ERIE-GEN: An Enhanced Multi-Flow Pre-training and Fine-tuning Framework for Natural Language Generation

Dongling Xiao*, Han Zhang*, Yukun Li, Yu Sun, Hao Tian, Hua Wu and Haifeng Wang
Baidu Inc., China
{xiaodongling, zhanghan17, liyukun01, sunyu02, tianhao, wu_hua, wanghaieng}@baidu.com

Abstract

Current pre-training works in natural language generation pay little attention to the problem of exposure bias on downstream tasks. To address this issue, we propose an enhanced multi-flow sequence-to-sequence pre-training and fine-tuning framework named ERIE-GEN, which bridges the discrepancy between training and inference with an infilling generation mechanism and a noise-aware generation method. To make generation closer to human writing patterns, this framework introduces a span-by-span generation flow that trains the model to predict semantically-complete spans consecutively rather than predicting word by word. Unlike existing pre-training methods, ERIE-GEN incorporates multi-granularity target sampling to construct pre-training data, which enhances the correlation between encoder and decoder. Experimental results demonstrate that ERIE-GEN achieves state-of-the-art results with a much smaller amount of pre-training data and parameters on a range of language generation tasks, including abstractive summarization (Gigaword and CNN/DailyMail), question generation (SQuAD), dialogue response generation (Persona-Chat) and generative question answering (CoQA). The source codes and pre-trained models have been released at https://github.com/PaddlePaddle/ERNIE.

1 Introduction

Pre-trained on large-scale unlabeled text corpora and fine-tuned on downstream tasks, self-supervised representation models such as GPT [Radford et al., 2018], BERT [Devlin et al., 2019] and UniLM [Yang et al., 2019b] have achieved remarkable improvements in natural language understanding (NLU). Different from encoder-only pre-training like BERT or decoder-only pre-training like GPT, natural language generation (NLG) relies on the sequence to sequence generation framework (seq2seq) which consists of a bidirectional encoder and a unidirectional decoder. Current pre-training works in NLG such as MASS [Song et al., 2019] and UniLM [Dong et al., 2019] mainly focus on jointly pre-training encoder and decoder on different self-supervised tasks. However, these works pay little attention to the exposure bias issue [Ranzato et al., 2016], a major drawback of teacher-forcing training. This issue is due to the fact that groundtruth words are used during training, while generated words, whether predicted correctly or not, are used for inference where mistakes tend to accumulate. To alleviate this issue, we present ERIE-GEN, an enhanced multi-flow seq2seq training framework characterized by a carefully-designed Multi-Flow Attention architecture based on Transformer [Vaswani et al., 2017], as illustrated in Figure 2. ERIE-GEN incorporates a novel infilling generation mechanism and a noise-aware generation method into pre-training and fine-tuning, which is proved to be effective through experiments in §4.3.

- **Infilling generation.** Instead of using last groundtruth word in training or last generated word in inference, we adopt an inserted artificial symbol [ATTN] along with its position to gather history contextual representations at each step in both training and inference, which diverts model’s attention away from last word and coerces it into focusing on all former representations, thus alleviating negative influence of previous mistakes to subsequent generation, as shown in Figure 1(b).

- **Noise-Aware generation.** We corrupt the input target se-
sequence by randomly replacing words to arbitrary words in the vocabulary. This setup, despite its simplicity, proves to be an effective way to make the model be aware of mistakes in training, so that the model is able to detect mistakes and ignore them during inference.

Moreover, in light of the fact that entities, phrases and sentences in human writing are organized in a coherent manner, we incorporate a **span-by-span generation task** into ERNIE-GEN as a new generation flow to train the model to predict semantically-complete spans consecutively rather than predicting word by word as traditional models do. This task is implemented through the infilling generation mechanism in parallel with an infilling-based word-by-word generation flow to facilitate convergence in training, as shown in Figure 1b.

In addition, as shown in Figure 1(c-d), recent pre-training works for NLG like UStLM and MASS only sample a single continuous segment as target sequence. However, this sampling method compromises the correlation between encoder and decoder when it comes to pre-training of long texts (typically 512 words), given that adjacent segments are often relevant semantically. ERNIE-GEN adopts a **multi-granularity target fragments** sampling strategy to force decoder to rely more on the encoder representations other than the previous generated words, thus enhancing the correlation between encoder and decoder, as shown in Figure 1e.

Empirically, ERNIE-GEN is particularly effective and achieves state-of-the-art results on a range of NLU tasks including abstractive summarization (Gigaword and CNN/DailyMail), question generation (SQuAD), dialogue response generation (Persona-Chat) and generative question answering (CoQA), utilizing a much smaller amount of pre-training data and parameters.

## 2 Related Work

### Pre-training for NLP Tasks

Recently, pre-training methods have achieved state-of-the-art results in multiple NLU tasks. ELMo [Peters et al., 2018] pre-trains two unidirectional language models (LMs) with forward and backward direction respectively to feature downstream tasks. GPT utilizes an adjusted Transformer [ Vaswani et al., 2017] to learn a forward LM and then fine-tunes the forward LM on supervised datasets. BERT proposes a masked language modeling (MLM) task to learn deep bidirectional representations. Nevertheless, above methods are usually implemented by just one encoder or decoder, which is less effective in encoder-decoder based generation tasks, thus several works have preliminarily explored the pre-training towards NLG by incorporating BERT’s MLM into the seq2seq framework and shown excellent performance on a range of generation tasks. MASS masks a consecutive fragment (50%) of the input sentence with [MASK] symbols to predict. UnLM masks several words in the input sequence which is a pair of segments for encoder and decoder, and then predicts the masked words in accordance with BERT’s MLM.

### Exposure Bias Issue

NLG tasks suffer from the exposure bias which is caused by teacher-forcing training. To address such issue, RNN-based variational autoencoders (VAEs) are leveraged in [Yang et al., 2019a; Wang et al., 2019], whereas it requires inference for both posterior and prior distribution. Reinforcement learning is also adopted to text generation against exposure bias issue [Ranzato et al., 2016; Wang et al., 2018], which is, however, inefficient during training because of the word-by-word sampling procedure. These methods are inefficient and less practical for pre-training that relies on large-scale unlabeled text corpora.

### Span-level Pre-training

[Sun et al., 2019; Sun et al., 2020; Joshi et al., 2019] verify that predicting spans reaches substantially better performance on NLU tasks. Meanwhile, inspired by characteristics of human expression, we hope the model have the foresight to generate a semantically-complete span at each step rather than a word. Consequently, a span-by-span generating task is proposed to make the model capable of generating texts more human-like.

## 3 Proposed Framework

Built on infilling generation mechanism, ERNIE-GEN adopts a Multi-Flow Attention architecture to train the model on word-by-word and span-by-span generation tasks in parallel. In this section, we describe ERNIE-GEN according to the training process shown in Figure 2.

### 3.1 Multi-Granularity Target Fragments

Given an input sequence \( S = \{s_1, ..., s_n\} \), we first sample a length distribution \( D_i \) from a distribution set \( D = \{D_1, ..., D_D\} \) with probability \( p_i \) for target fragments, and then select fragments according to \( D_i \) in \( S \) iteratively until the fragment budget has been spent (e.g. 25% of \( S \)). We denote \( S_{j}^{i} \) as the \( j \)-th fragment which is sampled in length distribution \( D_i \). Sampled fragments are then removed from \( S \) and stitched together to form target sequence \( T = \{T_1, ..., T_k\} = \{S_1^{i}, ..., S_k^{i}\} \). We denote \( S' \) as the left input sequence after removing sampled fragments. ERNIE-GEN performs pre-training by predicting the fragmented target sequence \( T \) and minimizing the negative log likelihood:

\[
\mathcal{L}(\theta; S, D_i) = -\log P(T|S', D_i; \theta) = \sum_{j=1}^{k} \log P(T_j|T_{<j}, S', D_i; \theta).
\]

where the target sequence \( T \) is sorted by the positions of sampled fragments. For each fragment \( T = \{t_1, ..., t_{|T|}\} \) in \( T \), we have \( P(T) = \prod_{j=1}^{|T|} P(t_j|t_{<j}) \).

```
Following preliminary trials, we set a hyperparameter $\gamma = 0.25$, which denotes the ratio of length of all fragments to that of the input sequence $S$. Besides, we introduce two uniform distributions $D = \{U(1, 4), U(4, 32)\}$ with probability of 0.4 and 0.6 respectively to sample fragments, which aims to learn representations from different perspectives. On the one hand, short fragments benefit learning of semantic relation between words; on the other hand, longer fragments help to memorize sentence-level expressions.

3.2 Noise-Aware Generation

To train a generation model which can detect the false prediction and mitigate its impact on subsequent generation, we corrupt the groundtruth sequence $T$ with a procedure where words are being replaced randomly, and the corrupted $T$ is represented as $T'$. There are two hyperparameters, $\rho_p$ and $\rho_f$, denoting the noising rate in pre-training and fine-tuning respectively.

3.3 Architecture: Multi-Flow Attention

Formally, given a source sequence $S = \{s_1, ..., s_n\}$, a noised target sequence $T' = \{t_1, ..., t_m\}$, we denote the inference of seq2seq network based on shared Transformer as follows:

$$s_i^{(l+1)} \leftarrow \text{MH-Attn}(Q = s_i^{(l)}, KV = S^{(l)}),$$

$$t_i^{(l+1)} \leftarrow \text{MH-Attn}(Q = t_i^{(l)}, KV = S^{(l)}, t_{\leq i}^{(l)}),$$

where $Q, K, V$ denote the query, key, and value in Multi-Head attention [Vaswani et al., 2017]. $s_i^{(l)}$ and $t_i^{(l)}$ indicate the $i$-th vector representations of the $l$-th layer of Multi-Head Attention for the encoder and the decoder respectively, $[\cdot]$ denotes the concatenation operation. In this work, we call the above procedure the Contextual Flow.

Word-by-word Generation Flow. Based on infilling generation mechanism, this flow utilizes an inserted [ATTN] symbol to gather history representations word by word (see Figure 1b). To facilitate this process, we place all inserted [ATTN] together to construct an artificial symbol sequence $A_W = \{[\text{ATTN}]_1, ..., [\text{ATTN}]_m\}$ which has the same length as $T'$, as shown in Figure 3b. To be specific, the word-by-word generation flow is updated as follow:

$$a_i^{(l+1)} \leftarrow \text{MH-Attn}(Q = a_i^{(l)}, KV = [S^{(l)}, t_{\leq i}^{(l)}, a_i^{(l)}]).$$

(3)

where $a_i^{(l)}$ indicates the $i$-th vector representation of the $l$-th layer for the artificial symbol sequence $A_W$.

Span-by-span Generation Flow. Different from word-by-word generation flow, span-by-span flow uses [ATTN] symbols to predict spans consecutively, as shown in Figure 3c. Formally, given a list of span boundaries $B = \{b_1, ..., b_B\}$, we conduct the span-by-span generation flow as:

$$a_{j+1}^{(l+1)} \leftarrow \text{MH-Attn}(Q = a_j^{(l)}, KV = [S^{(l)}, t_{\leq b_j}^{(l)}, a_j^{(l)}]).$$

(4)

where $j \in [b_i, b_{i+1}]$, and $a_j^{(l)}$ denotes the $(j - b_i)$-th vector representation of the $i$-th span. Essentially, the model is trained to predict a whole span $\{b_{i+1}, ..., b_{i+1+1}\}$ with the same history context $[S, t_{\leq b_i}]$. Instead of randomly sampling spans, we prefer sampling spans with semantical information and knowledge. Specifically, we consider the following two steps to sample spans consecutively in $T'$:

- Firstly, we implement a T-test to compute t-statistic scores of all bigrams and trigrams, which is based on an initial hypothesis $H_0$: a random span of $n$ arbitrary words $w = \{w_1, ..., w_n\}$ with probability $p'(w) = \prod_{i=1}^n p(w_i)$ cannot be a statistical $n$-gram. The t-statistic score is calculated by

$$\frac{(p(w) - p'(w))}{\sqrt{\sigma_N^2 / N}},$$

where $p(w) = \frac{\text{Count}(w)}{N}$ and $\sigma^2 = p(w)(1 - p(w))$.

- Secondly, we implement a multi-head attention flow to predict the span with the highest t-statistic score, which is denoted as $[\text{ATTN}]_{i+1}$.
\( p(w) \)), indicating the statistic probability and the standard deviation of \( w \) respectively. \( N \) denotes the total number of \( n \)-grams appearing in the training data. According to the \( t \)-statistic scores, we select the top 200,000 bigrams, top 50,000 trigrams and all unigrams to construct a specific vocabulary of spans, which is represented as \( V_{span} \).

- Secondly, we search the trigram, bigram and unigram in order, starting with current word until a span \( (n\text{-gram}, n \leq 3) \) is retrieved in \( V_{span} \).

**Multi-Flow Attention.** To integrate the word-by-word generation flow and span-by-span generation flow, we apply them in parallel with a shared contextual flow by leveraging the multi-flow attention architecture, as Figure 3a describes. The multi-flow attention is computed as:

\[
\begin{align*}
X^{(l+1)} & \leftarrow \text{MH-Att} (Q=X^{(l)}, KV=X^{(l)}, MC) \\
A^{(l+1)}_W & \leftarrow \text{MH-Att} (Q=A^{(l)}_W, KV=[X^{(l)}, A^{(l)}_W], MW) \\
A^{(l+1)}_S & \leftarrow \text{MH-Att} (Q=A^{(l)}_S, KV=[X^{(l)}, A^{(l)}_S], MS)
\end{align*}
\]

where \( X \) denotes the concatenation of \( S \) and \( T \), \( X^{(l)} \) is the vector sequence of the \( l \)-th layer for the contextual flow. \( A^{(l)}_W, A^{(l)}_S \) are vector sequences of the \( l \)-th layer for the word-by-word and span-by-span generation flow respectively. As shown in Figure 3d, attention mask matrix \( M \) determines whether query and key can attend to each other by modifying the attention weight \( W = \text{softmax}(QK^T / \sqrt{d_k} + M) \) [Vaswani et al., 2017]. Specifically, \( M \) is assigned as:

\[
M_{ij} = \begin{cases} 
0, & \text{can be attended} \\
-\infty, & \text{prevent from attending}
\end{cases}
\]

While training, we add the loss of the word-by-word and span-by-span generation flow with an coefficient \( \lambda \):

\[
\mathcal{L}(T) = \lambda \mathcal{L}_{Word}(T) + (1 - \lambda) \mathcal{L}_{Span}(T)
\]

\[
= -\lambda \log P(T|A^{(L-1)}_W) - (1 - \lambda) \log P(T|A^{(L-1)}_S),
\]

where \( T \) indicates the unnoised target sequence, and \( \mathcal{L}(\cdot) \) denotes the cross entropy loss function. In detail, we set \( \lambda = 0.5 \) and \( \lambda = 1.0 \) respectively in pre-training and fine-tuning.

### 3.4 Inference: Infilling Decoding

During inference, the target sequence \( T \) is unknown, we insert symbol \([\text{ATTN}]\) step by step to gather the representation of history context instead of preparing an artificial symbol sequence \( A \) in advance. Meanwhile, for the purpose of efficiency, we need to drop the inserted \([\text{ATTN}]\) after inference at each step, as detailed in Figure 4.

### 4 Experiments

In this section, we compare our ERNIE-GEN with previous works and conduct several ablation experiments to assess the performance of proposed methods in §3.

### 4.1 Pre-training and Implementation

Analogous to BERT and UniLM, ERNIE-GEN is trained on English Wikipedia\(^1\) and BookCorpus, totaling 16GB.

\(^1\)English Wikipedia version: enwiki-20181101.

The input sequence is lowercased and truncated to a maximum length of 512. We train a base model ERNIE-GEN\(_{BASE}\) \((L=12, H=768, A=12, \text{Total Parameters}=110M)\)\(^2\) and a large model ERNIE-GEN\(_{LARGE}\) \((L=24, H=1024, A=16, \text{Total Parameters}=340M)\) with parameters initialized by BERT\(_{BASE}\) and BERT\(_{LARGE}\) respectively. Specifically, Adam optimizer with \( \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-9} \) is employed. The peak learning rate is 5e-5 with warmup over the first 4,000 steps and linear decay scheduling. The noise rate \( \rho_p \) for pre-training is 0.05. Batches are organized by limiting the maximum number of tokens to 196,608. Pre-training experiments are carried out on PaddlePaddle platforms\(^3\) and Nvidia Tesla V100 GPU. By virtue of float16 mixed precision training, it takes almost 4 days for 400,000 steps to train ERNIE-GEN\(_{BASE}\) while almost 7 days for 450,000 steps to train ERNIE-GEN\(_{LARGE}\).

### 4.2 Fine-tuning on Downstream Tasks

**Abstractive Summarization** aims at generating fluent and concise summaries without being constrained to extracting sub-sequences from the input sentences. We execute experiments on Gigaword dataset [Rush et al., 2015] and CNN/DailyMail dataset [Hermann et al., 2015]. Gigaword dataset contains 3.8M articles extracted from the Gigaword corpus, while CNN/DailyMail dataset consists of 93k articles and 220k articles from the CNN and Daily Mail respectively.

<table>
<thead>
<tr>
<th>Model</th>
<th>Data (P)</th>
<th>Parameters (M)</th>
<th>RG-1 / RG-2 / RG-L</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>10k training samples</strong> : Gigaword 10k</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MASS [Song et al., 2019]</td>
<td>18G</td>
<td>160M</td>
<td>25.03 / 9.48 / 23.48</td>
</tr>
<tr>
<td>UniLM(_{LARGE}) [Dong et al., 2019]</td>
<td>16G</td>
<td>340M</td>
<td>32.96 / 14.68 / 30.56</td>
</tr>
<tr>
<td>ERNIE-GEN(_{BASE})</td>
<td>16G</td>
<td>110M</td>
<td>33.75 / 15.23 / 31.35</td>
</tr>
<tr>
<td>ERNIE-GEN(_{LARGE})</td>
<td>16G</td>
<td>340M</td>
<td>35.05 / 16.10 / 32.50</td>
</tr>
<tr>
<td><strong>Fully 3.8M training samples</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MASS [Song et al., 2019]</td>
<td>18G</td>
<td>160M</td>
<td>37.66 / 18.53 / 34.89</td>
</tr>
<tr>
<td>BERTSHARE [Rothe et al., 2019]</td>
<td>16G</td>
<td>110M</td>
<td>38.13 / 19.81 / 35.62</td>
</tr>
<tr>
<td>UniLM(_{LARGE}) [Dong et al., 2019]</td>
<td>16G</td>
<td>340M</td>
<td>38.45 / 19.45 / 35.75</td>
</tr>
<tr>
<td>PEGASUS(_{GEN}) [Zhang et al., 2019]</td>
<td>730G</td>
<td>568M</td>
<td>38.75 / 19.96 / 36.14</td>
</tr>
<tr>
<td>PEGASUS(_{HARE}) [Zhang et al., 2019]</td>
<td>3.8T</td>
<td>568M</td>
<td>39.12 / 19.86 / 36.24</td>
</tr>
<tr>
<td>ERNIE-GEN(_{BASE})</td>
<td>16G</td>
<td>110M</td>
<td>38.63 / 20.04 / 36.20</td>
</tr>
<tr>
<td>ERNIE-GEN(_{LARGE})</td>
<td>16G</td>
<td>340M</td>
<td>39.25 / 20.25 / 36.53</td>
</tr>
</tbody>
</table>

\(\ast\) We donate the number of layers as \( L \), the hidden size as \( H \) and the number of self-attention heads as \( A \).

\(\ast\) We donate the number of layers as \( L \), the hidden size as \( H \) and the number of self-attention heads as \( A \).

\(\ast\) We donate the number of layers as \( L \), the hidden size as \( H \) and the number of self-attention heads as \( A \).

\(\ast\) We donate the number of layers as \( L \), the hidden size as \( H \) and the number of self-attention heads as \( A \).
Table 1: Hyperparameters of fine-tuning for ERNIE-GEN_{BASE} and ERNIE-GEN_{LARGE}.

<table>
<thead>
<tr>
<th>Model</th>
<th>Data</th>
<th>Params</th>
<th>RG-1 / RG-2 / RG-L</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERTSHARE [Rothe et al., 2019]</td>
<td>16G</td>
<td>110M</td>
<td>39.25 / 18.09 / 36.45</td>
</tr>
<tr>
<td>BERTSUMABS [Liu and Lapata, 2019]</td>
<td>16G</td>
<td>110M</td>
<td>41.72 / 19.39 / 38.76</td>
</tr>
<tr>
<td>MASS [Song et al., 2019]</td>
<td>18G</td>
<td>160M</td>
<td>42.12 / 19.50 / 39.01</td>
</tr>
<tr>
<td>UnlLM_{LARGE} [Dong et al., 2019]</td>
<td>16G</td>
<td>340M</td>
<td>43.33 / 20.21 / 40.51</td>
</tr>
<tr>
<td>T5_{LARGE} [Raffel et al., 2019]</td>
<td>750G</td>
<td>340M</td>
<td>42.50 / 20.68 / 39.75</td>
</tr>
<tr>
<td>T5_{XLARGE} [Raffel et al., 2019]</td>
<td>750G</td>
<td>11B</td>
<td>43.52 / 21.55 / 40.69</td>
</tr>
<tr>
<td>BART_{LARGE} [Lewis et al., 2019]</td>
<td>160G</td>
<td>400M</td>
<td>44.16 / 21.28 / 40.90</td>
</tr>
<tr>
<td>PEGASUS [Zhang et al., 2019]</td>
<td>750G</td>
<td>568M</td>
<td>43.90 / 21.20 / 40.76</td>
</tr>
<tr>
<td>ERNIE-GEN_{BASE}</td>
<td>16G</td>
<td>110M</td>
<td>42.30 / 19.92 / 39.68</td>
</tr>
<tr>
<td>ERNIE-GEN_{LARGE}</td>
<td>16G</td>
<td>340M</td>
<td>44.02 / 21.17 / 41.26</td>
</tr>
</tbody>
</table>

Table 2: Evaluation results on CNN/DailyMail. C4 and HugeNews are two massive datasets of 750G and 3.8T respectively.

The results on Gigaword task with two scales (10k and 3.8M) are presented in Table 2, and the fine-tuning settings are shown in Table 1. On the low-resource task (Gigaword 10k), ERNIE-GEN_{LARGE} yields a gain of +1.94 ROUGE-L compared with UnlLM_{LARGE}. On the full training set, ERNIE-GEN_{LARGE} creates the state-of-the-art results, outperforming various previous methods. Specifically, ERNIE-GEN_{BASE} outperforms PEGASUS (568M and 750G) by using only 110M parameters and 16G training data.

Table 3 shows the performance on CNN/DailyMail. With a similar amount of pre-training data and parameters, ERNIE-GEN_{BASE} outperforms MASS by +0.67 ROUGE-L scores. Fairly compared with UnlLM_{LARGE}, ERNIE-GEN_{LARGE} obtains substantial gain of +0.73 ROUGE-L scores. Meanwhile, in spite of small pre-training data and parameters, our large model also achieves state-of-the-art result on ROUGE-L and comparable performance on ROUGE-1/2.

**Question Generation** is to generate a question according to a given input passage and a corresponding answer. We evaluate on the SQuAD 1.1 dataset [Rajpurkar et al., 2016] for question generation task (called SQuAD QG). Following UnlLM, we redistribute the original dataset into a new training set and testing set with the original development set unchanged. We also conduct experiment with the reversed dev/test split as [Zhao et al., 2018] indicates. In Table 4, we present the results of ERNIE-GEN and several previous works. Again, ERNIE-GEN outperforms UnlLM_{LARGE} and achieves a new state-of-the-art result on question generation by giving +1.82 BLEU-4 scores.

**Generative Question Answering / Dialogue Response** in multi-turn conversations are challenging because of complex background knowledge and diverse utterances. We conduct an experiment on Persona-Chat dataset [Zhao et al., 2018] to generate responses according to given multi-turn conversations and persona profile. Table 5 shows that ERNIE-GEN outperforms current task-specific pre-training model on dialogue generation. Beside, we also execute an experiment on CoQA dataset [Reddy et al., 2019] to generate free-form answers for input questions and conversations. As shown in Table 6, our generative question answering model works considerably better than early works by +2.0 F1-scores.

Table 6: Generative question answering results on the development set of CoQA.

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU-1/2</th>
<th>Distinct-1/2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIC [Bao et al., 2020]</td>
<td>40.5 / 32.0</td>
<td>0.019 / 0.113</td>
</tr>
<tr>
<td>PLATO_{w/o late} [Bao et al., 2020]</td>
<td>45.8 / 35.7</td>
<td>0.012 / 0.064</td>
</tr>
<tr>
<td>PLATO [Bao et al., 2020]</td>
<td>40.6 / 31.5</td>
<td>0.021 / 0.121</td>
</tr>
<tr>
<td>ERNIE-GEN_{LARGE}</td>
<td>46.8 / 36.4</td>
<td>0.023 / 0.168</td>
</tr>
</tbody>
</table>

Table 4: Question generation results on SQuAD. Models in the upper block and the lower block use different test <-> dev split method.

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU-4</th>
<th>METEOR</th>
<th>ROUGE-L</th>
</tr>
</thead>
<tbody>
<tr>
<td>SemGG [Zhang and Bansal, 2019]</td>
<td>20.76</td>
<td>24.20</td>
<td>48.91</td>
</tr>
<tr>
<td>UnlLM_{LARGE} [Dong et al., 2019]</td>
<td>22.12</td>
<td>25.06</td>
<td>51.07</td>
</tr>
<tr>
<td>ERNIE-GEN_{BASE} (beam size = 1)</td>
<td>22.28</td>
<td>25.13</td>
<td>50.58</td>
</tr>
<tr>
<td>ERNIE-GEN_{LARGE} (beam size = 1)</td>
<td>24.03</td>
<td>26.31</td>
<td>52.36</td>
</tr>
<tr>
<td>ERNIE-GEN_{LARGE} (beam size = 5)</td>
<td>25.40</td>
<td>26.92</td>
<td>52.84</td>
</tr>
</tbody>
</table>

4.3 Ablation Studies

To better understand the importance of each proposed generation methods, we conduct experiments concerning the following two aspects:

- The robustness of infilling generation mechanism and noise-aware generation method against the exposure bias.
- The effectiveness of span-by-span generation task and the complete ERNIE-GEN model.

In Table 8, we compare two ERNIE-GEN_{BASE} variants that are pre-trained with typical generation mechanism and infilling generation mechanism and that generate word by word. Row 1-3 shows that without noising groundtruth texts, infilling generation outperforms typical generation
Figure 5: Results of ablation studies. (a): Comparisons between typical generation and infilling generation on Gigaword 10k and SQuAD QG with different fine-tuning noising rate $\rho_f$. (b): Noising Analysis, average attention weights of source words, unnoised target words and noised target words for diverse fine-tuning noising rate $\rho_f$. (c): Ablation study on Gigaword 10k, the x-axis shows fine-tuning epochs.

<table>
<thead>
<tr>
<th># Fine-tuning method</th>
<th>Gigaword 10k</th>
<th>CNN/DailyMail 10k</th>
<th>SQuAD QG</th>
<th>Gigaword 10k</th>
<th>CNN/DailyMail 10k</th>
<th>SQuAD QG</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 ERNIE-GEN</td>
<td>32.98 / 14.67 / 30.51</td>
<td>39.25 / 16.70 / 36.65</td>
<td>39.71 / 17.55 / 37.18</td>
<td>23.34 / 25.54 / 51.30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 - noise-aware</td>
<td>33.37 / 15.15 / 31.28</td>
<td>39.78 / 17.63 / 37.23</td>
<td>39.75 / 17.62 / 37.21</td>
<td>23.37 / 25.56 / 51.32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 - 2 and 3</td>
<td>33.23 / 14.77 / 30.01</td>
<td>39.71 / 17.55 / 37.18</td>
<td>39.71 / 17.55 / 37.18</td>
<td>23.34 / 25.54 / 51.30</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7: Ablation study for ERNIE-GEN_BASE and its variants. Particularly, We set $\rho_p = 0.05$ in pre-training (row 1), while removing the span-by-span generation task (row 3), we set $\rho_p = 0.2$ because the pre-training becomes easier.

Table 8: Results of models pre-trained with typical generation and infilling generation. Tasks in the upper block are fine-tuned without noise, while the others are fine-tuned with noise-aware generation. 

Table 8: Results of models pre-trained with typical generation and infilling generation. Tasks in the upper block are fine-tuned without noise, while the others are fine-tuned with noise-aware generation. Tasks in the upper block are fine-tuned without noise, while the others are fine-tuned with noise-aware generation.

5 Conclusions

We present an enhanced multi-flow seq2seq pre-training and fine-tuning framework named ERNIE-GEN for language generation, which incorporates an infilling generation mechanism and a noise-aware generation mechanism to alleviate the exposure bias. Besides, ERNIE-GEN integrates a new span-by-span generation task to train the model to generate texts like human writing, which further improves the performance on downstream tasks. Through extensive experiments, ERNIE-GEN achieves state-of-the-art results on a range of NLG tasks.

Acknowledgments

This work was supported by the National Key Research and Development Project of China (No. 2018AAA0101900).
References


