Continuous Decomposition of Granularity for Neural Paraphrase Generation

Xiaodong Gu[1], **Zhaowei Zhang**[1], **Sang-Woo Lee**[2], **Kang Min Yoo**[2], **Jung-Woo Ha**[2]
[1] School of Software, Shanghai Jiao Tong University
[2] NAVER AI Lab
{xiaodong.gu,andy_zhangzw}@sjtu.edu.cn
{sang.woo.lee,kangmin.yoo,jungwoo.ha}@navercorp.com,

Abstract

While Transformers have had significant success in paragraph generation, they treat sentences as linear sequences of tokens and often neglect their hierarchical information. Prior work has shown that decomposing the levels of granularity (e.g., word, phrase, or sentence) for input tokens has produced substantial improvements, suggesting the possibility of enhancing Transformers via more fine-grained modeling of granularity. In this work, we propose continuous decomposition of granularity for neural paraphrase generation (C-DNPG). In order to efficiently incorporate granularity into sentence encoding, C-DNPG introduces a granularity-aware attention (GA-Attention) mechanism which extends the multi-head self-attention with: 1) a granularity head that automatically infers the hierarchical structure of a sentence by neurally estimating the granularity level of each input token; and 2) two novel attention masks, namely, *granularity resonance* and *granularity scope*, to efficiently encode granularity into attention. Experiments on two benchmarks, including Quora question pairs and Twitter URLs have shown that C-DNPG outperforms baseline models by a remarkable margin and achieves the state-of-the-art results in terms of many metrics. Qualitative analysis reveals that C-DNPG indeed captures fine-grained levels of granularity with effectiveness.

1 Introduction

With the continued success in NLP tasks (Vaswani et al., 2017), Transformer has been the mainstream neural architecture for paraphrase generation (Li et al., 2019; Kazemnejad et al., 2020; Guo et al., 2021; Hosking et al., 2022; Goyal and Durrett, 2020). The core element of Transformers is the self-attention network (SAN) (Vaswani et al., 2017) which computes sentence representations at each position by baking representations over all other positions in a parallel way. Despite their effectiveness, Transformer has been shown to be limited in structure modeling, that is, they process disperse words in a flat and uniform way without explicit modeling of the hierarchical structures (Raganato and Tiedemann, 2018; Hao et al., 2019; Li et al., 2020).

One potential route towards addressing this issues is multi-granularity text modeling (illustrated in Table 1) – which decomposes texts into multiple levels of granularity such as words, phrase and sentence (Li et al., 2019; Wiseman et al., 2018; Hao et al., 2019). For example, Li et al. (2019) attempts to achieve this via *decomposable neural paraphrase generator* (DNPG). DNPG decomposes paraphrase generation by Transformers into two levels of granularity: phrase-level (details) and sentence-level (templates). This allows a more flexible and controllable generating process. Paraphrases can be generated through rephrasing the templates while copying the detailed phrases. The decomposition is realized using a granularity separator, multiple encoder-decoder pairs, and an aggregator that summarizes results from all granularity levels.

Despite showing promising results, DNPG only captures a discrete (coarser-grained) decomposition of granularity, which restricts the capacity in representing more fine-grained semantic hierarchy. Furthermore, DNPG models the different levels of granularity using multiple encoders and decoders. That amounts to training multiple Transformers. The computational cost increases greatly as the number of granularity levels grows.

In this paper, we present C-DNPG (stands for *continuous decomposition of granularity for paraphrase generation*), a simple, fine-grained, and seamlessly integrated model for granularity-aware paragraph representation. C-DNPG extends the vanilla attention network in Transformer with a granularity head, which neurally estimates a continuous level of granularity for each token. In order to efficiently encode granularity into Transformers,

Text	What is the reason for World War II ?
decomposition 1	What is the reason for World War II ?
decomposition 2	What is the reason for World War II ?
decomposition 3	What is the reason for World War II ?
decomposition 4	What is the reason for World War II ?
decomposition 5	What is the reason for World War II ?

⇓

Levels of granularity (marked as superscripts):

What[1] is[1] the[2] reason[3] of[2] World[4] War[4] II[5] ?

Table 1: A motivation example of multi-granularity text decomposition. The given sentence can be decomposed according to 5 increasing levels of granularity, each corresponding to a partition of the sentence into a template (blue) at a specific level together with details (orange). Each row with colored text denotes a level of granularity where the blue words are in the sentence level (templates) and the remaining words are in the phrase level (details). The bottom half shows the level of granularity for each word according to the decomposition. We use integer numbers to indicate the extent of the granularity: the greater the number, the more detailed the word is.

C-DNPG adjusts the original self-attention weights using two novel attention masks: 1) a *granularity-resonance mask* which encourages attentions to exist between tokens with similar granularity; and 2) a *granularity scope mask* which encourages a small attention scope for lower-level (words or phrases) tokens. The granularity-aware attention mechanism provides continuous modeling sentence granularity and can be seamlessly integrated into the vanilla Transformer as the basic processing cell.

We evaluate the proposed C-DNPG on two commonly-used benchmarks, including the Quora question pairs and Twitter URL paraphrasing. Experimental results show that C-DNPG remarkably outperforms baseline models on both benchmarks and achieve the state-of-the-art results in many metrics. Qualitative study confirms the ability of the proposed approach in modeling fine-grained granularity.

Our contributions can be summarized as follows:

- We present a novel granularity-aware attention mechanism which provides a fine-grained decomposition of granularity for input tokens and hence yields a **continuous modeling of granularity** for natural language sentences.
- The proposed granularity-aware attention network can be **seamlessly integrated** into the Transformer for granularity-aware paraphrase generation.
- We conduct extensive evaluations of our methods on two popular paraphrase generation benchmarks and show that our approach remarkably outperforms previous works in terms of quantitative and qualitative results. We release all data and code at `https://github.com/guxd/C-DNPG`.

2 Background

Our approach is extended based on the Transformer and self-attention networks. We begin by introducing the background of these techniques.

2.1 Self-Attention Networks

The attention mechanism is a function which maps a query vector to a set of key-value vector pairs and summarizes an output vector as a weighted sum of the value vectors. The weight assigned to each value is computed using the query and key vectors. A typical attention function, for example, is the scaled dot-product attention (Vaswani et al., 2017):

$$\begin{aligned} &\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{A}^T\mathbf{V}, \\ &\mathbf{A} = \text{softmax}(\mathbf{Q}\mathbf{K}^T/\sqrt{d_k}) \end{aligned} \tag{1}$$

where $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V} \in \mathbb{R}^{L\times d}$ represent the query, key and value vectors, respectively. $\mathbf{A} \in [0,1]^{L\times L}$ is the attention score matrix with each $\mathbf{A}_{ij} = \mathbf{q}_i^T\mathbf{k}_j$; d_k denotes the dimension size of the key vector.

In particular, the *self-attention network* (SAN) is a special attention mechanism that computes the attention function over a single sequence. For a given sequence represented as a list of hidden states $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N] \in \mathbb{R}^{N\times d}$, the self-attention network computes representations of the sequence by relating hidden states at different positions (Vaswani

et al., 2017):

$$\begin{aligned} \mathbf{Q},\ \mathbf{K}, \mathbf{V} &= \mathbf{W}^{Q}\mathbf{H}, \mathbf{W}^{K}\mathbf{H}, \mathbf{W}^{V}\mathbf{H} \\ \text{SelfAttn}(\mathbf{H}) &= \text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) \end{aligned} \quad (2)$$

where $\mathbf{W}^{q}$, $\mathbf{W}^{k}$, and $\mathbf{W}^{v}$ are parameters to transform the input representation $\mathbf{H}$ to the query, key, and value respectively. Self-attention produces an abstraction and summary of a sequence in the hidden space and outputs the transformed hidden states (Vaswani et al., 2017).

2.2 Transformer

Transformer is an encoder-decoder model that is built upon the self-attention networks. It has been the common architectural choice for modeling paraphrase generation (Li et al., 2019; Guo et al., 2021; Kazemnejad et al., 2020). Transformer encodes a source sequence into hidden vectors and then generates a target sequence conditioned on the encoded vectors. Both the encoder and the decoder are composed of a stack of N identical layers, with each consists of a multi-head self-attention network followed by a position-wise fully connected network. Formally, the procedure of learning sequence representations through Transformer can be formulated as follows:

$$\begin{bmatrix} \bar{\mathbf{H}}^{l} = \text{LN}(\text{SelfAttn}(\mathbf{H}^{l-1}) + \mathbf{H}^{l-1}) \\ \mathbf{H}^{l} = \text{LN}(\text{FFN}^{l}(\bar{\mathbf{H}}^{l}) + \bar{\mathbf{H}}^{l}) \end{bmatrix}_{L} \quad (3)$$

where SelfAttn(.) denotes the multi-head self-attention network which performs the attention function over $\mathbf{H}^{l-1}$, the output hidden states of the l-1st layer; FFN and LN stand for the position-wise fully connected layer and the layer normalization, respectively; $[...]_{L}$ denotes the stack of L layers. The output of the final layer $\mathbf{H}^{L}$ is returned as the representation of the input sentence.

3 Approach

The vanilla Transformer processes disperse words in a flat and uniform way, which makes it difficult to represent words in terms of their syntactic guidance (Li et al., 2020). Prior work has shown that decomposing the levels of granularity (phrases or templates) has produced substantial gains in paraphrase generation (Li et al., 2019), suggesting the possibility of further improvement from finer-grained modeling of granularity (Hao et al., 2019).

Motivated by the benefit of explicitly denoting word granularity, we propose GA-attention, a new self-attention block which automatically decomposes fine-grained granularity and learns granularity-aware sentence representations. We integrate GA-attention into the vanilla Transformer to generate granularity-aware paraphrases.

3.1 Granularity-Aware Self-Attention

Unlike DNPG, which classifies each word into either phrase-level or sentence-level (Li et al., 2019), our method aims to give a soft classification of granularity for each word, yielding finer-grained decomposition of granularity for a sentence. For this purpose, we extend the vanilla self-attention with 1) a granularity head which estimates a continuous granularity level for each token, and 2) two new attention masks which bake the granularity into attentions for learning granularity-aware sentence representations. The overall architecture is illustrated in Figure 1a.

Granularity Head For a sequence of input tokens represented as hidden states $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N]$, the granularity head estimates a continuous granularity vector $\mathbf{z} = [z_1, \ldots, z_N] \in [0, 1]^N$, where $z_i \in [0, 1]$ measures the extent of token i belonging to details: a z_i that is close to 1 indicates that the token at position i tends to be a detailed word (i.e., in the phrase level), while a z_i approaching 0 indicates that token i tends to be a template word (i.e., in the sentence level). Specifically in the self-attention networks, the granularity for hidden states in layer l can be estimated as:

$$\mathbf{z}^{l} = \text{sigmoid}(\mathbf{W}^{G}\mathbf{H}^{l-1}), l = 2, ..., L \quad (4)$$

where $\mathbf{W}^{G}$ represents the training parameters; $\mathbf{H}^{l-1}$ denotes the hidden states of layer l-1.

Having estimated the granularity for input tokens, we want to effectively incorporate the granularity into attentions to control the learning of sentence representations. To this end, we propose two new attention masks, namely, the granularity resonance mask and the granularity scope mask, Our idea is to adjust the original attention weights using the two proposed masks.

Granularity Resonance Mask We first introduce the *granularity resonance mask* where "resonance" is analogy to an assumption of token correlations: sentence-level tokens attend more to sentence-level tokens, whereas phrase-level tokens attend more to phrase-level tokens (Li et al., 2019). In this sense, the term *granularity resonance* refers to the correlation between two tokens in terms of their levels of granularity.

Let z_i and z_j denote the granularity of tokens i and j respectively. In the binary case where $z_i \in \{0, 1\}$, their correlation in terms of granularity can be formulated as:

$$\mathbf{C}_{ij} = \begin{cases} 1, & \text{if } z_i = z_j \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where $\mathbf{C}_{ij}$ represents the regularization coefficient to the original attention weight $\mathbf{A}_{ij}$ in terms of granularity correlation. Such a discrete measure of resonance is limited in modeling token correlation as both z_i and $\mathbf{C}_{ij}$ are binary variables. To improve the capacity of the granularity-aware attention, we generalize the computation of $\mathbf{C}_{ij}$ (Equation 5) to a continuous function, that is,

$$\begin{aligned} \mathbf{C}_{ij} =& (1 - z_i) \times \max(0, 1 - (z_i + z_j)) \\ &+ z_i \times \min(1, 1 - z_i + z_j) \end{aligned} \tag{6}$$

where $z_i \in [0, 1]$ is a continuous value; z_i and 1-z_i control the extent of token i being in the word level or the sentence level, respectively. Equation 6 provides a smooth measure of the correlation between token i and j.

Granularity Scope Mask We further define the *granularity scope mask* where *granularity scope* measures the scope of attention according to the granularity level. This is based on the local attention assumption of phrases (Li et al., 2019): a phrase-level token tends to attend to surrounding tokens while a sentence-level token can attend to other tokens evenly with any distance. In that sense, phrasal tokens (with a large z_i) have a relatively smaller attention scope compared to sentence-level words (with a small z_i). For a given sequence of hidden states $\mathbf{h}_1, \ldots, \mathbf{h}_N$ with N words, the granularity scope for position i attending to position j can be defined as:

$$\mathbf{S}_{ij} = \begin{cases} 1 & \text{if } |i - j| < (N - \epsilon)^{(1-z_i)} + \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where $\mathbf{S}_{ij} \in \{0, 1\}$ denotes the penalty for the original attention weight $\mathbf{A}_{ij}$ in terms of granularity scope. ϵ denotes the maximum distance that a phrasal word attends to. We set ϵ to 2 according to a similar configuration in (Li et al., 2019). This equation can be intuitively interpreted as the receptive fields for different levels of granularity (Li et al., 2019): a phrase-level token i (z_i=1) attends only to the adjacent n (n=3) words, whereas a sentence-level token i (z_i=0) can attend to positions with any distances.

Similarly, we generalize Equation 7 to a continuous function:

$$\mathbf{S}_{ij} = \max(0, \min(1, (N - \epsilon)^{(1-z_i)} + \epsilon - |i - j|)) \tag{8}$$

Using the two granularity-based attention masks, we derive the granularity-aware self-attention as an adjustment of the original attention weights, namely,

$$\begin{aligned} &\text{GASelfAttn}(\mathbf{H}) = \tilde{\mathbf{A}}^T \mathbf{V} \\ &\tilde{\mathbf{A}} = \mathbf{A} \odot \mathbf{C} \odot \mathbf{S} \end{aligned} \tag{9}$$

where $\mathbf{A}$ and $\tilde{\mathbf{A}}$ denote the original and adjusted attention weights respectively; $\odot$ stands for the element-wise multiplication; GASelfAttn(.) represents the proposed granularity-aware self-attention function.

3.2 C-DNPG: Transformer with Granularity Aware Attention

Based on the proposed GA-Attention, we propose C-DNPG, which integrates GA-attention into Transformer in order to better generate paraphrases at fine-grained levels of granularity. Figure 1b illustrates the overall architecture of our model. Compared to the vanilla Transformer, our model simply replaces the self-attention layers in both the encoder and the decoder with the proposed GA-attention network. Similar to the vanilla Transformer, we perform the GA-attention function for multiple heads in parallel and concatenate the multi-head representations to yield the final representation. The procedure for the C-DNPG (Transformer with GA-Attention) can be summarized as:

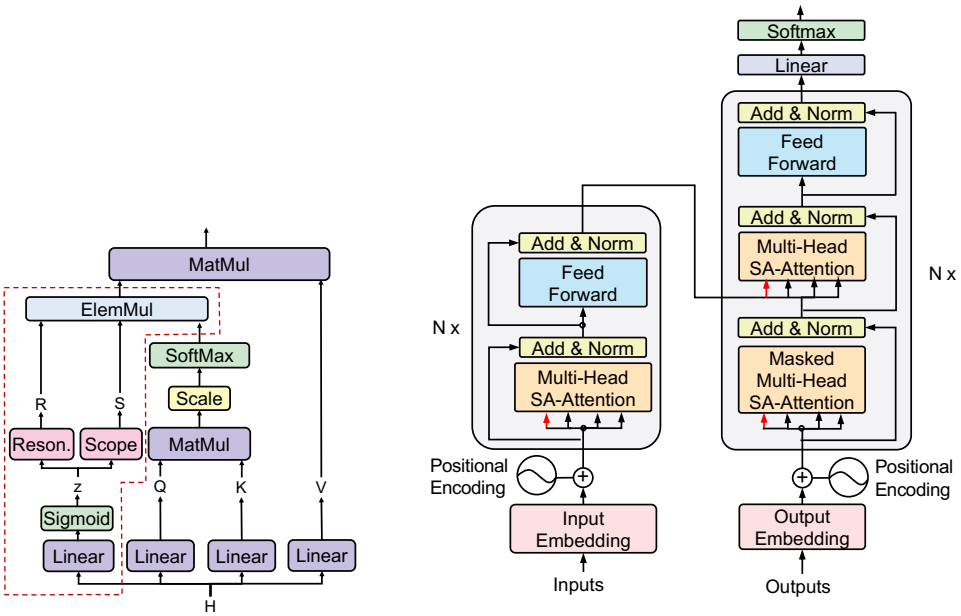
$$\left[\begin{aligned} \bar{\mathbf{H}}^l &= \text{LN}(\text{GASelfAttn}(\mathbf{H}^{l-1}) + \mathbf{H}^{l-1}) \\ \mathbf{H}^l &= \text{LN}(\text{FFN}^l(\bar{\mathbf{H}}^l) + \bar{\mathbf{H}}^l) \end{aligned}\right]_L \tag{10}$$

where LN denotes layer normalization (Ba et al., 2016).

4 Experiments

We evaluate our approach by experimenting on two widely used datasets including the Quora question pairs and the Twitter URLs. We will introduce the common experimental setup and the empirical results.

Implementation Details We implemented our approach on top of the Huggingface PyTorch Transformer (Wolf et al., 2019). For a fair comparison,



(a) Architecture of the GA-attention network. The red dotted area represents the extensions against the original attention network.

(b) Architecture of C-DNPG which seamlessly replaces the multi-head attention with our multi-head GA-attention. Red arrows denote additional z flows in GA-attention.

Figure 1: The architectures of GA-Attention and C-DNPG.

we followed the hyperparameter settings in related works (Li et al., 2019) for the Transformer. Both encoder and decoder consist of 3 transformer layers, have a hidden size of 450, and contain 9 attention heads (L=3, H=450, A=9). Following previous work (Li et al., 2019; Kazemnejad et al., 2020), we truncate sentences to 20 tokens. We utilize the pre-trained tokenizer by huggingface[1] (i.e., bert-base-uncased) for tokenization which has been common in NLP. During decoding, we employ beam search with a beam size of 8. All models were optimized with AdamW (Loshchilov and Hutter, 2018). The learning rate was varied under a linear schedule with warmup steps of 5,000 and the maximum learning rate of $5e^{-5}$. The model was training for 100,000 batches until achieving the best validation loss. The experiments were repeated for 5 times and were reported with their average results. All models were trained on a machine with NVIDIA Tesla V100 GPU allocated with a batch size of 32 samples.

Baseline Models We compare our approach with popular paraphrase generation methods including: (i) RedidualLSTM (Prakash et al., 2016): an LSTM sequence-to-sequence model using residuals between RNN layers; (ii) PointerGenerator (See et al., 2017): an RNN sequence-to-sequence model using copy mechanism; (iii) Transformer (Vaswani et al., 2017): the vanilla Transformer model; (iv) Transformer+Copy: an enhanced Transformer with copy mechanism (Gu et al., 2016); and (v) DNPG (Li et al., 2019): a popular paraphrase generation model based on Transformer. DNPG extends Transformer by generating paraphrases at multiple levels of granularity such as the phrase level and the sentence level. The model is composed of two encoder-decoder pairs, which correspond to phrase-level and sentence-level paraphrasing, respectively. We use the default settings of the baseline models as reported in their papers. (vi) FSET (Kazemnejad et al., 2020): the state-of-the-art paraphrase generation model that retrieves a paraphrase pair similar to the input sentence from a pre-defined index, then editing it using the extracted relations between the retrieved pair of sentences. We directly report the performance from their original paper.

Evaluation Metrics We perform automatic evaluation using five widely used metrics for text generation tasks, namely, BLEU (Papineni et al., 2002), iBLEU (Sun and Zhou, 2012), ROUGE-L (Lin, 2004) and METEOR (Lavie and Agarwal, 2007). We compute both BLEU-2 and BLEU-4 scores

[1] https://github.com/huggingface/tokenizers

in our experiments using the NLTK package[2]. iBLEU (Sun and Zhou, 2012) penalizes BLEU by n-gram similarity between output and input. Hence, it is taken as the main metric for paraphrasing.

Datasets We conducted the experiments on two widely used benchmarks: 1) the Quora question pairs benchmark[3], which contains 124K duplicate question pairs. The dataset was labeled by human annotators and has been widely used for paraphrase research (Li et al., 2019; Devlin et al., 2019). We split the original data into train, validation, and test sets with proportions of 100K, 4K, and 20K, respectively.

2) the Twitter URL paraphrasing dataset[4] is also a widely used benchmark for evaluating paraphrase generation (Li et al., 2018; Kazemnejad et al., 2020). The dataset contains two subsets which are manually and automatically labeled, respectively. Following (Li et al., 2018), we sample 110k instances from the automatically labeled subset as our training set and sample 5k and 1k instances from the manually annotated subset for the test and validation sets, respectively.

5 Results and Analysis

5.1 Automatic Evaluation

Table 2 shows the results of various approaches on the two benchmarks. As the results indicate, C-DNPG (with variants) achieves the best performance in terms of most automatic metrics, which suggests that our C-DNPG is effective in performing multi-granularity paraphrasing.

In particular, our approach outperforms DNPG, a multi-granularity Transformer based model with a significant margin. That means that by modeling more fine-grained granularity levels, C-DNPG can control the generating process more precisely. Thanks to the granularity attention mechanisms, it is more flexible for the model to leverage syntactic guidance (e.g., recognizing templates and details) for paraphrase generation.

It is interesting to note that either the resonance mask or the scope mask that we propose can achieve the best performance under specific settings. We hypothesize that there could be overlap between the two proposed attention masks in some cases. Therefore, combining them may amplify the extent of masking and hinder the ultimate performance. We also find that Transformer with a copy mechanism can outperform DNPG on the Twitter dataset. This might be due to the more noise in this dataset, which leads copy based models to be more effective.

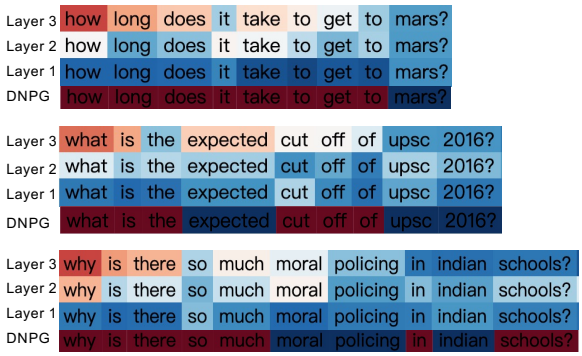


Figure 2: Examples of multi-granularity extracted by C-DNPG (Layer1-3) and DNPG (bottom) on the Quora dataset. Warmer colors represent higher levels of granularity (templates) while colder colors represent lower levels of granularity (details). We present granularity of all Transformer layers and compare the results with those of DNPG.

5.2 Qualitative Analysis

To gain a more in-depth insight into the performance, we qualitatively analyze the interpretability of GA-attention. We visualize the output of the granularity head in each attention layer to verify how effectively GA-attention captures fine-grained granularity. As shown in Figure 2, GA-attention can successfully capture continuous linguistic structures reflected as multiple levels of granularity (in the last layer). For example, it successfully yields four levels of templates: 1) *what* __, 2) *what is* __, and 3) *what is the expected* __ 4) *what is the expected cut off of* __ according to Example 1 in Figure 2. In contrast, DNPG can decompose only two levels of granularity for each sentence. This means that C-DNPG can successfully distinguish templates and detailed words for each sentence, thus generating more fine-grained paraphrases. Another interesting observation is that the continuous granularity is not extracted at once, instead, it is gradually summarized through transformer layers. This indicates that the proposed granularity-aware extensions blend naturally with attention networks in learning *coarse-to-fine* representations (Jawahar et al., 2019).

Overall, the results suggest that the proposed GA-attention naturally extends vanilla attention

[2]https://www.nltk.org/_modules/nltk/translate/bleu_score.html
[3]https://www.kaggle.com/c/quora-question-pairs (NC)
[4]https://github.com/lanwuwei/Twitter-URL-Corpus

Model	Quora					Twitter URL				
	iBLEU	BLEU-2	BLEU-4	ROUGE-L	METEOR	iBLEU	BLEU-2	BLEU-4	ROUGE-L	METEOR
ResidualLSTM	20.45	40.71	26.20	36.19	32.67	20.29	36.75	25.92	32.47	29.44
Pointer-generator	22.65	43.82	28.80	42.36	40.87	25.60	44.50	32.40	38.48	36.48
Transformer	21.14	37.97	26.88	40.14	38.21	24.44	44.45	31.12	31.97	32.49
Transformer+Copy	22.90	44.42	28.94	37.60	38.34	27.07	48.44	34.35	38.37	38.19
DNPG	24.55	47.72	31.01	42.37	42.12	25.92	46.36	32.91	36.77	36.28
FSET	-	**51.03**	33.46	-	38.57	-	46.35	34.62	-	31.67
C-DNPG (R)	**26.94**	47.58	**34.05**	46.17	44.75	27.96	49.98	35.80	38.67	39.39
C-DNPG (S)	26.68	47.48	33.93	**46.22**	**46.66**	28.19	49.10	35.95	38.89	39.06
C-DNPG (R$\odot$S)	25.96	46.25	33.02	44.64	44.25	**30.25**	49.00	**38.58**	**41.60**	**41.71**
C-DNPG (R+S)	26.66	50.96	33.69	44.45	43.33	28.73	**50.49**	36.61	39.80	40.42

Table 2: Results of paraphrase generation on two benchmarks. R stands for the resonance mask while S stands for the scope mask; R$\times$S stands for the combination of two masks through element-wise multiplication; R+S means we average the two masks for adjusting the original attention weights. We note that the results of baseline models are stronger than those reported in the DNPG paper, probably due to the BERT tokenizer we have utilized in our experiments. The pointer-generator outperforms the vanilla Transformer, as is consistent to the DNPG paper.

networks and enhances text representations with fine-grained granularity modeling.

5.3 Case Study

Table 3 presents two sample paraphrases generated by different models in the Quora test set. As the samples indicate, C-DNPG generates more coherent and fluent paraphrases than other models, which is consistent with the results of the automatic and human evaluation. According to the first sample, C-DNPG produces a more relevant and human-like paraphrase. For example, C-DNPG successfully paraphrases the word "first" as "*beginners*". The second sample shows more clear strength of C-DNPG which generates a paraphrase that is even better than the ground-truth question asked by human (e.g., the year "2100" is mistakenly paraphrased as "2099").

5.4 Human Evaluation

Besides the automatic evaluation, we also perform a human study to assess the performance of our approach qualitatively. We compare our approach with two typical methods, namely DNPG (Li et al., 2019) and the vanilla Transformer (Vaswani et al., 2017). They represent the state-of-the-art decomposition based method and the backbone model that our model is built upon, respectively. We randomly selected 200 Quora questions from the test set. For each one of the questions, one paraphrase was generated for each model. Then, three annotators from the Amazon Mechanical Turk were asked to compare the generated paraphrases by two models (ours vs. a baseline model) blindly based on two criteria, relevance and fluency. Relevance means that the generated paraphrases are semantically equivalent to the original question. Fluency means the generated paraphrases are natural and fluent sentences. Table 4 presents the comparison results. As can be seen, our model significantly outperforms the other two methods in terms of the two criteria. Moreover, the Fleiss' kappa κ shows fair agreement between annotators.

5.5 Computational Efficiency

As one of the key advantages of C-DNPG, we finally evaluate the time efficiency of our approach. We used the same setup as described in the Setup section. As Table 5 shows, the granularity aware attention mechanism in C-DNPG does not bring much additional computational cost to Transformers as opposed to the DNPG baseline approach, which indicates that GA-attention is lightweight to be integrated into Transformers.

6 Discussion

6.1 Why baking the two masks can enhance the performance?

Our idea is a generalization of the previous work DNPG (Li et al., 2019). In that paper, decomposing sentences into templates and details can improve the performance of paraphrasing because paraphrasing is usually generated by rewriting the sentence template while directly copying the detailed words. In that sense, template words pay more attention to template words while detailed words tend to pay attention to detailed words. Our paper generalizes this idea by extending the binary level (either 0 or 1) of granularity to a continuous range (between 0 and 1) of levels. This was implemented by the two proposed masks.

Sentence:	What is a good first programming language?
Transformer:	What is good?
DNPG:	What is good for coding?
C-DNPG:	What are the best programming languages for beginners?
Human:	Whats a good and easy programming language to learn?
Sentence:	What will the year 2100 be like?
Transformer:	What is likely to happen in the world?
DNPG:	What are did today. year - year of unique year of country?
C-DNPG:	What will the world look like in 2100?
Human:	What will the year 2099 be like?

Table 3: Sample paraphrases from multiple models with human reference.

Comparison	Relevance				Fluency			
	Win	**Tie**	**Loss**	Kappa	**Win**	**Tie**	**Loss**	Kappa
Ours *vs.* Transformer	67.8%	14.3%	17.8%	0.156	67.5%	15.8%	16.7%	0.166
Ours *vs.* DNPG	72.3%	12.8%	14.8%	0.171	68.1%	17.5%	14.3%	0.194
Ours *vs.* Human	43.7%	10.5%	45.8%	0.079	43.2%	11.0%	45.8%	0.095

Table 4: Human evaluation on the test set of Quora.

Model	Time (hours)
Transformer	1.2
TransformerCopy	1.2
DNPG	3.6
C-DNPG (ours)	1.5

Table 5: Training time (until model convergence) of various approaches on the Quora dataset.

7 Related Work

This work is closely related to (1) multi-granularity paraphrase generation and (2) multi-granularity attention.

Multi-Granularity Paraphrase Generation.

There has been an increasing interest in decomposing paraphrase generation into multiple levels of granularity such as word, phrase and sentence (Li et al., 2019; Wiseman et al., 2018; Hao et al., 2019). For example, (Li et al., 2019) present the decomposable neural paraphrase generator (DNPG). DNPG is a Transformer-based model that generates paraphrases at two levels of granularity in a disentangled way. The model is composed of two encoder-decoder pairs, corresponding to phrase-level and sentence-level, respectively. The difference between our C-DNPG and DNPG is of threefold: 1) C-DNPG estimates a continuous granularity level of each token and hence supports a continuous modeling of hierarchical structures; 2) Compared to DNPG, the GA-attention in C-DNPG can be naturally and seamlessly integrated into Transformer and is light-weighted in computation; and 3) While DNPG predicts granularity using a single fully connected layer, C-DNPG gradually summarizes granularity through a stack of Transformer layers.

Multi-Granularity Attention.

Another important line of work relates to multi-granularity self-attention. (Hao et al., 2019) proposed multi-granularity self-attention (MG-SA): MG-SA combines multi-head self-attention and phrase modeling by trains attention heads to attend to phrases in either n-gram or syntactic formalism. (Nguyen et al., 2020) proposed a tree-structured attention network which encodes parse tree structures into self-attention at constant time complexity. Despite similar names, our method differs from theirs greatly in principle and architecture. These two works rely on the existence of parsing trees, as opposed to GA-attention which infers latent structures from plain text. Furthermore, MG-SA only considers two-levels of granularity while GA-attention aims at continuous modeling of multiple granularity. (Liu et al., 2020) proposed a hybrid neural architecture named MahNN which integrates RNN and ConvNet, each learning a different aspect of

semantic from the linguistic structures. Like other related works, MahNN is based on a coarse-grained attention mechanism. The two levels of granularity it processes are represented by RNN and ConvNet, respectively. By contrast, GA-attention provides a fine-grained attention function extended from the vanilla attention mechanism. Therefore, GA-attention is a pure attention-based approach and can be naturally and seamlessly integrated into Transformers.

8 Conclusion

In this paper, we have proposed a novel paraphrase generation model named C-DNPG for continuously decomposing sentences at different levels of granularity. C-DNPG extends the multi-head attention with a granularity head which neurally estimates continuous granularity level of each input token. To efficiently incorporate granularity into attentions, we propose two novel attention masks, namely, granularity-resonance mask and granularity-scope mask, to adjust the original attention weights. Results on two paraphrase generation benchmarks show that C-DNPG remarkably outperforms baseline models in both quantitative and qualitative studies. In future work, we will investigate the effect of C-DNPG in pre-trained models and other NLP tasks.

Acknowledgments

We would like to thank the anonymous reviewers for their constructive feedback. Xiaodong Gu was sponsored by NSFC No. 62102244, CCF-Tencent Open Research Fund (RAGR20220129), and CCF-Baidu Open Fund (NO.2021PP15002000).

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *ACL*, pages 1631–1640.

Zilu Guo, Zhongqiang Huang, Kenny Q. Zhu, Guandan Chen, Kaibo Zhang, Boxing Chen, and Fei Huang. 2021. Automatically paraphrasing via sentence reconstruction and round-trip translation. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3815–3821. International Joint Conferences on Artificial Intelligence Organization.

Jie Hao, Xing Wang, Shuming Shi, Jinfeng Zhang, and Zhaopeng Tu. 2019. Multi-granularity self-attention for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 887–897.

Tom Hosking, Hao Tang, and Mirella Lapata. 2022. Hierarchical sketch induction for paraphrase generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022)*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Amirhossein Kazemnejad, Mohammadreza Salehi, and Mahdieh Soleymani Baghshah. 2020. Paraphrase generation by learning how to edit from samples. In *proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6010–6021.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, page 228–231, USA. Association for Computational Linguistics.

Yinghao Li, Rui Feng, Isaac Rehg, and Chao Zhang. 2020. Transformer-based neural text generation with syntactic guidance. *arXiv preprint arXiv:2010.01737*.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2018. Paraphrase generation with deep reinforcement learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3865–3878, Brussels, Belgium. Association for Computational Linguistics.

Zichao Li, Xin Jiang, Lifeng Shang, and Qun Liu. 2019. Decomposable neural paraphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3403–3414.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Zhenyu Liu, Chaohong Lu, Haiwei Huang, Shengfei Lyu, and Zhenchao Tao. 2020. Text classification based on multi-granularity attention hybrid neural network. *arXiv preprint arXiv:2008.05282*.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

Xuan-Phi Nguyen, Shafiq Joty, Steven Hoi, and Richard Socher. 2020. Tree-structured attention with hierarchical accumulation. In *International Conference on Learning Representations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Aaditya Prakash, Sadid A Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016. Neural paraphrase generation with stacked residual lstm networks. *arXiv preprint arXiv:1610.03098*.

Alessandro Raganato and Jörg Tiedemann. 2018. An analysis of encoder representations in transformer-based machine translation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 287–297.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Hong Sun and Ming Zhou. 2012. Joint learning of a dual smt system for paraphrase generation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 38–42.

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

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2018. Learning neural templates for text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3174–3187, Brussels, Belgium. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.