb.org/anthology/D13-1176

[16] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks (2014). arXiv:1409.3215.

[17] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)doi:10.1109/cvpr.2016.90.
URL http://dx.doi.org/10.1109/CVPR.2016.90

[18] J. Ba, J. R. Kiros, G. E. Hinton, Layer normalization, ArXiv abs/1607.06450.

[19] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: G. Gordon, D. Dunson, M. Dudík (Eds.), Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Vol. 15 of Proceedings of Machine Learning Research, PMLR, Fort Lauderdale, FL, USA, 2011, pp. 315–323.
URL http://proceedings.mlr.press/v15/glorot11a.html

[20] L. Kaiser, A. N. Gomez, F. Chollet, Depthwise separable convolutions for neural machine translation (2017). arXiv:1706.03059.

[21] F. Wu, A. Fan, A. Baevski, Y. N. Dauphin, M. Auli, Pay less attention with lightweight and dynamic convolutions, arXiv preprint arXiv:1901.10430.

[22] M. Ott, S. Edunov, D. Grangier, M. Auli, Scaling neural machine translation, Proceedings of the Third Conference on Machine Translation: Research Papersdoi:10.18653/v1/w18-6301.
URL http://dx.doi.org/10.18653/v1/w18-6301

[23] P. Gage, A new algorithm for data compression, C Users J. 12 (2) (1994) 23–38.

[24] J. Gehring, M. Auli, D. Grangier, D. Yarats, Y. N. Dauphin, Convolutional sequence to sequence learning, in: Proceedings of the 34th International

Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 1243–1252.

[25] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, M. Auli, fairseq: A fast, extensible toolkit for sequence modeling, arXiv preprint arXiv:1904.01038.

[26] K. Papineni, S. Roukos, T. Ward, W.-J. Zhu, Bleu: a method for automatic evaluation of machine translation, in: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, 2002, pp. 311–318. `doi:10.3115/1073083.1073135`.
URL `https://www.aclweb.org/anthology/P02-1040`

[27] L. Wu, Y. Wang, Y. Xia, F. Tian, F. Gao, T. Qin, J. Lai, T.-Y. Liu, Depth growing for neural machine translation, Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics`doi:10.18653/v1/p19-1558`.
URL `http://dx.doi.org/10.18653/v1/p19-1558`

[28] B. Zhang, I. Titov, R. Sennrich, Improving deep transformer with depth-scaled initialization and merged attention (2019). `arXiv:1908.11365`.

[29] B. Yang, L. Wang, D. F. Wong, L. S. Chao, Z. Tu, Convolutional self-attention networks, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4040–4045. `doi:10.18653/v1/N19-1407`.
URL `https://www.aclweb.org/anthology/N19-1407`

[30] N. Kitaev, L. Kaiser, A. Levskaya, Reformer: The efficient transformer, in: International Conference on Learning Representations, 2020.
URL `https://openreview.net/forum?id=rkgNKkHtvB`

[31] P. Shaw, J. Uszkoreit, A. Vaswani, Self-attention with relative position representations, Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)`doi:10.18653/v1/n18-2074`.
URL `http://dx.doi.org/10.18653/v1/n18-2074`

[32] H. Zhang, Y. N. Dauphin, T. Ma, Fixup initialization: Residual learning without normalization, arXiv preprint arXiv:1901.09321.

[33] O. Press, N. A. Smith, O. Levy, Improving transformer models by reordering their sublayers, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, 2020, pp. 2996–3005. `doi:10.18653/v1/2020.acl-main.270`.
URL `https://www.aclweb.org/anthology/2020.acl-main.270`

[34] K. Ahmed, N. S. Keskar, R. Socher, Weighted transformer network for machine translation (2017). `arXiv:1711.02132`.

[35] J. Hao, X. Wang, S. Shi, J. Zhang, Z. Tu, Multi-granularity self-attention for neural machine translation, Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)`doi:10.18653/v1/d19-1082`.
URL `http://dx.doi.org/10.18653/v1/d19-1082`

[36] T. He, X. Tan, Y. Xia, D. He, T. Qin, Z. Chen, T.-Y. Liu, Layer-wise coordination between encoder and decoder for neural machine translation, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Advances in Neural Information Processing Systems 31, Curran Associates, Inc., 2018, pp. 7944–7954.

[37] D. R. So, C. Liang, Q. V. Le, The evolved transformer, arXiv preprint arXiv:1901.11117.

[38] P.-S. Huang, C. Wang, S. Huang, D. Zhou, L. Deng, Towards neural phrase-based machine translation, arXiv preprint arXiv:1706.05565.

[39] J. Lin, X. Sun, X. Ren, M. Li, Q. Su, Learning when to concentrate or divert attention: Self-adaptive attention temperature for neural machine translation, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 2985–2990.

[40] J. Feng, L. Kong, P.-S. Huang, C. Wang, D. Huang, J. Mao, K. Qiao, D. Zhou, Neural phrase-to-phrase machine translation (2018). `arXiv:1811.02172`.

[41] H. Cui, S. Iida, P.-H. Hung, T. Utsuro, M. Nagata, Mixed multi-head self-attention for neural machine translation, in: Proceedings of the 3rd Workshop on Neural Generation and Translation, Association for Computational Linguistics, Hong Kong, 2019, pp. 206–214. `doi:10.18653/v1/D19-5622`.
URL `https://www.aclweb.org/anthology/D19-5622`

[42] Z. Wu*, Z. Liu*, J. Lin, Y. Lin, S. Han, Lite transformer with long-short range attention, in: International Conference on Learning Representations, 2020.
URL `https://openreview.net/forum?id=ByeMPlHKPH`

[43] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, J. Shlens, Stand-alone self-attention in vision models (2019). `arXiv:1906.05909`.

[44] S. Ma, X. Sun, Y. Wang, J. Lin, Bag-of-words as target for neural machine translation, Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)`doi:10.18653/v1/p18-2053`.
URL `http://dx.doi.org/10.18653/v1/P18-2053`

[45] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, CoRR abs/1409.4842. `arXiv:1409.4842`.
URL `http://arxiv.org/abs/1409.4842`

[46] M. Tan, Q. V. Le, Mixconv: Mixed depthwise convolutional kernels (2019). `arXiv:1907.09595`.

[47] S. Xie, R. Girshick, P. Dollar, Z. Tu, K. He, Aggregated residual transformations for deep neural networks, 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)doi:10.1109/cvpr.2017.634.
URL http://dx.doi.org/10.1109/CVPR.2017.634

[48] J. Yan, F. Meng, J. Zhou, Multi-unit transformers for neural machine translation, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online, 2020, pp. 1047–1059. doi:10.18653/v1/2020.emnlp-main.77.
URL https://www.aclweb.org/anthology/2020.emnlp-main.77

[49] M. Xu, D. F. Wong, B. Yang, Y. Zhang, L. S. Chao, Leveraging local and global patterns for self-attention networks, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019, pp. 3069–3075.

[50] Q. Guo, X. Qiu, P. Liu, X. Xue, Z. Zhang, Multi-scale self-attention for text classification (2019). arXiv:1912.00544.

[51] S. Sukhbaatar, E. Grave, G. Lample, H. Jegou, A. Joulin, Augmenting self-attention with persistent memory (2019). arXiv:1907.01470.

[52] A. W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, Q. V. Le, Qanet: Combining local convolution with global self-attention for reading comprehension (2018). arXiv:1804.09541.

[53] B. Zoph, Q. V. Le, Neural architecture search with reinforcement learning (2016). arXiv:1611.01578.