How Does Selective Mechanism Improve Self-Attention Networks?

Xinwei Geng[1*] **Longyue Wang**[2] **Xing Wang**[2] **Bing Qin**[1,3] **Ting Liu**[1,3] **Zhaopeng Tu**[2]
[1]Harbin Institute of Technology [2]Tencent AI Lab [3]Peng Cheng Laboratory
[1]{xwgeng, qinb, tliu}@ir.hit.edu.cn
[2]{vinnylywang, brightxwang, zptu}@tencent.com

Abstract

Self-attention networks (SANs) with selective mechanism has produced substantial improvements in various NLP tasks by concentrating on a subset of input words. However, the underlying reasons for their strong performance have not been well explained. In this paper, we bridge the gap by assessing the strengths of selective SANs (SSANs), which are implemented with a flexible and universal Gumbel-Softmax. Experimental results on several representative NLP tasks, including natural language inference, semantic role labelling, and machine translation, show that SSANs consistently outperform the standard SANs. Through well-designed probing experiments, we empirically validate that the improvement of SSANs can be attributed in part to mitigating two commonly-cited weaknesses of SANs: *word order encoding* and *structure modeling*. Specifically, the selective mechanism improves SANs by paying more attention to content words that contribute to the meaning of the sentence. The code and data are released at https://github.com/xwgeng/SSAN.

1 Introduction

Self-attention networks (SANs) (Lin et al., 2017) have achieved promising progress in various natural language processing (NLP) tasks, including machine translation (Vaswani et al., 2017), natural language inference (Shen et al., 2018b), semantic role labeling (Tan et al., 2018; Strubell et al., 2018) and language representation (Devlin et al., 2019). The appealing strength of SANs derives from high parallelism as well as flexibility in modeling dependencies among all the input elements.

Recently, there has been a growing interest in integrating *selective mechanism* into SANs, which has produced substantial improvements in a variety of NLP tasks. For example, some researchers incorporated a hard constraint into SANs to select a subset of input words, on top of which self-attention is conducted (Shen et al., 2018c; Hou et al., 2019; Yang et al., 2019b). Yang et al. (2018) and Guo et al. (2019) proposed a soft mechanism by imposing a learned Gaussian bias over the original attention distribution to enhance its ability of capturing local contexts. Shen et al. (2018c) incorporated reinforced sampling to dynamically choose a subset of input elements, which are fed to SANs.

Although the general idea of selective mechanism works well across NLP tasks, previous studies only validate their own implementations in a few tasks, either on only classification tasks (Shen et al., 2018c; Guo et al., 2019) or sequence generation tasks (Yang et al., 2018, 2019b). This poses a potential threat to the conclusive effectiveness of selective mechanism. In response to this problem, we adopt a flexible and universal implementation of selective mechanism using Gumbel-Softmax (Jang et al., 2017), called selective self-attention networks (*i.e.,* SSANs). Experimental results on several representative types of NLP tasks, including natural language inference (i.e., *classification*), semantic role labeling (i.e., *sequence labeling*), and machine translation (i.e., *sequence generation*), demonstrate that SSANs consistently outperform the standard SANs (§3).

Despite demonstrating the effectiveness of SSANs, the underlying reasons for their strong performance have not been well explained, which poses great challenges for further refinement. In this study, we bridge this gap by assessing the strengths of selective mechanism on capturing essentially linguistic properties via well-designed experiments. The starting point for our approach is recent findings: the standard SANs suffer from two representation limitation on modeling *word order encoding* (Shaw et al., 2018; Yang et al., 2019a)

[*] Work done when interning at Tencent AI Lab.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2986–2995
July 5 - 10, 2020. ©2020 Association for Computational Linguistics

and *syntactic structure modeling* (Tang et al., 2018; Hao et al., 2019a), which are essential for natural language understanding and generation. Experimental results on targeted linguistic evaluation lead to the following observations:

- SSANs can identify the improper word orders in both local (§4.1) and global (§4.2) ranges by learning to attend to the expected words.
- SSANs produce more syntactic representations (§5.1) with a better modeling of structure by selective attention (§5.2).
- The selective mechanism improves SANs by paying more attention to content words that posses semantic content and contribute to the meaning of the sentence (§5.3).

2 Methodology

2.1 Self-Attention Networks

SANs (Lin et al., 2017), as a variant of attention model (Bahdanau et al., 2015; Luong et al., 2015), compute attention weights between each pair of elements in a single sequence. Given the input layer $\mathbf{H} = \{\mathbf{h}_1, \cdots, \mathbf{h}_N\} \in \mathbb{R}^{N\times d}$, SANs first transform the layer $\mathbf{H}$ into the queries $\mathbf{Q} \in \mathbb{R}^{N\times d}$, the keys $\mathbf{K} \in \mathbb{R}^{N\times d}$, and the values $\mathbf{V} \in \mathbb{R}^{N\times d}$ with three separate weight matrices. The output layer $\mathbf{O}$ is calculated as:

$$\mathbf{O} = \mathrm{ATT}(\mathbf{Q}, \mathbf{K})\mathbf{V} \tag{1}$$

where the alternatives to $\mathrm{ATT}(\cdot)$ can be additive attention (Bahdanau et al., 2015) or dot-product attention (Luong et al., 2015). Due to time and space efficiency, we used the dot-product attention in this study, which is computed as:

$$\mathrm{ATT}(\mathbf{Q}, \mathbf{K}) = softmax(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}) \tag{2}$$

where $\sqrt{d}$ is the scaling factor with d being the dimensionality of layer states (Vaswani et al., 2017).

2.2 Weaknesses of Self-Attention Networks

Despite SANs have demonstrated its effectiveness on various NLP tasks, recent studies empirically revealed that SANs suffer from two representation limitations of modeling *word order encoding* (Yang et al., 2019a) and *syntactic structure modeling* (Tang et al., 2018). In this work, we concentrate on these two commonly-cited issues.

Word Order Encoding SANs merely rely on attention mechanism with neither recurrence nor convolution structures. In order to incorporate sequence order information, Vaswani et al. (2017) proposed to inject position information into the input word embedding with additional position embedding. Nevertheless, SANs are still weak at learning word order information (Yang et al., 2019a). Recent studies have shown that incorporating recurrence (Chen et al., 2018; Hao et al., 2019b,c), convolution (Song et al., 2018; Yang et al., 2019b), or advanced position encoding (Shaw et al., 2018; Wang et al., 2019a) into vanilla SANs can further boost their performance, confirming its shortcomings at modeling sequence order.

Structure Modeling Due to lack of supervision signals of learning structural information, recent studies pay widespread attention on incorporating syntactic structure into SANs. For instance, Strubell et al. (2018) utilized one attention head to learn to attend to syntactic parents of each word. Towards generating better sentence representations, several researchers propose phrase-level SANs by performing self-attention across words inside a n-gram phrase or syntactic constituent (Wu et al., 2018; Hao et al., 2019a; Wang et al., 2019b). These studies show that the introduction of syntactic information can achieve further improvement over SANs, demonstrating its potential weakness on structure modeling.

2.3 *Selective* Self-Attention Networks

In this study, we implement the selective mechanism on SANs by introducing an additional *selector*, namely SSANs, as illustrated in Figure 1. The selector aims to select a subset of elements from the input sequence, on top of which the standard self-attention (Equation 1) is conducted. We implement the selector with Gumbel-Softmax, which has proven effective for computer vision tasks (Shen et al., 2018a; Yang et al., 2019c).

Selector Formally, we parameterize selection action $a \in \{\mathrm{SELECT}, \mathrm{DISCARD}\}$ for each input element with an auxiliary policy network, where SELECT indicates that the element is selected for self-attention while DISCARD represents to abandon the element. The output action sequence $\mathbf{A} \in \mathbb{R}^N$ is calculated as:

$$\pi(\mathbf{A}) = sigmoid(\mathbf{E}_s) \tag{3}$$

$$\mathbf{E}_s = \mathbf{Q}_s\mathbf{K}_s^T \tag{4}$$

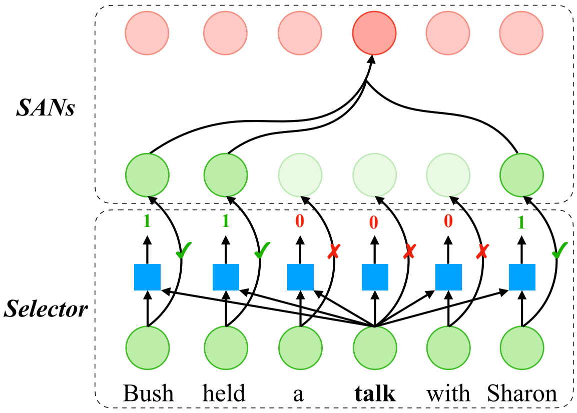


Figure 1: Illustration of SSANs that select a subset of input elements with an additional selector network, on top of which self-attention is conducted. In this example, the word "*talk*" performs attention operation over input sequence, where the words "*Bush*", "*held*" and "*Sharon*" are chosen as the truly-significant words.

where $\mathbf{Q}_s \in \mathbb{R}^{N\times d}$ and $\mathbf{K}_s \in \mathbb{R}^{N\times d}$ are transformed from the input layer $\mathbf{H}$ with distinct weight matrices. We utilize $sigmoid$ as activation function to calculate the distribution for choosing the action SELECT with the probability π or DISCARD with the probability $1-\pi$.

Gumbel Relaxation There are two challenges for training the selector: (1) the ground-truth labels indicating which words should be selected are unavailable; and (2) the discrete variables in $\mathbf{A}$ lead to a non-differentiable objective function. In response to this problem, Jang et al. (2017) proposed Gumbel-Softmax to give a continuous approximation to sampling from the categorical distribution. We adopt a similar approach by adding Gumbel noise (Gumbel, 1954) in the sigmoid function, which we refer as *Gumbel-Sigmoid*. Since sigmoid can be viewed as a special 2-class case ($\mathbf{E}_s$ and $\mathbf{0}$ in our case) of softmax, we derive the *Gumbel-Sigmoid* as:

$$\begin{aligned} &\textit{Gumbel-Sigmoid}(\mathbf{E}_s)\\ &\quad= \textit{sigmoid}((\mathbf{E}_s+\mathbf{G}'-\mathbf{G}'')/\tau)\\ &\quad= \frac{\exp((\mathbf{E}_s+\mathbf{G}')/\tau)}{\exp((\mathbf{E}_s+\mathbf{G}')/\tau)+\exp(\mathbf{G}''/\tau)} \end{aligned} \qquad (5)$$

where $\mathbf{G}'$ and $\mathbf{G}''$ are two independent Gumbel noises (Gumbel, 1954), and $\tau \in (0,\infty)$ is a temperature parameter. As τ diminishes to zero, a sample from the *Gumbel-Sigmoid* distribution becomes cold and resembles the one-hot samples. At training time, we can use *Gumbel-Sigmoid* to obtain differentiable sample $\mathbf{A}$ as *Gumbel-Sigmoid*$(\mathbf{E}_s)$. In inference, we choose the action with maximum probability as the final output.

3 NLP Benchmarks

To demonstrate the robustness and effectiveness of the SSANs, we evaluate it in three representative NLP tasks: language inference, semantic role labeling and machine translation. We used them as NLP benchmarks, which cover classification, sequence labeling and sequence generation categories. Specifically, the performances of semantic role labeling and language inference models heavily rely on structural information (Strubell et al., 2018), while machine translation models need to learn word order and syntactic structure (Chen et al., 2018; Hao et al., 2019c).

3.1 Experimental Setup

Natural Language Inference aims to classify semantic relationship between a pair of sentences, *i.e.,* a premise and corresponding hypothesis. We conduct experiments on the Stanford Natural Language Inference (SNLI) dataset (Bowman et al., 2015), which has three classes: Entailment, Contradiction and Neutral.

We followed Shen et al. (2018b) to use a token2token SAN layer followed by a source2token SAN layer to generate a compressed vector representation of input sentence. The selector is integrated into the token2token SAN layer. Taking the premise representation s^p and the hypothesis vector s^h as input, their semantic relationship is represented by the concatenation of s^p, s^h, $s^p - s^h$ and $s^p \cdot s^h$, which is passed to a classification module to generate a categorical distribution over the three classes. We initialize the word embeddings with 300D GloVe 6B pre-trained vectors (Pennington et al., 2014), and the hidden size is set as 300.

Semantic Role Labeling is a shallow semantic parsing task, which aims to recognize the predicate-argument structure of a sentence, such as "who did what to whom", "when" and "where". Typically, it assigns labels to words that indicate their semantic role in the sentence. Our experiments are conducted on CoNLL2012 dataset provided by Pradhan et al. (2013).

We evaluated selective mechanism on top of DEEPATT[1] (Tan et al., 2018), which consists of

[1] https://github.com/XMUNLP/Tagger.

stacked SAN layers and a following softmax layer. Following their configurations, we set the number of SAN layers as 10 with hidden size being 200, the number of attention heads as 8 and the dimension of word embeddings as 100. We use the GloVe embeddings (Pennington et al., 2014), which are pretrained on Wikipedia and Gigaword, to initialize our networks, but they are not fixed during training. We choose the better feed-forward networks (FFN) variants of DEEPATT as our standard settings.

Machine Translation is a conditional generation task, which aims to translate a sentence from a source language to its counterpart in a target language. We carry out experiments on several widely-used datasets, including small English⇒Japanese (En⇒Ja) and English⇒Romanian (En⇒Ro) corpora, as well as a relatively large English⇒German (En⇒De) corpus. For En⇒De and En⇒Ro, we respectively follow Li et al. (2018) and He et al. (2018) to prepare WMT2014[2] and IWSLT2014[3] corpora. For En⇒Ja, we use KFTT[4] dataset provided by Neubig (2011). All the data are tokenized and then segmented into subword symbols using BPE (Sennrich et al., 2016) with 32K operations.

We implemented the approach on top of advanced TRANSFORMER model (Vaswani et al., 2017). On the large-scale En⇒De dataset, we followed the base configurations to train the NMT model, which consists of 6 stacked encoder and decoder layers with the layer size being 512 and the number of attention heads being 8. On the small-scale En⇒Ro and En⇒Ja datasets, we followed He et al. (2018) to decrease the layer size to 256 and the number of attention heads to 4.

For all the tasks, we applied the selector to the first layer of encoder to better capture lexical and syntactic information, which is empirically validated by our further analyses in Section 4.

3.2 Experimental Results

Table 1 shows the results on the three NLP benchmarks. Clearly, introducing selective mechanism significantly and consistently improves performances in all tasks, demonstrating the universality and effectiveness of the selective mechanism for SANs. Concretely, SSANs relatively improve prediction accuracy over SANs by +0.8% and +0.5% respectively on the NLI and SRL tasks, showing their superiority on structure modeling. Shen et al. (2018c) pointed that SSANs can better capture dependencies among semantically important words, and our results and further analyses (§5) provide supports for this claim.

Task	**Size**	**SANs**	**SSANs**	△
Natural Language Inference (Accuracy)				
SNLI	550K	85.60	86.30	*+0.8%*
Semantic Role Labeling (F1 score)				
CoNLL	312K	82.48	82.88	*+0.5%*
Machine Translation (BLEU)				
En⇒Ro	0.18M	23.22	23.91	*+3.0%*
En⇒Ja	0.44M	31.56	32.17	*+1.9%*
En⇒De	4.56M	27.60	28.50	*+3.3%*

Table 1: Results on the NLP benchmarks. "Size" indicates the number of training examples, and "△" denotes relative improvements over the vanilla SANs.

In the machine translation tasks, SSANs consistently outperform SANs across language pairs. Encouragingly, the improvement on translation performance can be maintained on the large-scale training data. The relative improvements on the En⇒Ro, En⇒Ja, and En⇒De tasks are respectively +3.0%, +1.9%, and +3.3%. We attribute the improvement to the strengths of SSANs on word order encoding and structure modeling, which are empirically validated in Sections 4 and 5.

Shen et al. (2018c) implemented the selection mechanism with the REINFORCE algorithm. Jang et al. (2017) revealed that compared with Gumbel-Softmax (Maddison et al., 2014), REINFORCE (Williams, 1992) suffers from high variance, which consequently leads to slow converge. In our preliminary experiments, we also implemented REINFORCE-based SSANs, but it underperforms the Gumbel-Softmax approach on the benchmarking En⇒De translation task (BLEU: 27.90 vs. 28.50, not shown in the paper). The conclusion is consistent with Jang et al. (2017), and we thus use Gumbel-Softmax instead of REINFORCE in this study.

4 Evaluation of Word Order Encoding

In this section, we investigate the ability of SSANs of capturing both *local* and *global* word orders on the *bigram order shift detection* (§4.1) and *word reordering detection* (§4.2) tasks.

[2] http://www.statmt.org/wmt14.

[3] https://wit3.fbk.eu/mt.php?release=2014-01.

[4] http://www.phontron.com/kftt.

Model	Layer	Acc.	△
SANs	–	52.23	–
SSANs	1	**62.55**	*+19.8%*
	2	53.73	*+2.9%*
	3	54.65	*+4.6%*
	4	54.29	*+3.9%*
	5	54.78	*+4.9%*
	6	54.23	*+3.8%*

Table 2: Results on the *local* bigram order shift detection task when *SSANs* are applied into different layers.

4.1 Detection of Local Word Reordering

Task Description Conneau et al. (2018) propose a *bigram order shift detection* task to test whether an encoder is sensitive to local word orders. Given a monolingual corpus, a certain portion of sentences are randomly extracted to construct instances with illegal word order. Specially, given a sentence $X = \{x_1, \ldots, x_N\}$, two adjacent words (*i.e.*, x_n, x_{n+1}) are swapped to generate an illegal instance X' as a substitute for X. Given processed data which consists of intact and inverted sentences, examined models are required to distinguish intact sentences from inverted ones. To detect the shift of bigram word order, the models should learn to recognize normal and abnormal word orders.

The model consists of 6-layer SANs and 3-layer MLP classifier. The layer size is 128, and the filter size is 512. We trained the model on the open-source dataset[5] provided by Conneau et al. (2018). The accuracy of SAN-based encoder is higher than previously reported result on the same task (Li et al., 2019) (52.23 vs. 49.30).

Detection Accuracy Table 2 lists the results on the local bigram order shift detection task, in which SSANs are applied to different encoder layers. Clearly, all the SSANs variants consistently outperform SANs, demonstrating the superiority of SSANs on capturing local order information. Applying the selective mechanism to the first layer achieves the best performance, which improves the prediction accuracy by +19.8% over SANs. The performance gap between the SSANs variants is very large (i.e., 19.8% vs. around 4%), which we attribute to that the detection of local word reorder depends more on lexical information embedded in the bottom layer.

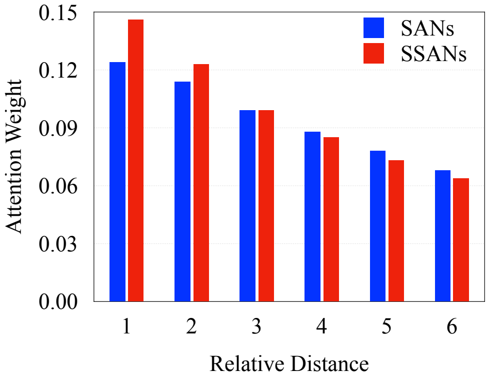
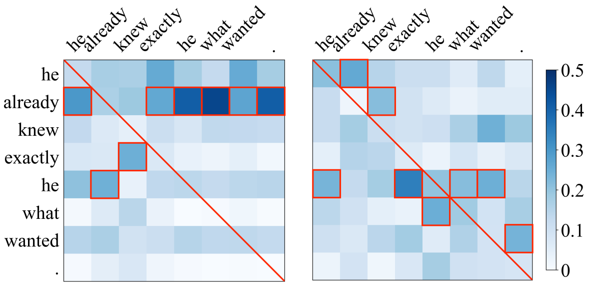


Figure 2: Attention weights over attended words with different relative distance from the query word on the *local* reordering task. *SSANs pay more attention to the adjacent words (distance=1) than SANs.*



(a) SANs (b) SSANs

Figure 3: Visualization of attention weights from an example on the *local* reordering detection task. We highlight the attended word (Y-axis) with maximum attention weight for each query (X-axis) in red rectangles.

Attention Behaviors The objective of local reordering task is to distinguish the swap of two adjacent words, which requires the examined model to pay more attention to the adjacent words. Starting from this intuition, we investigate the attention distribution over the attended words with different relative distances from the query word, as illustrated in Figure 2. We find that both SANs and SSANs focus on neighbouring words (e.g., distance < 3), and SSANs pays more attention to the adjacent words (distance=1) than SANs (14.6% vs. 12.4%). The results confirm our hypothesis that the selective mechanism helps to exploit more bigram patterns to accomplish the task objective. Figure 3 shows an example, in which SSANs attend most to the adjacent words except the inverted bigram "*he what*". In addition, the surrounding words "*exactly*" and "*wanted*" also pay more attention to the exceptional word "*he*". We believe such features help to distinguish the abnormally local word order.

[5] https://github.com/facebookresearch/SentEval/tree/master/data/probing.

Model	Layer	Insert	Original	Both
SANs	–	73.20	66.00	60.10
SSANs	1	**81.52**	**72.19**	**66.77**
	2	80.14	70.01	63.97
	3	79.82	69.69	63.93
	4	79.08	70.22	63.67
	5	80.19	69.84	64.12
	6	80.27	69.50	63.73

Table 3: Performance on the *global* word reordering detection (WRD) task.

4.2 Detection of Global Word Reordering

Task Description Yang et al. (2019a) propose a *word reordering detection* task to investigate the ability of SAN-based encoder to extract global word order information. Given a sentence $X = \{x_1, \ldots, x_N\}$, a random word x_i is popped and inserted into another position j ($i \neq j$). The objective is to detect both the original position the word is popped out (labeled as "O"), and the position the word is inserted (labeled as "I").

The model consists of 6-layer SANs and a output layer. The layer size is 512, and the filter size is 2048. We trained the model on the open-source dataset[6] provided by Yang et al. (2019a).

Detection Accuracy Table 3 lists the results on the global reordering detection task, in which all the SSANs variants improve prediction accuracy. Similarly, applying the selective mechanism to the first layer achieves the best performance, which is consistent with the results on the global word reordering task (Table 2). However, the performance gap between the SSANs variants is much lower that that on the local reordering task (i.e., 4% vs. 15%). One possible reason is that the detection of global word reordering may also need syntactic and semantic information, which are generally embedded in the high-level layers (Peters et al., 2018).

Attention Behaviors The objective of the WRD is to distinguish a global reordering (averaged distance is 8.7 words), which requires the examined model to pay more attention to distant words. Figure 4 depicts the attention distribution according to different relative distances. SSANs alleviate the leaning-to-local nature of SANs and pay more attention to distant words (e.g., distance> 5), which better accomplish the task of detecting global reordering. Figure 5 illustrates an example, in which

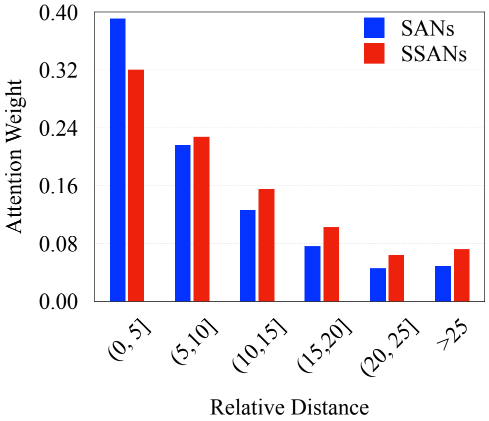
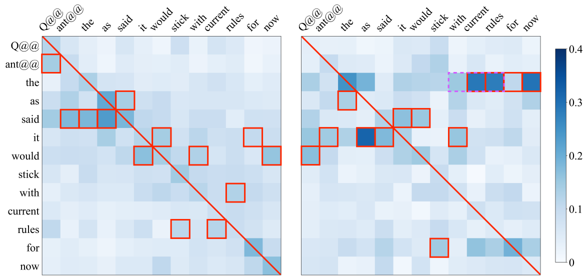


Figure 4: Attention weights over attended words with different relative distance from the query word on the *global* WRD task. *SSANs pay more attention to the distant words (distance> 5) than SANs.*



(a) SANs (b) SSANs

Figure 5: Visualization of attention weights from an example on the *global* reordering detection task. We highlight the attended word (Y-axis) with maximum attention weight for each query (X-axis) in red rectangles.

more queries in SSANs attend most to the inserted word "*the*" than SANs. Particularly, SANs pay more attention to the surrounding words (e.g., distance < 3), while the inserted word "*the*" only accepts subtle attention. In contrast, SSANs dispense much attention over words centred on the inserted position (*i.e.,* "*the*") regardless of distance, especially for the queries "*current rules for now*". We speculate that SSANs benefits from such features on detecting the global word reordering .

5 Evaluation of Structure Modeling

In this section, we investigate whether SSANs better capture structural information of sentences. To this end, we first empirically evaluate the syntactic structure knowledge embedded in the *learned representations* (§5.1). Then we investigate the *attention behaviors* by extracting constituency tree from the attention distribution (§5.2).

[6] `https://github.com/baosongyang/WRD`.

Class	Ratio	SANs	SSANs	△
5	6.9%	68.66	75.22	*+9.6%*
6	14.3%	56.10	64.09	*+14.2%*
7	16.3%	46.63	55.05	*+18.1%*
8	17.9%	39.68	50.88	*+28.2%*
9	17.4%	38.33	50.97	*+33.0%*
10	15.3%	35.54	49.88	*+40.3%*
11	11.9%	48.86	56.39	*+15.4%*
All	100%	45.68	55.90	*+22.4%*

Table 4: F1 score on the tree depth task. "Ratio" denotes the portion each class takes.

Type	Ratio	SANs	SSANs	△
Ques.	10%	95.90	97.06	+1.2%
Decl.	60%	88.48	91.34	+3.2%
Clau.	25%	72.78	78.32	+7.6%
Other	5%	50.67	61.13	+20.6%
All	100%	83.78	87.25	*+4.1%*

Table 5: F1 score on the top constituent task. We report detailed results on 4 types of sentences: question ("Ques."), declarative ("Decl."), a clause ("Clau."),nd other ("Other") sentences.

5.1 Structures Embedded in Representations

Task Description We leverage two linguistic probing tasks to assessing the syntactic information embedded in a given representation. Both tasks are cast as multi-label classification problem based on the representation of a given sentence, which is produced by an examined model:

Tree Depth (**TreeDepth**) task (Conneau et al., 2018) checks whether the examined model can group sentences by the depth of the longest path from root to any leaf in their parsing tree. Tree depth values range from 5 to 11, and the task is to categorize sentences into the class corresponding to their depth (7 classes).

Top Constituent (**TopConst**) task (Shi et al., 2016) classifies the sentence in terms of the sequence of top constituents immediately below the root node, such as "ADVP NP VP .". The top constituent sequences fall into 20 categories: 19 classes for the most frequent top constructions, and one for all other constructions.

We trained the model on the open-source dataset provided by Conneau et al. (2018), and used the same model architecture in Section 4.1.

Probing Accuracy Table 4 lists the results on the TreeDepth task. SSANs significantly outperform SANs by 22.4% on the overall performance. Concretely, the performance of SANs dramatically drops as the depth of the sentences increases.[7] On the other hand, SSANs is more robust to the depth of the sentences, demonstrating the superiority of SSANs on capturing complex structures.

Metric	SANs	SSANs	△
BP	21.09	22.07	*+4.7%*
BR	22.05	23.07	*+4.6%*
F1	21.56	22.56	*+4.2%*

Table 6: Evaluation on constituency trees generated from the attention distribution.

Table 5 shows the results on the TopConst task. We categorize the 20 classes into 4 categories based on the types of sentences: question sentence ("* SQ ."), declarative sentence ("* NP VP *" etc.), clause sentence ("SBAR *" and "S *"), and others ("OTHER"). Similarly, the performance of SANs drops as the complexity of sentence patterns increases (e.g., "Ques." ⇒ "Others", 95.90 ⇒ 50.67). SSANs significantly improves the prediction F1 score as the complexity of sentences increases, which reconfirm the superiority of SSANs on capturing complex structures.

5.2 Structures Modeled by Attention

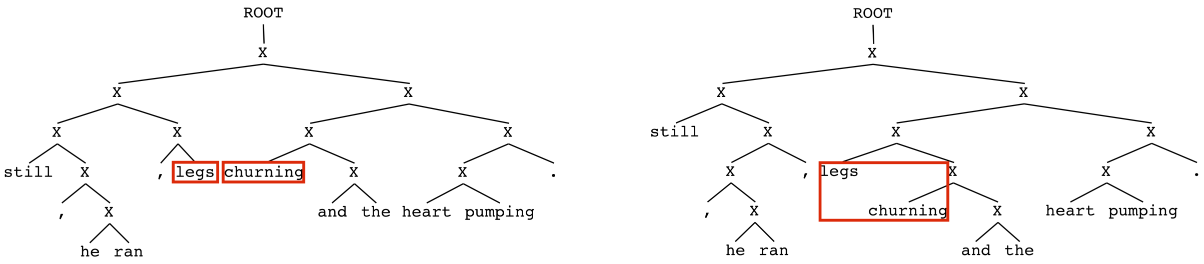
Task Description We evaluate the ability of self-attention on structure modeling by constructing constituency trees from the attention distributions. Under the assumption that attention distribution within phrases is stronger than the other, Mareček and Rosa (2018) define the score of a constituent with span from position i to position j as the attention merely inside the span denoted as score(i, j). Based on these scores, a binary constituency tree is generated by recurrently splitting the sentence. When splitting a phrase with span (i, j), the target is to look for a position k maximizing the scores of the two resulting phrases:

$$k = \arg\max_{k'}(\text{score}(i, k') \cdot \text{score}(k', j)) \qquad (6)$$

We utilized Stanford CoreNLP toolkit to annotate English sentences as golden constituency trees. We used EVALB[8] to evaluate the generated constituency trees, including bracketing precision, bracketing recall, and bracketing F1 score.

[7] The only exception is the class of "11", which we attribute to the extraction of feature of associating "very complex sentence" with maximum depth "11".

[8] http://nlp.cs.nyu.edu/evalb.



(a) SANs

(b) SSANs

Figure 6: Example of constituency trees generated from the attention distributions.

Type		TreeDepth			TopConst			En⇒De Translation		
		SANs	*SSANs*	△	*SANs*	*SSANs*	△	*SANs*	*SSANs*	△
Content	Noun	0.149	0.245	*+64.4%*	0.126	0.196	*+55.6%*	0.418	0.689	*+64.8%*
	Verb	0.165	0.190	*+15.2%*	0.165	0.201	*+21.8%*	0.146	0.126	*-13.7%*
	Adj.	0.040	0.069	*+7.3%*	0.033	0.054	*+63.6%*	0.077	0.074	*-3.9%*
	Total	0.354	0.504	*+42.4%*	0.324	0.451	*+39.2%*	0.641	0.889	*+38.7%*
Content-Free	Prep.	0.135	0.082	*-39.3%*	0.123	0.119	*-3.3%*	0.089	0.032	*-64.0%*
	Dete.	0.180	0.122	*-32.2%*	0.103	0.073	*-29.1%*	0.070	0.010	*-85.7%*
	Punc.	0.073	0.068	*-6.8%*	0.078	0.072	*-7.7%*	0.098	0.013	*-86.7%*
	Others	0.258	0.224	*-13.2%*	0.373	0.286	*-23.3%*	0.102	0.057	*-41.1%*
	Total	0.646	0.496	*-23.3%*	0.676	0.549	*-18.8%*	0.359	0.111	*-69.1%*

Table 7: Attention distributions on linguistic roles for the structure modeling probing tasks (§5.1, "TreeDepth" and "TopConst") and the constituency tree generation task (§5.2, "En⇒De Translation").

Parsing Accuracy As shown in Table 6, SSANs consistently outperform SANs by 4.6% in all the metrics, demonstrating that SSANs better model structures than SANs. Figure 6 shows an example of generated trees. As seen, the phrases "*he ran*" and "*heart pumping*" can be well composed for both SANs and SSANS. However, SANs fail to parse the phrase structure "*legs churning*", which is correctly parsed by SSANs.

5.3 Analysis on Linguistic Properties

In this section, we follow He et al. (2019) to analyze the linguistic characteristics of the attended words in the above structure modeling tasks, as listed in Table 7. Larger relative increase ("△") denotes more attention assigned by SSANs. Clearly, *SSANs pay more attention to content words in all cases*, although there are considerable differences among NLP tasks.

Content words possess semantic content and contribute to the meaning of the sentence, which are essential in various NLP tasks. For example, the depth of constituency trees mainly relies on the nouns, while the modifiers (e.g., adjective and content-free words) generally make less contributions. The top constituents mainly consist of VP (95% examples) and NP (75% examples) categories, whose head words are verbs and nouns respectively. In machine translation, content words carry essential information, which should be fully transformed to the target side for producing adequate translations. Without explicit annotations, SANs are able to learn the required linguistic features, especially on the machine translation task (e.g., dominating attention on nouns). SSANs further enhance the strength by paying more attention to the content words.

However, due to their high frequency with a limited vocabulary (e.g., 150 words[9]), content-free words, or function words generally receive a lot of attention, although they have very little substantive meaning. This is more series in structure probing tasks (i.e., TreeDepth and TopConst), since the scalar guiding signal (i.e., class labels) for a whole sentence is non-informative as it does not necessarily preserve the picture about the intermediate syntactic structure of the sentence that is being

[9] https://en.wikipedia.org/wiki/Function_word.

generated for the prediction. On the other hand, the problem on content-free words is alleviated on machine translation tasks due to the informative sequence signals. SSANs can further alleviate this problem in all cases with a better modeling of structures.

6 Conclusion

In this work, we make an early attempt to assess the strengths of the selective mechanism for SANs, which is implemented with a flexible Gumbel-Softmax approach. Through several well-designed experiments, we empirically reveal that the selective mechanism migrates two major weaknesses of SANs, namely *word order encoding* and *structure modeling*, which are essential for natural language understanding and generation. Future directions include validating our findings on other SAN architectures (e.g., BERT (Devlin et al., 2019)) and more general attention models (Bahdanau et al., 2015; Luong et al., 2015).

Acknowledgments

We thank the anonymous reviewers for their insightful comments. We also thank Xiaocheng Feng, Heng Gong, Zhangyin Feng, and Xiachong Feng for helpful discussion. This work was supported by the National Key R&D Program of China via grant 2018YFB1005103 and National Natural Science Foundation of China (NSFC) via grant 61632011 and 61772156.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *ACL*.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What You Can Cram Into A Single Vector: Probing Sentence Embeddings for Linguistic Properties. In *ACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Xiaocheng Feng, Ming Liu, Jiahao Liu, Bing Qin, Yibo Sun, and Ting Liu. 2018. Topic-to-essay generation with neural networks. In *IJCAI*.

Xiaocheng Feng, Duyu Tang, Bing Qin, and Ting Liu. 2016. English-chinese knowledge base translation with neural network. In *COLING*.

Emil Julius Gumbel. 1954. *Statistical theory of extreme values and some practical applications: a series of lectures*. U. S. Govt. Print. Office.

Maosheng Guo, Yu Zhang, and Ting Liu. 2019. Gaussian transformer: a lightweight approach for natural language inference. In *AAAI*.

Jie Hao, Xing Wang, Shuming Shi, Jinfeng Zhang, and Zhaopeng Tu. 2019a. Multi-granularity self-attention for neural machine translation. In *EMNLP*.

Jie Hao, Xing Wang, Shuming Shi, Jinfeng Zhang, and Zhaopeng Tu. 2019b. Towards better modeling hierarchical structure for self-attention with ordered neurons. In *EMNLP*.

Jie Hao, Xing Wang, Baosong Yang, Longyue Wang, Jinfeng Zhang, and Zhaopeng Tu. 2019c. Modeling recurrence for transformer. In *NAACL*.

Shilin He, Zhaopeng Tu, Xing Wang, Longyue Wang, Michael Lyu, and Shuming Shi. 2019. Towards understanding neural machine translation with word importance. In *EMNLP*.

Tianyu He, Xu Tan, Yingce Xia, Di He, Tao Qin, Zhibo Chen, and Tie-Yan Liu. 2018. Layer-wise coordination between encoder and decoder for neural machine translation. In *NIPS*.

Xiaochen Hou, Jing Huang, Guangtao Wang, Kevin Huang, Xiaodong He, and Bowen Zhou. 2019. Selective attention based graph convolutional networks for aspect-level sentiment classification. *ArXiv*, abs/1910.10857.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *ICLR*.

Jian Li, Zhaopeng Tu, Baosong Yang, Michael R. Lyu, and Tong Zhang. 2018. Multi-Head Attention with Disagreement Regularization. In *EMNLP*.

Jian Li, Baosong Yang, Zi-Yi Dou, Xing Wang, Michael R. Lyu, and Zhaopeng Tu. 2019. Information aggregation for multi-head attention with routing-by-agreement. In *NAACL*.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *ICLR*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *EMNLP*.

Chris J Maddison, Daniel Tarlow, and Tom Minka. 2014. A* sampling. In *NIPS*.

David Mareček and Rudolf Rosa. 2018. Extracting syntactic trees from transformer encoder self-attentions. In *BlackboxNLP@EMNLP*.

Graham Neubig. 2011. The Kyoto free translation task. http://www.phontron.com/kftt.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *NAACL*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using ontonotes. In *CoNLL*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *ACL*.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-Attention with Relative Position Representations. In *NAACL*.

Chen Shen, Guo-Jun Qi, Rongxin Jiang, Zhongming Jin, Hongwei Yong, Yaowu Chen, and Xian-Sheng Hua. 2018a. Sharp attention network via adaptive sampling for person re-identification. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(10):3016–3027.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018b. DiSAN: Directional Self-Attention Network for RNN/CNN-Free Language Understanding. In *AAAI*.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Sen Wang, and Chengqi Zhang. 2018c. Reinforced self-attention network: a hybrid of hard and soft attention for sequence modeling. In *IJCAI*.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does String-based Neural MT Learn Source Syntax? In *EMNLP*.

Kaitao Song, Xu Tan, Di He, Jianfeng Lu, Tao Qin, and Tie-Yan Liu. 2018. Double path networks for sequence to sequence learning. In *COLING*.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *EMNLP*.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2018. Deep semantic role labeling with self-attention. In *AAAI*.

Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. 2018. Why Self-Attention? A Targeted Evaluation of Neural Machine Translation Architectures. In *EMNLP*.

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 *NIPS*.

Xing Wang, Zhaopeng Tu, Longyue Wang, and Shuming Shi. 2019a. Self-attention with structural position representations. In *EMNLP*.

Yaushian Wang, Hung-Yi Lee, and Yun-Nung Chen. 2019b. Tree transformer: Integrating tree structures into self-attention. In *EMNLP*.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, pages 229–256.

Wei Wu, Houfeng Wang, Tianyu Liu, and Shuming Ma. 2018. Phrase-level self-attention networks for universal sentence encoding. In *EMNLP*.

Baosong Yang, Zhaopeng Tu, Derek F. Wong, Fandong Meng, Lidia S. Chao, and Tong Zhang. 2018. Modeling localness for self-attention networks. In *EMNLP*.

Baosong Yang, Longyue Wang, Derek F. Wong, Lidia S. Chao, and Zhaopeng Tu. 2019a. Assessing the ability of self-attention networks to learn word order. In *ACL*.

Baosong Yang, Longyue Wang, Derek F. Wong, Lidia S. Chao, and Zhaopeng Tu. 2019b. Convolutional self-attention networks. In *NAACL*.

Jiancheng Yang, Qiang Zhang, Bingbing Ni, Linguo Li, Jinxian Liu, Mengdie Zhou, and Qi Tian. 2019c. Modeling point clouds with self-attention and gumbel subset sampling. In *CVPR*.