# Neural Sentence Simplification with Semantic Dependency Information

**Zhe Lin, Xiaojun Wan**
Wangxuan Institute of Computer Technology, Peking University
Center for Data Science, Peking University
The MOE Key Laboratory of Computational Linguistics, Peking University
{linzhe, wanxiaojun}@pku.edu.cn

### Abstract

Most previous works on neural sentence simplification exploit seq2seq model to rewrite a sentence without explicitly considering the semantic information of the sentence. This may lead to the semantic deviation of the simplified sentence. In this paper, we leverage semantic dependency graph to aid neural sentence simplification system. We propose a new sentence simplification model with semantic dependency information, called SDISS (as shorthand for **S**emantic **D**ependency **I**nformation guided **S**entence **S**implification), which incorporates semantic dependency graph to guide sentence simplification. We evaluate SDISS on three benchmark datasets and it outperforms a number of strong baseline models on the SARI and FKGL metrics. Human evaluation also shows SDISS can produce simplified sentences with better quality.

## Introduction

Sentence simplification aims to reduce the linguistic complexity of a sentence, while still preserving its salient information and meaning. Sentence simplification has many practical applications. For instance, it can provide assistance for low-literacy reader (Watanabe et al. 2009) or for patients with linguistic and cognitive disabilities (Carroll et al. 1999). In addition, a simplification component could be used to improve the performance of tasks such as parsing(Chandrasekar, Doran, and Srinivas 1996), summarization(Klebanov, Knight, and Marcu 2004) and so on.

Inspired by the success of machine translation (MT), many text simplification (TS) systems treat sentence simplification as a monolingual translation task. Most current TS systems based on neural machine translation employ seq2seq models to transform a source sentence to a simplified target sentence. The common practice for Seq2seq models is to use recurrent neural networks (RNNs) with Long Short-Term Memory (Hochreiter and Schmidhuber 1997) or Transformer (Vaswani et al. 2017). However, there still exists the problem of semantic irrelevance and deviation from the source sentence in many case, which will reduce the quality of simplified sentences. An example of sentence simplification is shown in Table 1. The Transformer model without using semantic information tends to copy the whole source sentence and cannot catch the theme of the original sentence.

Many researches in other tasks have got positive effect by introducing semantic information into neural models. For instance, Song et al. (Song, Zhao, and Liu 2018) combined source syntactic structures into neural sentence summarization to help the model identify summary-worthy content and avoid content deviation. Song et al. (2019) incorporated abstract meaning representation into machine translation models. But few attempts have been made for the task of sentence simplification yet.

In this study, we aim to investigate the use of semantic information in neural text simplification systems and we focus on the semantic dependency graph of the source sentence. Semantic dependency graph contains predicate-argument relations between content words in a sentence. There are various semantic dependency representation schemes based on different annotation systems (Oepen et al. 2016). We leverage DM (DELPH-IN MRS Bi-Lexical Dependencies) graph as an additional input to TS system because of its high consistency and accuracys. In DM graph, nodes represent words in the sentence and edges represent semantic relationships between words. Non-content words, such as punctuation, are left out of the analysis.

Our model consists of three parts: sentence encoder, graph encoder and sentence decoder. We leverage Transformer (Vaswani et al. 2017) encoder as sentence encoder. We propose a new graph encoder by leveraging graph attention (Velickovic et al. 2018), which encodes bi-directional graphs separately. sentence decoder is based on Transformer decoder to aggregate sentence information and graph information. We evaluate our model on three benchmark datasets and it has made a significant improvement over the SARI and FKGL metrics on all the datasets. The results of human evaluation also show our model can produce simplified sentences with better quality. We empirically show that semantic information can significantly improve the performance of TS systems. As shown in Table 1, our proposed model (i.e., SDISS) can produce a simplified sentence with better fluency and semantic relevance.

The contributions of our work are summarized as below:

1) To the best of our knowledge, we are the first to explore semantic dependency information for neural sentence simplification and introduce semantic dependency graph into

| | |
|---|---|
| Source | To prevent overfishing , the agreement would , among other things , make it much easier to establish marine protected areas -LRB- MPAs -RRB- in the high seas . |
| Reference | An agreement would make it easier to create marine protected areas . |
| Transformer | To prevent lionfish , the agreement would among other things , make much easier to protected areas -LRB- MPAs -RRB- in the high seas . |
| SDISS (ours) | The agreement would make it much easier to establish marine protected areas . |

Table 1: An example for sentence simplification using different models.

neural TS system.

2) We propose a new model to extract semantic information from semantic dependency graph by splitting the graph into forward and reverse ones according to the edge's direction and encode them with graph attention network separately. Our code is publicly available at https://github.com/L-Zhe/SDISS.

3) Automatic and human evaluations demonstrate that our model can simplify sentences better and achieve new state-of-the-art performances on three benchmark datasets.

## Related Work

There are two major categories of models for sentence simplification: statistical machine translation (SMT) based models and neural machine learning (NMT) based models. SMT based models, including tree-based MT (TBMT) (Zhu, Bernhard, and Gurevych 2010; Woodsend and Lapata 2011), phrase-based MT (PBMT) (Coster and Kauchak 2011; Wubben, van den Bosch, and Krahmer 2012; Štajner, Béchara, and Saggion 2015) and syntax-based MT (SBMT) (Xu et al. 2016), applied traditional statistical models to sentence simplification.

With the success of NMT, researchers started to develop sentence simplification models based on NMT. Nisioi et al. (2017) implemented a neural sentence simplification model by using Long Short-Term Memory (LSTM), and got better performance than PBMT. Zhang and Lapata (2017) proposed to train encoder-decoder model with deep reinforcement learning and the reward has three components to capture key aspects of the target output: simplicity, relevance, and fluency. Vu et al. (2018) employed a pointer-generator network with neural semantic encoder. Dong et al. (2019) came up with EditNTS, leveraged neural programmer-interpreter to produce a series of edit operations to operate on the original sentence. Recently, Zhao et al. (2020) proposed BTTS and BTTSRL based on unsupervised and semi-supervised learning methods. Martin et al. (2020) leveraged multilingual unsupervised method to train TS system in languages except English.

There are also some models to introduce semantic information to simplification tasks (Narayan and Gardent 2016; Sulem, Abend, and Rappoport 2018b), but all of them were based on traditional methods. Differently, we introduce semantic dependency graph into neural simplification models. Graph structure can be encoded by leveraging graph neural network (GNN) (Scarselli et al. 2008). Kipf and Welling (2017) proposed graph convolutional network (GCN) to introduce convolutional operation on graph neural network. Velickovic et al. (2018) combined GCN with attention mechanism to put forward graph attention network (GAT) and made significant improvement in many tasks. Inspired by GAT, we leverage an attention mechanism to aggregate the neighbor relationships within semantic dependency graphs.

## Our SDISS Model

In this section, we introduce the components of our model in detail. First, we define the sentence simplification task and introduce the notations. Then, we describe the sentence encoder, graph encoder and sentence decoder, respectively. Finally, we introduce the loss with copy penalty to prevent over-replication and length token for length control. Figure 1 shows the overview of our model.

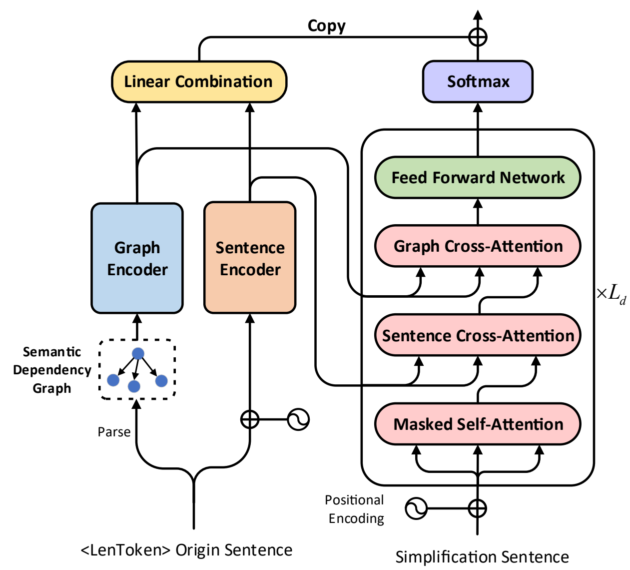


Figure 1: An overview of SDISS, which consists of sentence encoder, graph encoder and sentence decoder.

### Notations

Our model regards simplification as monolingual translation task. Given an input sentence $X = (x_1, x_2, \cdots, x_N)$, where $N$ is the sentence length. The corresponding simplified sentence is $Y = (y_1, y_2, \cdots, y_M)$, where $M$ is the length of simplified sentence. In addition, we parse $X$ into its semantic dependency graph $G = (V, E)$ by using neural factorization-based SDP parser (Chen, Ye, and Sun 2019), where $V$ denotes the set of nodes in the graph,

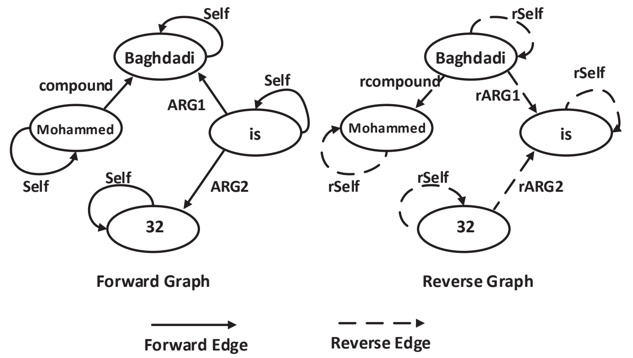


Figure 2: Example forward and reverse graphs for the sentence 'Mohammed Baghdadi is 32.'

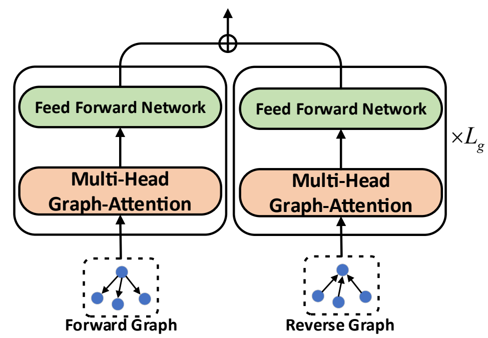


Figure 3: The structure of our graph encoder. The whole graph encoder consists of two component graph encoders, one for forward graph and another for reverse graph. Each component graph encoder is a stack of $L_g$ identical layers, with a multi-head graph-attention block and a feed forward network. Finally, we combine the outputs of two encoders as the output of graph encoder.

and $E$ denotes the set of edges. Each edge represents a semantic relation, denoted as a triple $(head, type, tail)$, where $head, tail \in V$ represent the head node and the tail node of edge, and $type \in R_f = \{Self, BV, ARG1, \cdots\}$ represents the type of edge. Among them, $Self$ represents self to self relationship and other edge types represent the different semantic relationships in semantic dependency graph. In order to improve the information propagation process in the graph, we add reverse edge to represent the relationship from tail node to head node, e.g. $(tail, rtype, head)$, where $rtype \in R_r = \{rSelf, rBV, rARG1, \cdots\}$. $R_r$ represents the set of reverse edge relationships corresponding with $R_f$. Then, we split the directed graph into a forward graph which the edge type in $R_f$ and a reverse graph which the edge type in $R_r$. Figure 2 show an example of forward graph and reverse graph. We denote the input sentence's word embedding matrix with positional encoding as $h_p$.

## Sentence Encoder

Sentence encoder aims to obtain the input sentence's contextual representations. We choose Transformer encoder (Vaswani et al. 2017) as our sentence encoder because of its excellent performance in many tasks.

The Transformer encoder is a stack of $L_s$ identical layers, and each layer includes a multi-head self-attention and a fully connected feed-forward network. The input to sentence encoder is $h_p$. We take the output of Transformer encoder as the final output of sentence encoder, which is denoted as $s$.

## Graph Encoder

Most graph models consider directed graph as undirected graph. These models ignore the direction information in the graph. Instead, we split the semantic dependency graph into a forward graph and a reverse graph, and send them to different encoders to capture semantic relationships in different directions. Finally, we combine information in both directions as the output of graph encoder. Figure 3 shows the structure of the graph encoder.

Our graph encoder consists of a forward graph encoder and a reverse graph encoder. Both graph encoders are a stack of $L_g$ identical layers. Each layer include a multi-head graph attention block and a fully connected feed-forward network block. For convenience, we only describe one of them. The input to the graph encoder is $h_p$. We denote the output of $l$-th layer as $g^l = \{g_1^l, g_2^l, \cdots, g_N^l\}$, where $g_i^l \in \mathbb{R}^{d_{model}}$ is the representation of the $i$-th node in the graph.

First, we define a graph attention operation. Graph attention (Velickovic et al. 2018) is used to aggregate the information from neighbor nodes. For each edge $(head, type, tail)$, considering that different relations might have different influences on the tail node, we concatenate the representations of the tail node and the edge type, and further employ a GLU (Dauphin et al. 2017) for transformation. For one edge with the head node's representation as $u$[1], we denote the representation of the tail node as $v_{tail}$ and the representation of the edge type as $e_{type}$, where $u, v_{tail}, e_{type} \in \mathbb{R}^{d_u}$. The aggregate operation is as follows:

$$\begin{aligned} v_{agg} &= [v_{tail} \| e_{type}]\mathbf{W}_{agg} + \mathbf{b}_{agg} \\ gate &= \sigma([v_{tail} \| e_{type}]\mathbf{W}_{gate} + \mathbf{b}_{gate}) \\ v &= gate \otimes v_{agg} \end{aligned} \tag{1}$$

where $\otimes$ is the element-wise product between matrices. $\|$ is concatenation operation. Learnable parameter matrices include $\mathbf{W}_{agg}, \mathbf{W}_{gate} \in \mathbb{R}^{2d_u \times d_u}$, $\mathbf{b}_{agg}, \mathbf{b}_{gate} \in \mathbb{R}^{d_u}$. $v \in \mathbb{R}^{d_u}$ is the representation vector by aggregating information of the tail node and the edge type.

Graph attention is used to induce a new representation of $u$ by aggregating the representations of its neighbor relations $v$, which can be computed as follows:

$$\mathrm{GAT}(u) = \sum_{v \in \mathcal{N}_u} \alpha_{uv} v \tag{2}$$

where $\mathcal{N}_u$ denotes all neighbor relation representations of node $u$. $\alpha_{uv}$ is the attention coefficient computed as follows:

[1]Note that we also use $u$ to denote the head node interchangeably.

$$\alpha_{uv} = \frac{\exp\left(\phi\left(\mathbf{a}^\top\left[\mathbf{W}_a u \| \mathbf{W}_a v\right]\right)\right)}{\sum_{v_k \in \mathcal{N}_u} \exp\left(\phi\left(\mathbf{a}^\top\left[\mathbf{W}_a u \| \mathbf{W}_a v_k\right]\right)\right)} \tag{3}$$

where $\phi$ is LeakyReLU activation function, $\mathbf{W}_a \in \mathbb{R}^{d_u \times d_u}, \mathbf{a} \in \mathbb{R}^{2d_u \times 1}$.

Then, we introduce a multi-head operation in graph attention. For simplicity and clarity, we omit the layer index $l$ for nodes.

$$\begin{aligned} \hat{g}_i^j &= \mathrm{GAT}(g_i \mathbf{W}_h^j) \\ \mathrm{Head}_j &= (\hat{g}_1^j, \hat{g}_2^j, \cdots, \hat{g}_N^j) \\ \mathrm{MultiHeadGAT}(g) &= \left(\Big\|_{j=1}^{H} \mathrm{Head}_j\right) \mathbf{W}_o \end{aligned} \tag{4}$$

where $H$ is the head number, $\mathbf{W}_h^j \in \mathbb{R}^{d_{model} \times d_u}$, $\mathbf{W}_o \in \mathbb{R}^{H * d_u \times d_{model}}$.

Finally, a feed-forward connection block follows multi-head GAT. The complete graph encoder is as follows:

$$\begin{aligned} \tilde{g}^l &= \mathrm{LN}(g^{l-1} + \mathrm{MultiHeadGAT}(g^{l-1})) \\ g^l &= \mathrm{LN}(\tilde{g}^l + \mathrm{FFN}(\tilde{g}^l)) \end{aligned} \tag{5}$$

where $\mathrm{FFN}(x)$ is a fully connected feed-forward network (Vaswani et al. 2017) and $\mathrm{LN}(x)$ is a layer normalization (Ba, Kiros, and Hinton 2016).

For each graph encoder, we take the output of the last layer as its final output. We send forward graph and reverse graph into different encoders, and get the forward graph encoder's output $\overrightarrow{g}$ and the reverse graph encoder's output $\overleftarrow{g}$. Finally, we combine the outputs in two directions as the final output of our whole graph encoder.

$$g = \overrightarrow{g} + \overleftarrow{g} \tag{6}$$

## Sentence Decoder

We leverage Transformer decoder as sentence decoder. We denote the output of the $l$-th decoder layer as $d^l$ and the total number of decoder layers as $L_d$. In order to utilize semantic graph information, we add graph attention block into each layer, which will be describe as follows.

The first block of each layer is a multi-head self-attention. Then, the output of self-attention is fed into two cross-attention blocks, one with sentence encoder's output and another with graph encoder's output. The output of cross-attention block is fed into a feed-forward network. The final output of the layer is as follows:

$$\begin{aligned} \tilde{d}^l &= \mathrm{LN}(d^{l-1} + \mathrm{MultiHeadAtt}(d^{l-1}, d^{l-1}, d^{l-1})) \\ \tilde{d}_s^l &= \mathrm{LN}(\tilde{d}^l + \mathrm{MultiHeadAtt}(\tilde{d}^l, s, s)) \\ \tilde{d}_g^l &= \mathrm{LN}(\tilde{d}_s^l + \mathrm{MultiHeadAtt}(\tilde{d}_s^l, g, g)) \\ d^l &= \mathrm{LN}(\tilde{d}_g^l + \mathrm{FFN}(\tilde{d}_g^l)) \end{aligned} \tag{7}$$

where $\mathrm{MultiHeadAtt}(Q, K, V)$ is a multi-head self-attention (Vaswani et al. 2017).

Finally, we map the output of the decoder to the target vocabulary size by linear transformation, and then use a softmax layer to calculate the probability.

$$P_g = \mathrm{softmax}(d^{L_d} \mathbf{W}_{out} + \mathbf{b}_{out}) \tag{8}$$

where $\mathbf{W}_{out} \in \mathbb{R}^{d_{model} \times d_{vocab}}, \mathbf{b}_{out} \in \mathbb{R}^{d_{vocab}}$ and $d_{vocab}$ is the size of target vocabulary.

Copy mechanism (See, Liu, and Manning 2017) has been proposed to tackle the out-of-vocabulary problem. We combine the outputs of graph encoder and sentence encoder to calculate the probability of copying from the original sentence. Finally, we replace UNK tokens with the word of the highest copy probability in the original sentence.

$$\begin{aligned} h_{comb} &= \mathrm{ReLU}([g \| s] \mathbf{W}_{comb} + \mathbf{b}_{comb}) \\ P_{copy} &= \mathrm{softmax}(d^{L_d} h_{comb}^T + \mathbf{b}_\epsilon) \end{aligned} \tag{9}$$

where $\mathbf{W}_{comb} \in \mathbb{R}^{2d_{model} \times d_{model}}$, $\mathbf{b}_{comb}, \mathbf{b}_\epsilon \in \mathbb{R}^{d_s}$, $d_s$ is the length of the original sentence.

The final output is a mixture of $P_g$ and $P_{copy}$ with a generation probability $\eta \in [0, 1]$, which is calculated by a linear transformation on $d^{L_d}$.

$$\begin{aligned} \eta &= \sigma(d^{L_d} \mathbf{W}_{eta} + \mathbf{b}_{eta}) \\ P &= \eta P_g + (1 - \eta) P_{copy} \end{aligned} \tag{10}$$

where $\mathbf{W}_{out} \in \mathbb{R}^{d_{model} \times d_{model}}, \mathbf{b}_{eta} \in \mathbb{R}^{d_{model}}$, and $\sigma$ is the sigmoid activation function.

## Objective Function

We leverage cross entropy with label smoothing as our loss function. We find that copy mechanism tends to copy original sentence completely without restriction. So we raise copy loss to penalize over-replication. Intuitively, adding the probability of copying from the original sentence into the loss can make copy mechanism avoid over-replication. So our loss function is as follows:

$$\begin{aligned} loss_c &= \sum_{i=1}^{d_s} (1 - \eta) p_c^i(w_t) \\ loss(w_t) &= -\,log\, P(w_t) + \lambda \times loss_c \end{aligned} \tag{11}$$

where $p_c^i(w_t)$ denotes the probability of copying the word $w_t$ from the $i$-th position of the original sentence, and $\lambda$ is a hyper-parameter.

## Length Token

Lakew, Gangi, and Federico (2019) introduced length token (LenTok) to control output length of neural machine translation, which can improve the fluency and readability of outputs.

In our model, we employ LenTok to compress outputs. We add $< \mathrm{SHORT} >$, $< \mathrm{MIDDLE} >$ and $< \mathrm{LONG} >$ tokens to the original sentence according to its target sentence's length in the training set. If the target sentence's length is less than $t_{min}$, we add $< \mathrm{SHORT} >$ to the beginning of the original sentence, and $< \mathrm{LONG} >$ will be added if the target sentence's length is greater than $t_{max}$.

The other sentences with target sentence's length during $[t_{min}, t_{max}]$ are added with $< \text{MIDDLE} >$. During testing, we add $< \text{MIDDLE} >$ to all original sentences. This can help to improve sentence quality.

# Experiments

## Datasets

We evaluate our SDISS model on three benchmark datasets: Newsela, WikiSmall and WikiLarge. **Newsela** (Xu, Callison-Burch, and Napoles 2015) consists of 1130 news articles which were rewritten four times by professional editors for children at different grade levels (0-4 from complex to simple). Zhang and Lapata (2017) provide standard splits and the train/dev/test sets contain 94,208/1,129/1,076 sentence pairs, respectively. **WikiLarge** (Zhang and Lapata 2017) is the largest TS corpus with 296,402/2,000/359 complex-simple sentence pairs for train/dev/test sets, constructed by merging previously created simplification corpora (Zhu, Bernhard, and Gurevych 2010; Woodsend and Lapata 2011; Kauchak 2013). The test set was created by employing Amazon Mechanical Turk workers to pair each complex sentence with 8 reference simplifications (Xu et al. 2016). **WikiSmall** was built by Zhu, Bernhard, and Gurevych (2010), and we use the standard splits with 88,837/205/100 pairs provided by Zhang and Lapata (2017) as train/dev/test sets. Table 2 provides statistics of the three benchmark training sets.

| **Dataset** | **Vocab Size** | | **token/sent** | |
|---|---|---|---|---|
| | **src** | **tgt** | **src** | **tgt** |
| Newsela | 41,066 | 30,193 | 25.94 | 15.89 |
| WikiSmall | 113,368 | 93,835 | 24.26 | 20.33 |
| WikiLarge | 201,841 | 168,962 | 25.17 | 18.51 |

Table 2: Statistic for training sets: the vocabulary size (Vocab Size) and the average token number per sentence (token/sent) for source (src) and target (tgt).

## Evaluation

We use two widely-used metrics to evaluate sentence simplification as in (Zhang and Lapata 2017)[2]: SARI (Xu et al. 2016) and FKGL (Kincaid et al. 1975) at the corpus-level. SARI evaluates the system's outputs by comparing them against the origin and reference sentences. FKGL measures the readability of the system's output (lower FKGL means simpler sentence). In addition, Dong et al. (2019) indicated that the MT-based models tend to learn a safe but undesirable strategy of copying the source sentences directly. So we test the percentage of unchanged output sentences compared with original sentences.

We also report BLEU score as reference. But recent studies have found that BLEU can not reflect simplification (Xu et al. 2016) and it is in fact negatively correlated with simplicity (Sulem, Abend, and Rappoport 2018a). Table 3 also shows that models with higher BLEU may cause lower SARI. Recently, more and more papers have not reported BLEU(Zhao et al. 2018; Dong et al. 2019). So we do not employ BLEU to evaluate our model. The subsequent human evaluation also shows that our model can produce sentences with good fluency although our BLEU is not the highest.

## Baseline

We compare our model with a variety of SMT-based TS models and NMT-based TS models. SMT-based TS models include phrase-based MT based model with re-ranking (PBMT-R) [3] (Wubben, van den Bosch, and Krahmer 2012) and Hybrid model [3] (Narayan and Gardent 2014) which leverages syntactic transformation. NMT-based TS models include DRESS and DRESS-Ls [3] (Zhang and Lapata 2017) based on deep reinforcement learning, $\text{N}_{\text{SE}}\text{L}_{\text{STM}}$-B and $\text{N}_{\text{SE}}\text{L}_{\text{STM}}$-S [4] Vu et al. (2018) based on neural semantic encoder, and EditNTS [5] (Dong et al. 2019) based on neural programmer-interpreter. For WikiLarge, we also compare with other two models with using external human knowledge: SBMT-SARI [3] (Xu et al. 2016), a SBMT-based system with external simplification component, and DMASS+DCSS [6] (Zhao et al. 2018), a Transformer-based model with external simplification rules. In addition, we compare our model against Transformer [7].

## Training Details

We set the dimensions of word embedding and hidden units $d_{model}$ to 256. For multi-head attention, we set the number of heads to 4. The number of layers for graph encoder, sentence encoder and decoder are all set to 6. We set $\lambda$ to 5 for Newsela, and 10 for WikiSmall and WikiLarge. We set $t_{max}$ to 15 for Newsela and 20 for WikiSmall, and $t_{min}$ is 0 for both datasets. For WikiLarge, because of its complexity, we set $t_{min}$ to 5 and $t_{max}$ to 50.

In order to reduce the vocabulary size, we refer to Zhang and Lapata (2017) to tag words with their named entities using the Stanford CoreNLP tool (Manning et al. 2014), and anonymize with a NE@$N$ token, where NE $\in$ {PER, LOC, ORG, MISC, $\cdots$}. $N$ indicates NE@$N$ is the $N$-th distinct NE typed entity. We replace the word with frequency no more than 2 as UNK token. During testing, we replace UNK with the original word with the highest copy probability.

[2] corpus-SARI script and FKGL tool are provided at https://github.com/XingxingZhang/dress.

[3] The outputs of PBMT-R, Hybrid, DRESS, DRESS-Ls and SBMT-SARI are provided by Zhang and Lapata (2017) via https://github.com/XingxingZhang/dress.

[4] As the full outputs of $\text{N}_{\text{SE}}\text{L}_{\text{STM}}$ are not available, we cannot compute the FKGL and human evaluation for this system.

[5] The outputs of EditNTS is provided by Dong et al. (2019) via https://github.com/yuedongP/EditNTS.

[6] The outputs of DMASS+DCSS is provided by Zhao et al. (2018) via https://github.com/Sanqiang/text_simplification.

[7] Transformer is implemented by ourselves.

## Results

### Automatic Evaluation

Table 3 shows the result of automatic evaluation. On the three datasets, our model significantly improves FKGL and SARI scores, and generally gets the sate-of-the-art performance.

| WikiLarge | BLEU | SARI ↑ | FKGL ↓ | % unc. |
|---|---|---|---|---|
| Reference | - | - | 8.88 | 15.88 |
| PBMT-R | 81.81 | 38.56 | 8.33 | **10.58** |
| Hybrid | 48.97 | 31.40 | **4.57** | 36.21 |
| Transformer | 84.27 | 35.34 | 7.92 | 23.08 |
| DRESS | 77.18 | 37.08 | 6.59 | 22.28 |
| DRESS-Ls | 80.12 | 37.27 | 6.62 | 27.02 |
| $\mathrm{N_{SE}L_{STM}}$-S | 80.43 | 36.88 | - | - |
| $\mathrm{N_{SE}L_{STM}}$-B | 92.02 | 33.43 | - | - |
| EditNTS | 86.68 | 38.22 | 7.30 | 10.86 |
| SDISS(ours) | 77.36 | **38.66** | 7.07 | 13.37 |
| **Models with external human knowledge** | | | | |
| SBMT-SARI | 73.08 | 39.96 | 7.29 | 9.47 |
| DMASS+DCSS | 80.53 | 40.45 | 7.79 | 6.69 |

| WikiSmall | BLEU | SARI ↑ | FKGL ↓ | % unc. |
|---|---|---|---|---|
| Reference | - | - | 8.86 | 3.00 |
| PBMT-R | 46.31 | 15.97 | 11.42 | 14.00 |
| Hybrid | 53.94 | 30.46 | 9.20 | 4.00 |
| Transformer | 49.85 | 27.92 | 8.00 | 15.03 |
| DRESS | 34.53 | 27.48 | 7.48 | 11.00 |
| DRESS-Ls | 36.32 | 27.24 | 7.55 | 13.00 |
| $\mathrm{N_{SE}L_{STM}}$-S | 29.72 | 29.75 | - | - |
| $\mathrm{N_{SE}L_{STM}}$-B | 53.42 | 17.47 | - | - |
| EditNTS | 23.87 | 32.35 | 5.47 | 0.00 |
| SDISS(ours) | 24.25 | **34.06** | **4.58** | **0.00** |

| Newsela | BLEU | SARI ↑ | FKGL ↓ | % unc. |
|---|---|---|---|---|
| Reference | - | - | 3.20 | 0.00 |
| PBMT-R | 18.19 | 15.77 | 7.59 | 5.85 |
| Hybrid | 14.46 | 30.00 | 4.01 | **3.34** |
| Transformer | 27.89 | 29.32 | 3.97 | 9.71 |
| DRESS | 23.21 | 27.37 | 4.11 | 11.98 |
| DRESS-Ls | 24.30 | 26.63 | 4.20 | 15.51 |
| $\mathrm{N_{SE}L_{STM}}$-S | 22.62 | 29.58 | - | - |
| $\mathrm{N_{SE}L_{STM}}$-B | 26.31 | 27.42 | - | - |
| EditNTS | 19.85 | 31.41 | 3.40 | 4.27 |
| SDISS(ours) | 18.81 | **32.30** | **2.38** | 5.01 |

Table 3: Automatic evaluation on Newsela, WikiSmall, and WikiLarge test sets. We report BLEU, SARI and FKGL at corpus-level, and unchanged sentence percentage (%unc.).

SARI has been the most important measurement for the simplification task, and it explicitly measures the goodness of words that are added, deleted and kept by the systems (Xu et al. 2016). Xu et al. (2016) demonstrated that SARI is highly correlated with human judgement. On SARI, our result is higher than that of the latest neural-network-based model EdiNTS by 0.89, 1.71 and 0.44 points on Newsela, WikiSmall and WikiLarge respectively. This means that our model can more accurately simplify sentence on the semantic level. SBMT-SARI and DMASS+DCSS have higher SARI scores than our model on WikiLarge, which is due to the use of external human knowledge. However, our model has better human evaluation results than them, as discussed later.

In term of FKGL, it can measure the readability of sentences. The lower value of FKGL means the sentences are easier to understand. Our model has lowest FKGL on Newsela and WikiSmall. This means that our model can produce sentences for easier understanding than previous works. On WikiLarge, although the FKGL scores of Hybrid and DRESS are lower than that of our model, their SARI scores are much lower than that of our model, which means they tend to generate easy-to-understand sentences by sacrificing simplification accuracy. Because of the use of copy loss, our model has significantly lower values of unchanged sentence percentage than other MT-based models.

### Ablation Study

We perform ablation study on Newsela to investigate the influence of different modules in our SDISS model. We compare the full model with its variants. We remove copy loss, LenTok, graph encoder and sentence encoder separately to obtain four variant models. The results are shown in Table 4.

| Newsela | SARI ↑ | FKGL ↓ | %unc. |
|---|---|---|---|
| SDISS | **32.30** | **2.38** | **5.01** |
| w/o copy loss | 31.89 | 3.32 | 13.35 |
| w/o LenTok | 31.30 | 4.20 | 8.57 |
| w/o GEncoder | 30.24 | 3.56 | 9.07 |
| w/o SenEncoder | 31.55 | 3.98 | 10.32 |

Table 4: Results of ablation study on Newsela. GEncoder stands for graph encoder and SenEncoder stands for sentence encoder.

We can see that each module in our model does contribute to the overall performance. The use of graph encoder can substantially improve the performance of TS system. Meanwhile, the traditional sentence encoder can complement the graph encoder. In addition, copy loss is helpful to control copying the whole sentence and LenTok is helpful to compress sentence to be easier to understand.

### Human Evaluation

We perform human evaluation of system outputs with respect to three aspects: fluency, adequacy and simplicity. Fluency indicates if the output is syntactically correct; Adequacy indicates if the meaning expressed in the original sentence is preserved in the output; Simplicity indicates if the output simplifies the original sentence. All ratings were obtained using a five point Likert scale (the larger, the better). We follow the approach of Zhang and Lapata (2017) to sample 100 instances, including 30 from Newsela, 30 from WikiSmall and 40 from WikiLarge. We employ 6 graduate students to rate each instance, and we ensure every instance is rated by at least three judges. The results are shown in Table 5.

| | **Newsela** | | | | **WikiSmall** | | | | **WikiLarge** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | F | A | S | avg. | F | A | S | avg. | F | A | S | avg. |
| Reference | 4.29 | 2.84 | 3.73 | 3.62 | 4.28 | 3.78 | 3.24 | 3.77 | 4.26 | 4.09 | 2.62 | 3.65 |
| PBMT-R | 3.68 | **3.64** | 2.12 | 3.15 | 4.10 | **4.04** | 2.20 | 3.45 | 4.09 | 3.92 | 2.43 | 3.48 |
| SBMT-SARI | - | - | - | - | - | - | - | - | 4.03 | 3.72 | 2.52 | 3.42 |
| DMASS+DCSS | - | - | - | - | - | - | - | - | 4.12 | 3.75 | 2.76 | 3.54 |
| Hybrid | 2.98 | 2.62 | 2.74 | 2.78 | 3.41 | 3.64 | 2.50 | 3.18 | 2.93 | 2.66 | **3.68** | 3.09 |
| DRESS | 3.93 | 2.95 | 3.01 | 3.30 | **4.29** | 3.42 | 3.54 | 3.75 | 4.30 | 3.65 | 3.32 | 3.76 |
| Transformer | 3.81 | 2.72 | 3.04 | 3.19 | 4.25 | 3.62 | 2.43 | 3.43 | 4.32 | 3.97 | 1.89 | 3.39 |
| EditNTS | **3.98** | 3.02 | 3.23 | 3.41 | 4.09 | 3.12 | **3.94** | 3.72 | 4.40 | 4.03 | 2.98 | 3.80 |
| SDISS(ours) | 3.82 | 3.23 | **3.32** | **3.46** | 4.12 | 3.67 | 3.85 | **3.88** | **4.62** | **4.13** | 3.23 | **3.99** |

Table 5: Human evaluation on Newsela, WikiSmall and WikiLarge test sets. Metrics include Fluency(F), Adequacy(A), Simplicity(S), and Average score(avg.). We choose 100 sentences for human judgement, including 30 sentences from Newsela, 30 sentences from WikiSmall and 40 sentences from WikiLarge. The references are also manually evaluated for comparison.

| **Cases from Newsela** | |
|---|---|
| Source | To prevent overfishing , the agreement would , among other things , make it much easier to establish marine protected areas -LRB- MPAs -RRB- in the high seas . |
| Reference | An agreement would make it easier to create marine protected areas . |
| PBMT-R | To prevent overfishing , the agreement would , among other things , make it much easier to establish marine protected areas -LRB- MPAs -RRB- in the high seas . |
| Hybrid | Overfishing to prevent the agreement would make it to establish areas . |
| DRESS | To prevent overfishing , the agreement would , among other things , make it much easier to establish marine protected areas . |
| EditNTS | The agreement would make it much easier to protected areas . |
| SDISS(ours) | The agreement would make it much easier to establish marine protected areas . |

| **Cases from WikiSmall** | |
|---|---|
| Source | A naval mine is a self-contained explosive device placed in water to destroy ships or submarines . |
| Reference | A naval mine is a bomb placed in water to destroy ships or submarines . |
| PBMT-R | A naval mine is a separate explosive device placed in the water to destroy ships and submarines . |
| Hybrid | A naval mine is a device explosive placed in water to destroy ships and submarines . |
| DRESS | A naval mine is a self-contained explosive device placed in water to destroy ships or submarines . |
| EditNTS | A naval mine is a explosive device can be . |
| SDISS (ours) | A naval mine is a explosive device placed in water to destroy ships or submarines . |

Table 6: System outputs for two sentences from Newsela and WikiSmall respectively.

From the table, we can see that the outputs of our model have better adequacy than that of previous neural network based models and also have high scores in fluency. Especially on complex datasets like WikiLarge, the outputs of our model show very good adequacy and fluency. With respect to simplicity, our model achieves relatively high scores and it tends to generate more adequate sentences rather than simpler sentences without adequacy. In all, our model achieves the highest average scores on the three datasets.

## Case Study

We perform case studies for better understanding the model performance. In Table 6 we choose two samples from Newela and WikiSmall, respectively, and compare our model with previous strong TS systems including PBMT-R, Hybrid, DRESS and EditNTS. Obviously, PBMT-R and DRESS tend to copy the whole sentence without change, and Hybrid and EditNTS prefer to generate short sentences with big semantic deviation. In contrast, our SDISS model can generate sentences as simple as possible but without semantic deviation.

After analyzing the cases, we find MT-based models cannot remove some redundant modifiers effectively like adjective and adverbs, because they only focus on the word order of sentences. EditNTS usually deletes important information of the original sentence and generates shorter sentences. By introducing semantic information, SDISS can generate simplified sentences by removing redundant parts from a semantic perspective, and thus generate sentences with high semantic relevance.

# Conclusion

In this paper, we explore to incorporate semantic dependency graph into neural sentence simplification model. We propose a new model called SDISS, which can leverage the semantic dependency graph of input sentence to guide the simplification process. Our model generally achieves state-of-the-art performance on three benchmark datasets. Both automatic evaluation and human judgement indicate that our model improves semantic relevance. In the future, we will consider other semantic formalism like AMR and MRS to simplify sentences.

## Acknowledgments

This work was supported by National Natural Science Foundation of China (61772036), Beijing Academy of Artificial Intelligence (BAAI) and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). We appreciate the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

## References

Ba, L. J.; Kiros, R.; and Hinton, G. E. 2016. Layer Normalization. *CoRR* abs/1607.06450.

Carroll, J. A.; Minnen, G.; Pearce, D.; Canning, Y.; Devlin, S.; and Tait, J. 1999. Simplifying text for language-impaired readers. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*.

Chandrasekar, R.; Doran, C.; and Srinivas, B. 1996. Motivations and methods for text simplification. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, 1041–1044. Association for Computational Linguistics.

Chen, Y.; Ye, Y.; and Sun, W. 2019. Peking at MRP 2019: Factorization- and Composition-Based Parsing for Elementary Dependency Structures. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, 166–176. Hong Kong: Association for Computational Linguistics. doi:10.18653/v1/K19-2016.

Coster, W.; and Kauchak, D. 2011. Learning to Simplify Sentences Using Wikipedia. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, 1–9.

Dauphin, Y. N.; Fan, A.; Auli, M.; and Grangier, D. 2017. Language modeling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 933–941. JMLR. org.

Dong, Y.; Li, Z.; Rezagholizadeh, M.; and Cheung, J. C. K. 2019. EditNTS: An Neural Programmer-Interpreter Model for Sentence Simplification through Explicit Editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 3393–3402.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation* 9(8): 1735–1780.

Kauchak, D. 2013. Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 1: Long papers)*, 1537–1546.

Kincaid, J. P.; Fishburne Jr, R. P.; Rogers, R. L.; and Chissom, B. S. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Naval Technical Training Command Millington TN Research Branch.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Klebanov, B. B.; Knight, K.; and Marcu, D. 2004. Text simplification for information-seeking applications. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, 735–747. Springer.

Lakew, S. M.; Gangi, M. A. D.; and Federico, M. 2019. Controlling the Output Length of Neural Machine Translation. *CoRR* abs/1910.10408.

Manning, C. D.; Surdeanu, M.; Bauer, J.; Finkel, J. R.; Bethard, S.; and McClosky, D. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 55–60.

Martin, L.; Fan, A.; de la Clergerie, É.; Bordes, A.; and Sagot, B. 2020. Multilingual Unsupervised Sentence Simplification. *CoRR* abs/2005.00352.

Narayan, S.; and Gardent, C. 2014. Hybrid Simplification using Deep Semantics and Machine Translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 435–445.

Narayan, S.; and Gardent, C. 2016. Unsupervised Sentence Simplification Using Deep Semantics. In *The 9th International Natural Language Generation conference*, 111–120.

Nisioi, S.; Štajner, S.; Ponzetto, S. P.; and Dinu, L. P. 2017. Exploring neural text simplification models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 85–91.

Oepen, S.; Kuhlmann, M.; Miyao, Y.; Zeman, D.; Cinková, S.; Flickinger, D.; Hajic, J.; Ivanova, A.; and Uresova, Z. 2016. Towards comparability of linguistic graph banks for semantic parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 3991–3995.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2008. The graph neural network model. *IEEE Transactions on Neural Networks* 20(1): 61–80.

See, A.; Liu, P. J.; and Manning, C. D. 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)*, 1073–1083.

Song, K.; Zhao, L.; and Liu, F. 2018. Structure-Infused Copy Mechanisms for Abstractive Summarization. In *Proceedings of the 27th International Conference on Computational Linguistics*, 1717–1729. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Song, L.; Gildea, D.; Zhang, Y.; Wang, Z.; and Su, J. 2019. Semantic Neural Machine Translation using AMR. *Transactions of the Association for Computational Linguistics* 7: 19–31.

Štajner, S.; Béchara, H.; and Saggion, H. 2015. A deeper exploration of the standard PB-SMT approach to text simplification and its evaluation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 823–828.

Sulem, E.; Abend, O.; and Rappoport, A. 2018a. BLEU is Not Suitable for the Evaluation of Text Simplification. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018* 738–744. doi:10.18653/v1/d18-1081.

Sulem, E.; Abend, O.; and Rappoport, A. 2018b. Simple and Effective Text Simplification Using Semantic and Neural Methods. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* 162–173. doi:10.18653/v1/P18-1016.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings* .

Vu, T.; Hu, B.; Munkhdalai, T.; and Yu, H. 2018. Sentence Simplification with Memory-Augmented Neural Networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 79–85.

Watanabe, W. M.; Junior, A. C.; Uzêda, V. R.; Fortes, R. P. d. M.; Pardo, T. A. S.; and Aluísio, S. M. 2009. Facilita: reading assistance for low-literacy readers. In *Proceedings of the 27th ACM international conference on Design of communication*, 29–36.

Woodsend, K.; and Lapata, M. 2011. Learning to Simplify Sentences with Quasi-Synchronous Grammar and Integer Programming. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 409–420.

Wubben, S.; van den Bosch, A.; and Krahmer, E. 2012. Sentence Simplification by Monolingual Machine Translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1015–1024.

Xu, W.; Callison-Burch, C.; and Napoles, C. 2015. Problems in Current Text Simplification Research: New Data Can Help. *Transactions of the Association for Computational Linguistics* 3: 283–297.

Xu, W.; Napoles, C.; Pavlick, E.; Chen, Q.; and Callison-Burch, C. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics* 4: 401–415.

Zhang, X.; and Lapata, M. 2017. Sentence Simplification with Deep Reinforcement Learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 584–594.

Zhao, S.; Meng, R.; He, D.; Saptono, A.; and Parmanto, B. 2018. Integrating Transformer and Paraphrase Rules for Sentence Simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3164–3173.

Zhao, Y.; Chen, L.; Chen, Z.; and Yu, K. 2020. Semi-Supervised Text Simplification with Back-Translation and Asymmetric Denoising Autoencoders. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, 9668–9675. AAAI Press. URL https://aaai.org/ojs/index.php/AAAI/article/view/6515.

Zhu, Z.; Bernhard, D.; and Gurevych, I. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd international conference on computational linguistics*, 1353–1361. Association for Computational Linguistics.