Denoising based Sequence-to-Sequence Pre-training for Text Generation

Liang Wang[1], Wei Zhao[1], Ruoyu Jia[1], Sujian Li[2], Jingming Liu[1]
[1]Yuanfudao AI Lab, Beijing, China
[2]Key Laboratory of Computational Linguistics, Peking University, MOE, China
{wangliang01,zhaowei01,jiary,liujm}@fenbi.com
lisujian@pku.edu.cn

Abstract

This paper presents a new sequence-to-sequence (seq2seq) pre-training method PoDA (**P**re-training **o**f **D**enoising **A**utoencoders), which learns representations suitable for text generation tasks. Unlike encoder-only (e.g., BERT) or decoder-only (e.g., OpenAI GPT) pre-training approaches, PoDA jointly pre-trains both the encoder and decoder by denoising the noise-corrupted text, and it also has the advantage of keeping the network architecture unchanged in the subsequent fine-tuning stage. Meanwhile, we design a hybrid model of Transformer and pointer-generator networks as the backbone architecture for PoDA. We conduct experiments on two text generation tasks: abstractive summarization, and grammatical error correction. Results on four datasets show that PoDA can improve model performance over strong baselines without using any task-specific techniques and significantly speed up convergence. [1]

1 Introduction

Methods based on unsupervised pre-training and supervised fine-tuning for NLP have achieved phenomenal successes in the last two years. Most of the proposed methods in the literature choose language modeling or its variant as the pre-training task. After the pre-training stage, ELMo (Peters et al., 2018) and CoVe (McCann et al., 2017) directly use the learned representations as additional features for downstream tasks, while BERT (Devlin et al., 2018), ULMFiT (Howard and Ruder, 2018), XLM (Lample and Conneau, 2019), and OpenAI GPT (Radford et al., 2018, 2019) require fine-tuning both pre-trained parameters and task-specific parameters on labeled data. The state-of-the-art performances have been significantly advanced for classification and sequence labeling tasks, such as natural language inference (Bowman et al., 2015), named-entity recognition, SQuAD question answering (Rajpurkar et al., 2016) etc.

However, little attention has been paid to pre-training for seq2seq text generation (Sutskever et al., 2014). A typical seq2seq network consists of a bidirectional encoder, a unidirectional decoder and attention between the encoder and decoder. Previous work mainly focuses on encoder-only or decoder-only pre-training. For example, BERT pre-trains a bidirectional encoder, and OpenAI GPT pre-trains a language model which is essentially a unidirectional decoder. Ramachandran et al. (2016) propose to train two independent language models for the encoder and decoder respectively. All of the aforementioned methods are only able to partially pre-train the seq2seq networks, and therefore are unable to unleash the full potential of transfer learning for text generation.

In this paper, we present PoDA, a denoising based pre-training method that is able to jointly pre-train all components of seq2seq networks. Like denoising autoencoders, PoDA works by denoising the noise-corrupted text sequences. Any noising function that fits in the seq2seq framework can be used. We experiment with three types of noises: randomly shuffle, delete or replace the words in a given sequence. It is noted PoDA is simple, easy-to-implement and applicable to virtually all seq2seq architectures, including ConvS2S (Gehring et al., 2017) and Transformer (Vaswani et al., 2017). Here, we adopt the hybrid architecture of Transformer and pointer-generator networks (See et al., 2017). Transformer is effective at modeling long-distance dependencies, highly parallelizable and demonstrates good performance empirically. Pointer-generator network incorporates copying mechanism (Gu et al., 2016; Gulcehre et al., 2016) which is helpful for most text

[1]The code and pre-trained models are available at https://github.com/yuantiku/PoDA.

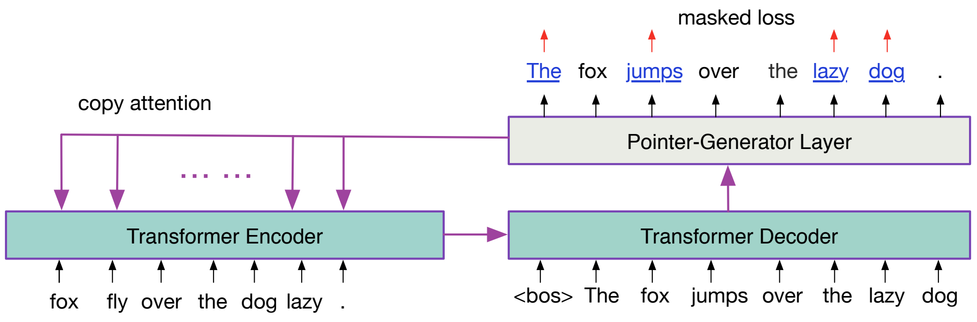
masked loss
The fox jumps over the lazy dog .
copy attention
Pointer-Generator Layer
Transformer Encoder
Transformer Decoder
fox fly over the dog lazy .
<bos> The fox jumps over the lazy dog

Figure 1: PoDA model architecture. The masked loss is calculated only for the blue underlined words. "<bos>" is a special begin-of-sequence padding symbol. The example input-output pair is explained in Section 2.2.

generation tasks.

The text corpora used for pre-training are the Billion Word Benchmark (Chelba et al., 2013) and English Wikipedia, both of which are publicly available and consists of nearly 2.99 billion words in total. We conduct experiments on two abstractive summarization datasets (CNN/Daily Mail (See et al., 2017) and Gigaword (Rush et al., 2015)), and two grammatical error correction datasets (CoNLL-2014 (Ng et al., 2014) and JFLEG (Napoles et al., 2017)). With simple maximum likelihood training and no task-specific techniques, PoDA achieves superior or comparable performance against state-of-the-art systems and speeds up convergence for all four datasets.

2 Method

2.1 Model Architecture

First, we design a seq2seq model as the backbone architecture of our proposed pre-training method, which is a combination of Transformer and pointer-generator networks, as shown in Figure 1.

The input representations are the sum of word embeddings and sinusoidal positional encodings. Both the Transformer encoder and the decoder consist of 6 layers of transformer blocks, and each block is a multi-head self-attention layer followed by one layer of positionwise feedforward network.

For the output layer, we use a pointer-generator layer to allow both copying from the input sequence and generation from a fixed vocabulary. The implementation is detailed in Appendix.

As a side note, we want to point out that the seq2seq architecture is not limited to the one we propose and other networks such as ConvS2S, RNN-based seq2seq models are also applicable. Pointer-generator networks are also not the only solution for handling out-of-vocabulary(OOV) words, and subword-based methods such as sentencepiece (Kudo and Richardson, 2018) can be used at the cost of making the input and output sequences longer.

2.2 Noising and Denoising

Similar to denoising autoencoders, PoDA involves two parts: noising and denoising. The noising part corrupts a given word sequence $\mathbf{x} = \{\mathbf{x}_i\}_{i=1}^{n}$ and gets a noisy word sequence $\mathbf{x}' = \{\mathbf{x}'_i\}_{i=1}^{n'}$. The denoising part tries to recover $\mathbf{x}$ given $\mathbf{x}'$ using a seq2seq model.

We use three noising functions: randomly shuffle, delete or replace the words in $\mathbf{x}$. The details are shown in Algorithm 1, where $N(0, \sigma)$ is a gaussian distribution with mean 0 and variance σ. $B(p)$ is a Bernoulli distribution, and $Beta(\alpha, \beta)$ is a beta distribution serving as the prior for $B(p)$. Take function DELETE (line 10 to line 15 in Algorithm 1) as an example, it first samples a Bernoulli distribution with expectation p from $Beta(\alpha, \beta)$, then each word is deleted with probability p. The usage of $Beta(\alpha, \beta)$ prior can make the model robust to different degrees of noises.

We exemplify the operations above in Figure 1. The original word sequence $\mathbf{x}$ =*"The fox jumps over the lazy dog ."*, after three noising operations: delete *"The"*, replace *"jumps"* with *"fly"* and swap *"lazy"* and *"dog"*, we get the noisy word sequence $\mathbf{x}'$ =*"fox fly over the dog lazy ."*.

The denoising part maximizes the conditional probability $p(\mathbf{x}|\mathbf{x}')$, which can be factorized as:

$$p(\mathbf{x}|\mathbf{x}') = \Pi_{i=1}^{n} p(\mathbf{x}_i|\mathbf{x}', \mathbf{x}_{<i}) \tag{1}$$

When predicting $\mathbf{x}_i$, it is conditioned on the noise-corrupted full context $\mathbf{x}'$ and the clean left

Algorithm 1 The Noising Algorithm

Input: $\mathbf{x}$ is a sequence of words
α, β, σ are hyperparameters

```
1:  function NOISING($\mathbf{x}$)
2:      Apply SHUFFLE, DELETE, REPLACE to $\mathbf{x}$ in random order
3:  end function

4:  function SHUFFLE($\mathbf{x}$)
5:      for $i \leftarrow 1$ to $len(\mathbf{x})$ do
6:          $indices[i] \leftarrow i + \delta_i \sim N(0, \sigma)$
7:      end for
8:      Rearrange $\mathbf{x}$ based on $argsort(indices)$
9:  end function
10: function DELETE($\mathbf{x}$)
11:     Sample $p \sim Beta(\alpha, \beta)$
12:     for $w$ in $\mathbf{x}$ do
13:         Delete $w$ if $\mu \sim B(p)$ is 1
14:     end for
15: end function
16: function REPLACE($\mathbf{x}$)
17:     Sample $p \sim Beta(\alpha, \beta)$
18:     for $w$ in $\mathbf{x}$ do
19:         Replace $w$ with $w'$ sampled from unigram distribution if $\mu \sim B(p)$ is 1
20:     end for
21: end function
```

context $\mathbf{x}_{<i}$. This shows that our seq2seq formulation is capable of unifying both encoder-only pre-training and decoder-only pre-training methods, since a bidirectional language model used by BERT can be seen as simulating $p(\mathbf{x}_i|\mathbf{x}')$, while a traditional unidirectional language model used by OpenAI GPT as resembling $p(\mathbf{x}_i|\mathbf{x}_{<i})$.

Like BERT, we add a mask to the target sequence when computing the loss function. To force the model to learn meaningful representations, instead of copying from the input most of the time, the positions where the corresponding words are corrupted in the input are kept. We also keep a small percentage (3%) of positions where the words are not corrupted, so that the model can learn to copy from the input when appropriate. Then, the training loss with mask is as follows ($\mathbf{\Theta}$ is model parameters):

$$L = -\sum_{i=1}^{n} mask(i) \cdot \log p(\mathbf{x}_i|\mathbf{x}', \mathbf{x}_{<i}, \mathbf{\Theta}) \quad (2)$$

Empirically, we set $\sigma = 0.5$ for Gaussian distribution. α and β are chosen to have a Beta distribution with mean 0.15 and standard deviation 0.03.

2.3 Pre-training Procedure

Corpus	#words
English Wikipedia	2.22B
Billion Word Benchmark	0.76B
Total	2.99B

Table 1: Text corpora used for pre-training.

For pre-training, we use two text corpora: the full dump of English Wikipedia[2] and the Billion Word Benchmark[3], as shown in Table 1. For English Wikipedia, we remove paragraphs with less than 3 words or more than 30% OOV words, and each paragraph is split into text segments with no more than 128 words for each segment. The Billion Word Benchmark is a sentence-level corpus. Sentences with more than 500 words are ignored during training.

The pre-training is performed on 4 GPUs using synchronous data parallelism, gradients are averaged across different GPUs. Each batch on a single GPU consists of at most 3000 tokens. We pre-train the network for 5 million iterations, which is roughly 14 epochs over the entire dataset. The final perplexity on the validation set is about 6.8. Each epoch takes approximately 2 days. Details on the network hyperparameters and optimizers are given in Section 3.1.

2.4 Fine-tuning Procedure

With our pre-training method, we do not need to change the network architecture during the fine-tuning stage, since both the pre-training task and text generation tasks take a source sequence as input and return a target sequence as output. The network is initialized with pre-trained parameter values. For fine-tuning, the preprocessing is dataset-specific, but the learning rate scheduling, dropout, early stopping, and gradient clipping are exactly the same as pre-training.

The objective function for fine-tuning is the word-level negative log-likelihood. Here we do not use reinforcement learning to tune towards the automatic evaluation metrics such as ROUGE

[2]`https://dumps.wikimedia.org/`

[3]`http://www.statmt.org/lm-benchmark/`, we do not use BooksCorpus (Zhu et al., 2015) used by BERT, because it is not publicly available now.

(Lin, 2004) or BLEU (Papineni et al., 2002), because it may overfit evaluation metrics and barely show improvements in human evaluations (Wu et al., 2016).

3 Experiments

3.1 Setup

The network architecture used by our experiments has 97 million parameters. It consists of 6 layers of encoder blocks, 6 layers of decoder blocks, and 1 pointer-generator layer. The hidden size of each positionwise feedforward layer is 4096. We use 8 heads for all multi-head attention layers. The vocabulary consists of the top $50k$ most frequent words (case sensitive), and the dimension of word embedding is 512. We tie the parameters of encoder word embeddings, decoder word embeddings, and the output softmax layer. NAG (Nesterov Accelerated Gradient) optimizer is used with initial learning rate 2×10^{-3}. Dropout of 0.2 is applied for all self-attention layers, positionwise feedforward layers and input embedding layers. The gradient norm is clipped to have a maximum value of 2. We follow the Transformer implementation from *fairseq*[4].

For task-specific fine-tuning, unless explicitly specified, we reuse the hyperparameters from the pre-training stage. After each training epoch, we compute the validation loss and halve the learning rate whenever the validation loss stops decreasing. The training procedure terminates if the learning rate drops below 10^{-4}. Exponential moving average (EMA) with decay rate 0.9995 is used to make the training stabilized. At inference time, we use standard beam search decoding based on the length-normalized log-likelihood. For ensemble models, we use different random seeds and pre-trained checkpoints for fine-tuning. Ensemble decoding is used by averaging the output probabilities from different models at every decoding step.

When reporting experimental results, *"PoDA w/o pre-training"* refers to the proposed architecture in Section 2.1 trained only on the supervised data, and *"PoDA w/o fine-tuning"* only pre-trains on unlabeled data. *PoDA* first pre-trains a denoising autoencoder and then fine-tunes on the supervised data.

[4]https://github.com/pytorch/fairseq

3.2 Abstractive Summarization

Datasets We use two summarization datasets: CNN/Daily Mail[5] (See et al., 2017) and Gigaword (Rush et al., 2015) dataset. The official split for training, validation, and test is shown in Table 2.

Corpus	# of examples		
	train	valid	test
CNN/Daily Mail	287,113	13,368	11,490
Gigaword	3,803,957	189,651	1,951

Table 2: Dataset statistics for abstractive summarization.

The CNN/Daily Mail dataset contains approximately $300k$ news articles with an average of 781 words for each article, and each article is paired with summaries with 56 words on average. We use the preprocessing script[6] provided by See et al. (2017). The articles are truncated to 800 words for both training and testing. The summaries are truncated to 130 words for training.

The Gigaword is a headline-generation dataset consisting of nearly 4 million examples. Headline generation can be seen as a sentence summarization task. Each example in Gigaword consists of one sentence with an average length of 31.3 words, which is much shorter than CNN/Daily Mail, and one short headline with an average length of 8.3 words. The Gigaword dataset provided by Rush et al. (2015) is already tokenized and lower-cased. Since our vocabulary is case-sensitive, such inconsistency is expected to hurt our system's performance.

Evaluation We report evaluation results in terms of of ROUGE-1, ROUGE-2 and ROUGE-L (Lin, 2004) using the *pyrouge*[7] package. For the CNN/Daily Mail dataset, PGNet (See et al., 2017), Lead3 (See et al., 2017), rnn-ext + RL (**?**), NeuSum (Zhou et al., 2018) are used as baselines. For the Gigaword dataset, ABS+ (Rush et al., 2015), CGU (Lin et al., 2018), FTSum (Cao et al., 2018b), and Re^{3}Sum (Cao et al., 2018a) are used as baselines.

Results for CNN/Daily Mail Considering the characteristics of news articles, baselines such as Lead3 (simply choose the first 3 sentences) can achieve strong performance in terms of ROUGE

[5]We use the non-anonymized version, which is considered to be more realistic.

[6]https://github.com/abisee/cnn-dailymail

[7]https://github.com/andersjo/pyrouge

System	ROUGE		
	1	2	L
Lead3	40.34	17.70	36.57
PGNet	36.44	15.66	33.42
rnn-ext + RL	41.47	18.72	37.76
NeuSum	41.59	19.01	37.98
PoDA w/o pre-training	40.82	18.46	37.61
PoDA	**41.87**	**19.27**	**38.54**

Table 3: ROUGE scores for CNN/Daily Mail dataset.

scores, as shown in Table 3. "rnn-ext+RL" combines both extractive and abstractive methods and achieves performance improvements (?). PoDA is a purely abstractive summarization system and performs stably better than all the methods. "PoDA w/o pre-training" only has moderate success with ROUGE-1 40.82, ROUGE-2 18.46 and ROUGE-L 37.61. When combined with pre-training, PoDA establishes new state-of-the-art on CNN/Daily Mail dataset.

System	ROUGE		
	1	2	L
ABS+	29.76	11.88	26.96
CGU	36.3	18.0	33.8
FTSum	37.27	17.65	34.24
Re^3Sum	37.04	19.03	34.46
PoDA w/o pre-training	37.24	18.28	34.53
PoDA	**38.29**	**19.06**	**35.45**

Table 4: ROUGE scores for Gigaword dataset.

Results for Gigaword The Gigaword dataset is much larger than CNN/Daily Mail, and this enables "PoDA w/o pre-training" to have competitive performance even without pre-training. As in Table 4, PoDA substantially improves ROUGE-1 from 37.24 to 38.29(+1.05), ROUGE-2 from 18.28 to 19.06(+0.78), and ROUGE-L from 34.53 to 35.45(+0.92), with pre-training added. We can see that PoDA performs favorably over the state-of-the-art Re^3Sum method, which utilizes unlabeled text corpora by first retrieving and then rewriting relevant snippets.

3.3 Grammatical Error Correction (GEC)

Datasets GEC can also be seen as a text generation task, where the input sequence is a sentence possibly containing some grammatical errors, and the output is a clean and grammatical sentence. We experiment PoDA on two GEC datasets: CoNLL-2014 (Ng et al., 2014) and JFLEG (Napoles et al., 2017). We use three public datasets for training: Lang-8 NAIST (Mizumoto et al., 2011), NUCLE (Dahlmeier et al., 2013) and CLC FCE (Felice et al., 2014). The test set of CoNLL-2013 shared task is used as validation set for the CoNLL-2014 task. JFLEG has its own validation set. For preprocessing, we use NLTK[8] to tokenize sentences, and remove all sentence pairs without any edits in Lang-8 NAIST. Simple spelling errors are corrected based on edit distance. The dataset statistics are shown in Table 5.

Corpus	#Sent Pairs	Split
Lang-8 NAIST	$1,097,274$	train
NUCLE	$57,113$	train
CLC FCE	$32,073$	train
CoNLL-2013 test set	$1,381$	valid
JFLEG valid set	754	valid
CoNLL-2014 test set	$1,312$	test
JFLEG test set	747	test

Table 5: Dataset statistics for grammatical error correction. Due to the lack of standard preprocessing script, the number of sentence pairs in the training set are slightly different from previous work.

Evaluation To compare with previous work, we use the official evaluation metrics: MaxMatch (M^2) $F_{0.5}$ (Dahlmeier and Ng, 2012) for CoNLL-2014 and GLEU (Napoles et al., 2015) for JFLEG dataset. Both metrics are shown to correlate well with human evaluation scores. MLConv (Chollampatt and Ng, 2018a), char-seq2seq (Xie et al., 2016), dual-boost (Ge et al., 2018a), Hybrid SMT-NMT (Grundkiewicz and Junczys-Dowmunt, 2018), NQE (Chollampatt and Ng, 2018b), and NRL (Sakaguchi et al., 2017) are used as baselines.

Results As shown in Table 6 and Table 7, when trained only on the supervised data, "PoDA w/o pre-training" can still achieve an impressive performance with $F_{0.5}$ score 54.01 on CoNLL-2014 test set and GLEU score 56.52 on JFLEG test set, surpassing previous state-of-the-art single model results. This once again shows the effectiveness of the Transformer architecture. For GEC task, most words in the output sequence also appear in the input sequence, pointer-generator makes it easier to learn such prior. With denoising based pre-training, PoDA greatly boosts the $F_{0.5}$ score from 54.01 to 59.40(+5.39) for CoNLL-2014 dataset,

[8] `https://www.nltk.org/`

System (single)	P	R	$F_{0.5}$
char-seq2seq	49.24	23.77	40.56
MLConv	60.90	23.74	46.38
dual-boost	62.70	27.69	50.04
PoDA w/o fine-tuning	19.22	31.73	20.86
PoDA w/o pre-training	65.63	31.62	54.01
PoDA	**70.10**	**36.88**	**59.40**
Ensemble			
MLConv(+rerank)	65.49	33.14	54.79
SMT-NMT(+rerank)	66.77	34.49	56.25
NQE	-	-	56.52
PoDA	**71.01**	**37.68**	**60.34**

Table 6: Precision (P), recall (R) and $F_{0.5}$ scores for CoNLL-2014 test set. We only list systems trained on public data. Ge et al. (2018b) reported better performance with additional 4 million non-public sentence pairs.

System (single)	valid	test
MLConv	47.71	51.34
NRL	49.82	53.98
dual-boost	51.35	56.33
PoDA w/o fine-tuning	34.43	36.83
PoDA w/o pre-training	51.57	56.52
PoDA	**53.16**	**59.02**
Ensemble		
MLConv(+rerank)	52.48	57.47
SMT-NMT(+rerank)	-	**61.50**
PoDA	**53.29**	59.48
Human	-	62.38

Table 7: GLEU scores for JFLEG dataset.

and GLEU score from 56.52 to 59.02(+2.50) for JFLEG. By ensembling 4 models initialized with different pre-trained checkpoints and trained with different random seeds, the performance can be further boosted on both datasets (+0.94 for CoNLL-2014 and +0.46 for JFLEG), outperforming the other ensemble models such as "Hybrid SMT-NMT".

We also report the performance of "PoDA w/o fine-tuning" which does not conduct fine-tuning. The $F_{0.5}$ score only reaches 20.86 on CoNLL-2014 dataset and the GLEU score is 36.83 on JFLEG. These results are even worse than the weakest baselines in Table 6 and Table 7. The denoising based pre-training and the GEC task share some similarities in the sense that both of them attempt to convert noisy texts to clean and grammatical texts. However, the poor results of "PoDA w/o fine-tuning" show that PoDA cannot be seen as a simple data augmentation method for GEC. Instead, PoDA learns generic text representations and requires task-specific fine-tuning.

Techniques from previous work for GEC such as language model based rerank (Chollampatt and Ng, 2018a), data augmentation (Ge et al., 2018a), and domain adaptation (Junczys-Dowmunt et al., 2018) can be easily incorporated. A parallel work (Zhao et al., 2019) observes similar gain by combining simpler pre-training strategy and various GEC-specific techniques.

4 Analysis

In the following analysis, we only choose one task (summarization or GEC) to analyze each aspect due to space limitation. Similar conclusions also hold for the other task.

4.1 Linguistic Quality Analysis

In Table 8, we show some generated summaries by PoDA from Gigaword dataset. In the first example, PoDA successfully deletes the relatively unimportant modifier "ivory coast striker", and keeps the picture complete by including both "bremen" and "saint-etienne" in the summary. In the second example, "PoDA w/o pre-training" misses an important date ("first quarter") of the event "economic crisis". Both examples show our model is able to identify the important text snippets in the source sequence and organize them into a short and fluent summary.

More example outputs by PoDA are listed in Appendix.

4.2 Effects of the Number of Pre-training Iterations

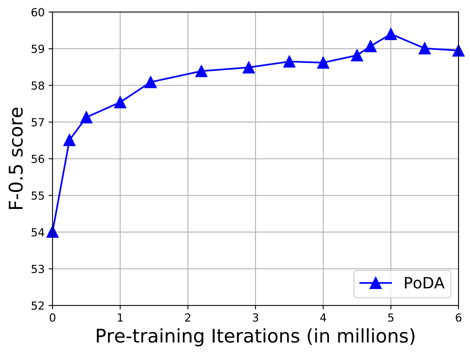

Figure 2: $F_{0.5}$ score on CoNLL-2014 test set with respect to the number of pre-training iterations.

Source	ivory coast striker boubacar sanogo is set to leave werder bremen for french first division side saint-etienne.
Target	sanogo set to sign for saint-etienne
PoDA w/o pre-training	ivory coast striker sanogo set to join saint-etienne
PoDA	sanogo set to leave bremen for saint-etienne
Source	thailand's battered economy should start to bottom out in the first quarter of #### provided the government's efforts are n't neutralized by outside developments, a news report said monday.
Target	economic crisis to bottom out early next year minister says
PoDA w/o pre-training	thai economy expected to start to bottom out in UNK
PoDA	thai economy to start to bottom out in first quarter

Table 8: Examples of generated summaries from Gigaword dataset.

In Figure 2, we show the $F_{0.5}$ score on CoNLL-2014 dataset when the model is initialized with different pre-trained checkpoints. Though the $F_{0.5}$ score has some fluctuations due to the random factors in training neural networks and the limited size of the test set, the overall trend is very clear: the $F_{0.5}$ score first improves greatly and then keeps a slow improvement after about 1 million iterations, from 54 at the very beginning to 59 after convergence.

4.3 Effects of Pre-trained Encoder and Decoder

To show the effectiveness of the pre-trained encoder and decoder, we train the model by only using the encoder-side pre-trained parameters ("w/o pre-trained decoder") or decoder-side pre-trained parameters ("w/o pre-trained encoder") We do not compare with pre-trained encoder from BERT or pre-trained decoder from OpenAI GPT, mainly because the corresponding model capacity, tokenization and text corpora used for pre-training are very different.

System	P	R	$F_{0.5}$
Fully pre-trained	**70.10**	**36.88**	**59.40**
w/o pre-trained encoder	66.14	34.67	55.98
w/o pre-trained decoder	66.62	36.10	56.98
w/o pre-training	65.63	31.62	54.01

Table 9: Ablations for pre-trained encoder and decoder on CoNLL-2014 test set.

Table 9 shows that the performance degrades by a large margin if the network is only partially pre-trained. The pre-trained encoder (-3.42 drop in $F_{0.5}$) is more important than the pre-trained decoder (-2.42 drop in $F_{0.5}$).

4.4 Effects of Dataset Size

We also conduct ablations in few-shot learning settings to see how the performance changes when the model only accesses a small percentage of labeled data. We randomly sample 10^3 to 10^5 training examples from the Gigaword dataset and train "PoDA w/o pre-training" and PoDA (with pre-training) using exactly the same hyperparameters.

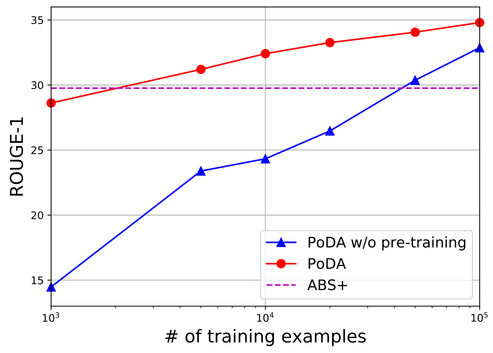

Figure 3: ROUGE-1 on Gigaword test set with respect to the number of training examples. ABS+ is a baseline method from Rush et al. (2015) using attention. The x-axis is in log scale.

Figure 3 shows the ROUGE-1 score on Gigaword test set. With only 10^3 training examples, PoDA reaches a reasonably good performance comparable to ABS+ (an attention-based system trained on nearly 4 million examples). With more labeled data available, the performance gap between "PoDA w/o pre-training" and PoDA slowly decreases from 15 to 2 in Figure 3. However, the pre-training still helps even when the models are trained on the full dataset (shown in Table 4).

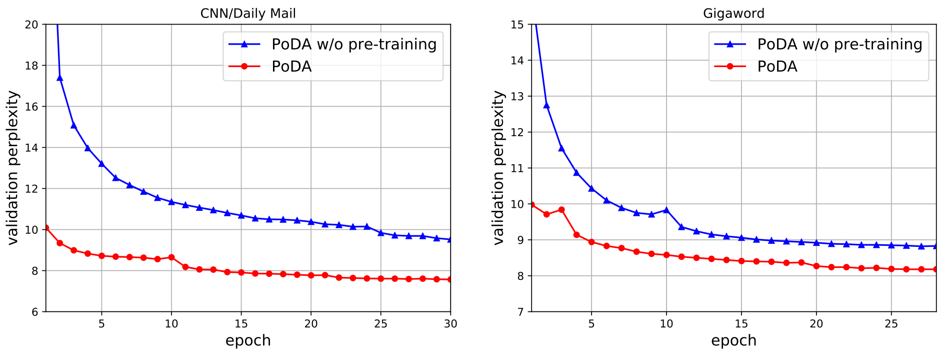

Figure 4: Validation perplexity with respect to the training epochs on CNN/Daily Mail and Gigaword datasets. Perplexity is related to the vocabulary size, so the values here are not comparable with previous work.

4.5 Convergence Analysis

Pre-training can not only achieve better final performance but also helps the model converge faster. In Figure 4, we show the validation perplexity after each training epoch for both "PoDA w/o pre-training" and PoDA.

We can clearly see that the validation perplexity of PoDA is consistently lower than that of "PoDA w/o pre-training", especially at the first few epochs. After 5 epochs, PoDA can arrive at a validation perplexity that "PoDA w/o pre-training" usually takes 30 or more epochs for both CNN/Daily Mail and Gigaword datasets. This nice property is particularly helpful when the computational resources are limited. Other pre-training methods such as BERT also demonstrate similar behaviors.

5 Related Work

Network Pre-training The idea of pre-training neural networks dates back to the early days of deep learning. Bengio et al. (2007) proposed layer-wise pre-training for deep belief networks (DBN) to tackle the difficulty of training deep neural networks based on a reconstruction objective. (Erhan et al., 2010; Dahl et al., 2012) showed the effectiveness of pre-training for tasks such as speech recognition. In the area of computer vision, using ImageNet pre-trained models have become a standard practice. In NLP community, using pre-trained word embeddings is the most popular way to transfer knowledge from the unlabeled corpus. There are also work on semi-supervised sequence learning (Dai and Le, 2015; Peters et al., 2017) attempting to incorporate language modeling as an auxiliary task. Recently, several pre-training methods based on language models are presented, such as ELMo (Peters et al., 2018), OpenAI GPT (Radford et al., 2018), BERT (Devlin et al., 2018), XLM (Lample and Conneau, 2019) etc. The combination of more compute, larger model capacity and large-scale text corpora lead to significant improvements on NLP benchmarks (Wang et al., 2018).

Autoencoders have long been used for representation learning of images (Vincent et al., 2010) and text (Li et al., 2015). However, precisely reconstructing the clean input is probably too easy for high-capacity models. Sparse autoencoders (Deng et al., 2013), contractive autoencoders (Rifai et al., 2011), and denoising autoencoders (Vincent et al., 2010) are several popular variants. Denoising autoencoders (DA) are shown to be able to learn better representations for downstream tasks (Vincent et al., 2010, 2008; Hill et al., 2016). Freitag and Roy (2018) use seq2seq DAs for unsupervised natural language generation in dialogue, and (Kim et al., 2018) propose to improve the quality of machine translation with DAs.

Text Generation covers a wide spectrum of NLP tasks, including machine translation (Wu et al., 2016), summarization (See et al., 2017), response generation (Vinyals and Le, 2015), paraphrase generation, grammatical error correction etc. Early studies on text generation mainly adopt template-based (Reiter and Dale, 2000) or example-based (Watanabe and Takeda, 1998) methods. With the emergence of deep learning for NLP, seq2seq models (Sutskever et al., 2014) become a popular choice for text generation tasks and show better performance in terms of both automatic evaluation metrics and human evaluations (Wu et al., 2016). There are also studies focusing on text generation from structured data such as SQL-to-text (Xu et al., 2018). Previous pre-training for text generation is usually done by independently pre-training encoder-side or decoder-side language models (Ramachandran et al., 2016). Concurrent to our work, Edunov et al. augment encoder representation with ELMo-style models, MASS (Song et al., 2019) masks continuous text fragments for pre-training, and UNILM (Dong et al., 2019) proposes to pre-train for both language understanding and generation tasks.

6 Conclusion

This paper presents a new transfer learning approach for seq2seq text generation named PoDA. It involves two steps: first, pre-train a customized seq2seq denoising autoencoder on large-scale unlabeled text corpora; then, fine-tune on in-domain labeled data. The pre-training step is independent of downstream tasks and jointly learns both encoder and decoder representations. PoDA is simple, intuitive and doesn't require changing network architecture during the fine-tuning stage. Experiments on several abstractive summarization and grammatical error correction datasets demonstrate that PoDA leads to better performance and faster convergence.

For future work, we would like to validate our model on other tasks such as response generation, explore more effective unsupervised sequence-to-sequence pre-training methods, and handle cross-lingual tasks such as machine translation.

Acknowledgments

We want to thank three anonymous reviewers for their valuable comments, and EMNLP-IJCNLP 2019 organizers for their efforts.

References

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. 2007. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*, pages 153–160.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2018a. Retrieve, rerank and rewrite: Soft template based neural summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 152–161.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018b. Faithful to the original: Fact aware neural abstractive summarization. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

Shamil Chollampatt and Hwee Tou Ng. 2018a. A multilayer convolutional encoder-decoder neural network for grammatical error correction. *arXiv preprint arXiv:1801.08831*.

Shamil Chollampatt and Hwee Tou Ng. 2018b. Neural quality estimation of grammatical error correction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2528–2539.

George E Dahl, Dong Yu, Li Deng, and Alex Acero. 2012. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, 20(1):30–42.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572. Association for Computational Linguistics.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner english: The nus corpus of learner english. In *Proceedings of the eighth workshop on innovative use of NLP for building educational applications*, pages 22–31.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in neural information processing systems*, pages 3079–3087.

Jun Deng, Zixing Zhang, Erik Marchi, and Bjorn Schuller. 2013. Sparse autoencoder-based feature transfer learning for speech emotion recognition. In *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, pages 511–516. IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *arXiv preprint arXiv:1905.03197*.

Sergey Edunov, Alexei Baevski, and Michael Auli. 2019. Pre-trained language model representations for language generation. *arXiv preprint arXiv:1903.09722*.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660.

Mariano Felice, Zheng Yuan, Øistein E Andersen, Helen Yannakoudakis, and Ekaterina Kochmar. 2014. Grammatical error correction using hybrid systems and type filtering. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 15–24.

Markus Freitag and Scott Roy. 2018. Unsupervised natural language generation with denoising autoencoders. *arXiv preprint arXiv:1804.07899*.

Tao Ge, Furu Wei, and Ming Zhou. 2018a. Fluency boost learning and inference for neural grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1055–1065.

Tao Ge, Furu Wei, and Ming Zhou. 2018b. Reaching human-level performance in automatic grammatical error correction: An empirical study. *arXiv preprint arXiv:1807.01270*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*.

Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2018. Near human-level performance in grammatical error correction with hybrid machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 284–290.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 140–149.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *Proceedings of NAACL-HLT*, pages 1367–1377.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 328–339.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching neural grammatical error correction as a low-resource machine translation task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 595–606.

Yunsu Kim, Jiahui Geng, and Hermann Ney. 2018. Improving unsupervised word-by-word translation with language model and denoising autoencoder. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 862–868.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Jiwei Li, Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1106–1115.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Junyang Lin, Xu Sun, Shuming Ma, and Qi Su. 2018. Global encoding for abstractive summarization. *arXiv preprint arXiv:1805.03989*.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6294–6305.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning sns for automated japanese error correction of second language learners. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 147–155.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. 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)*, volume 2, pages 588–593.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 229–234.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The conll-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14.

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.

Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1756–1765.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 2227–2237.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pretraining. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/language-unsupervised/language_ understanding_paper. pdf*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Prajit Ramachandran, Peter J Liu, and Quoc V Le. 2016. Unsupervised pretraining for sequence to sequence learning. *arXiv preprint arXiv:1611.02683*.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. 2011. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 833–840. Omnipress.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2017. Grammatical error correction with neural reinforcement learning. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 366–372.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning*, pages 5926–5936.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

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.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Alex Wang, Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Hideo Watanabe and Koichi Takeda. 1998. A pattern-based machine translation system extended by example-based processing. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 1369–1373. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Ziang Xie, Anand Avati, Naveen Arivazhagan, Dan Jurafsky, and Andrew Y Ng. 2016. Neural language correction with character-based attention. *arXiv preprint arXiv:1603.09727*.

Kun Xu, Lingfei Wu, Zhiguo Wang, Yansong Feng, and Vadim Sheinin. 2018. Sql-to-text generation with graph-to-sequence model. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 931–936.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. 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 156–165.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 654–663.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

A Supplemental Material

A.1 Implementation of Pointer-Generator Layer

The pointer-generator layer calculates a probability distribution over the union of a fixed vocabulary $\mathbf{V}$ and the words in the input sequence $\{\mathbf{x}_i\}_{i=1}^n$.

For a word w, the probability $p(w)$ can be calculated as follows:

$$\begin{aligned}\boldsymbol{\alpha} &= MultiHeadAttention(\mathbf{h}^t_{dec}, \mathbf{h}_{enc})\\ p_{gen} &= \sigma(\mathbf{W}_1 AttPooling(\mathbf{h}_{enc}, \boldsymbol{\alpha}))\\ p(w) &= p_{gen} p_v(w) + (1 - p_{gen}) \sum_{i:\mathbf{x'}_i = w} \boldsymbol{\alpha}_i\end{aligned} \tag{3}$$

$\mathbf{h}_{enc}$ is the top layer output of the Transformer encoder, and $\mathbf{h}^t_{dec}$ is the output of the Transformer decoder at timestep t. $p_v(w)$ is the standard softmax probability for word w over the vocabulary ($p_v(w) = 0$ if w is an OOV word). $\boldsymbol{\alpha}$ is the copy attention over the input sequence, *AttPooling* is the attentive pooling of encoder outputs $\mathbf{h}_{enc}$ with distribution $\boldsymbol{\alpha}$, and p_{gen} denotes the probability of generating from the fixed vocabulary. $p(w)$ is a linear interpolation of the generation and copying probabilities.

Source	I do think there is difference in it and I believe many of us will agree.
Target	I do think there is **a difference** and I believe many of us will agree.
PoDA w/o pre-training	I do think there is **difference in it** and I believe many of us will agree.
PoDA	I do think there is **a difference in it** and I believe many of us will agree.
Source	Almost all students and young adults possess the Facebook or Twitter account.
Target	Almost all students and young adults possess **a Facebook or Twitter account**.
PoDA w/o pre-training	Almost all students and young adults possess **Facebook or Twitter accounts**.
PoDA	Almost all students and young adults possess **a Facebook or Twitter account**.

Table 10: Examples of corrected sentences from CoNLL-2014 dataset. The important text snippets are highlighted with bold font.

Source	But, on the contrary, he argues that fluoride also some disadvantage.
Target	But, on the contrary, he argues that fluoride also **has some disadvantages**.
PoDA w/o pre-training	But, on the contrary, he argues that **there are also some disadvantages**.
PoDA	But, on the contrary, he argues that fluoride also **has some disadvantages**.
Source	Such people impressed other people through their strong well and divoution to duty.
Target	Such people impressed others through their **strong will and devotion to duty**.
PoDA w/o pre-training	Such people impressed other people through their **strong and divoution to duty**.
PoDA	Such people impressed other people through their **strong will and devotion to duty**.

Table 11: Examples of corrected sentences from JFLEG dataset. The important text snippets are highlighted with bold font.

Input	Output samples
the best university in the world.	Harvard is considered the 10th best university in the world. I believe that's the best university in the world. Nevada offers the best continuing education in the whole world. Greek is becoming the best university in the world.
The meaning of life is.	The meaning of daily life is unclear. "The real meaning of daily life is peaceful." The underlying meaning of your life is lost forever. The immediate meaning of our political life is undeniable.

Table 12: Some input-output samples by our pre-trained PoDA models. We input some incomplete sentences and output the generated sentences by sampling the output distribution of pre-trained PoDA models at each timestep. We can see that PoDA successfully transforms the inputs to coherent and grammatical sentences, though the statements entailed by these output sentences are not always correct.