MLAE: Encoder-decoder Pre-training with Non-autoregressive Modeling

Anonymous ACL submission

Abstract

Encoder-decoder pre-training has proven successful in natural language processing. Most of the existing works on encoder-decoder pre-training are based on the autoregressive architecture. In this paper, we introduce MLAE, a new pre-training framework based on a non-autoregressive encoder-decoder architecture. It behaves like a masked autoencoder and reconstructs the masked language tokens in a non-autoregressive manner. Our model combines the best of two worlds: the advantages of the encoder-only models on the understanding tasks and the capabilities of the autoregressive encoder-decoder on the generation tasks. Extensive experiments show that MLAE outperforms strong baselines on various benchmarks, including language understanding, autoregressive generation, as well as non-autoregressive generation.[1]

1 Introduction

Recent years have witnessed a trend towards large-scale pre-trained language models (Devlin et al., 2019; Liu et al., 2019; Joshi et al., 2020; Song et al., 2019; Raffel et al., 2020; Lewis et al., 2020; Qi et al., 2021). The pre-trained models significantly improve the performance on downstream tasks. From the perspective of the model architecture, we can classify current language models into three categories: non-autoregressive (NAR) encoder (Devlin et al., 2019; Liu et al., 2019), autoregressive (AR) decoder (Radford et al., 2018), and encoder-decoder (Raffel et al., 2020; Lewis et al., 2020). AR decoders (e.g., GPT) show impressive performance of in-context learning, while the others are better at fine-tuning on the downstream tasks.

The NAR encoders, or the encoder-only models (e.g., BERT, RoBERTa, etc) are superior on natural language understanding (NLU) tasks, such as text classification and question answering. However, due to the lack of pre-trained decoder, they can not naturally be fine-tuned on natural language generation (NLG) tasks. Therefore, current works usually adopt the vanilla encoder-decoder architecture (e.g., T5, BART, etc). Although the vanilla encoder-decoder pre-training provides a pre-trained decoder, its AR decoder undermines the ability of the encoder, which hurts the quality of generation.

In this paper, we introduce a simple yet effective pre-training framework based on NAR encoder-

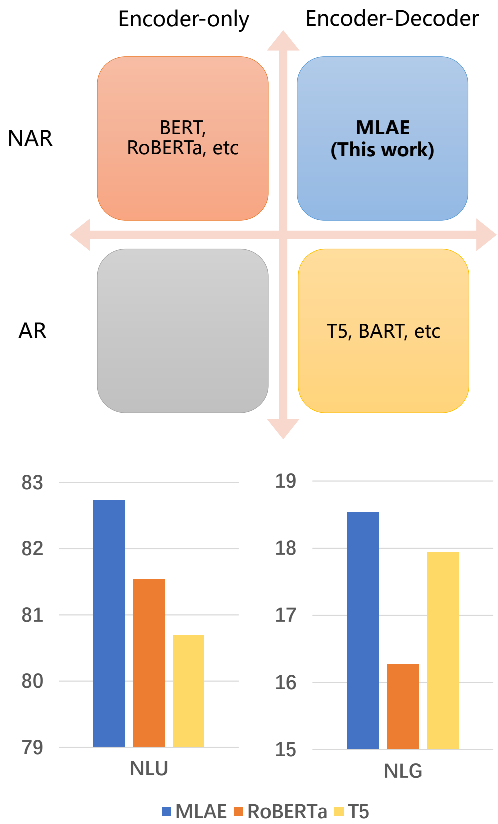

Figure 1: **Top**: the architectures of the mainstream pre-trained language models. **Bottom**: the average score on the NLU tasks(GLUE benchmark) and ROUGE-2 on the NLG tasks(XSum dataset) for MLAE, RoBERTa and T5.

[1]We will release the code for reproducibility.

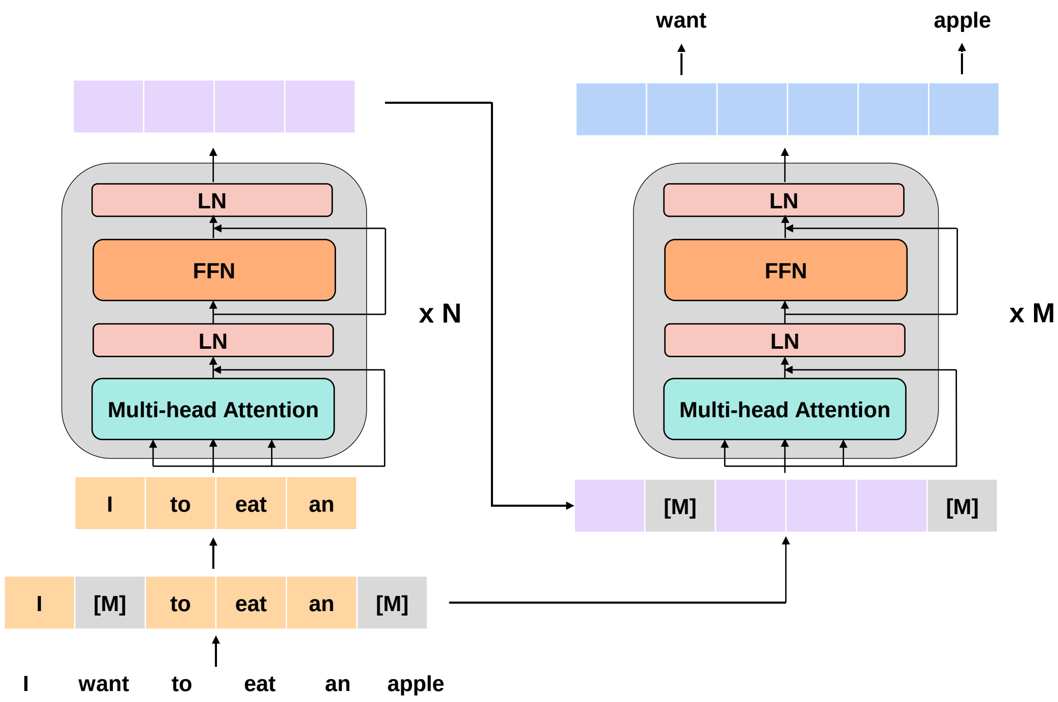

Figure 2: Overview for the pre-training of MLAE. The encoder only processes the unmasked tokens. After the encoder, the masked tokens are concatenated with latent representations. A light-weight decoder non-autoregressively reconstructs the masked tokens from a full set of tokens.

decoder architecture, named as **M**asked **L**anguage **A**uto**E**ncoder (**MLAE**). The proposed MLAE not only provides a pre-trained decoder for generation tasks but also significantly improves the encoder's ability. As shown in Figure 1, the encoder part of MLAE is more powerful than the counterpart of the vanilla encoder-decoder, even the encoder-only model. Besides, we introduce scheduled masking to bridge the gap between pre-training and generation for MLAE. Above all, our model has shown superior performance on both NLU and NLG tasks. Especially for NAR generation, MLAE outperforms strong baselines by an improvement of 3.03 ROUGE-2 on XSum dataset.

2 Background

We begin with the observation: while pre-trained encoder-decoder models are good at language generation, they trade off the encoders' ability. During pertaining, the encoder maps the unmasked tokens into latent representations, while the decoder autoregressively reconstructs the masked tokens. However, this architecture makes the model more rely on the AR decoder to generate target tokens rather than the encoder. Therefore, the encoders pre-trained through AR encoder-decoder models are generally weaker than the encoder-only model (Liu et al., 2019).

To verify the above observation, we pre-train a 12L RoBERTa and a 12L-4L T5 with the combination of English-Wikipedia and the BookCorpus. The performance on NLU tasks is a good metric for the ability of encoder. We thus evaluate RoBERTa and T5 encoder on the large tasks of GLUE benchmark (Wang et al., 2019) and SQuAD2 dataset (Rajpurkar et al., 2018). We use the same fine-tuning method and hyper-parameters for a fair comparison. Table 1 demonstrates that RoBERTa outperforms T5 encoder on both two datasets. It shows that although the AR encoder-decoder pre-training provides a pre-trained decoder to benefit generation tasks, the ability of its encoder is undermined by AR decoder.

In this work, inspired by the recent success of Masked Autoencoder (MAE), we explore a way to combine the best of two worlds: the architecture of encoder-decoder models and the NAR objective of encoder-only models.

3 MLAE

In this section, we first introduce the model architecture of MLAE. Then we demonstrate the fine-tuning methods for NLU and NLG tasks, including two generation paradigms: AR and NAR generation, respectively.

3.1 Pre-training

MLAE is based on asymmetric NAR Transformer encoder-decoder. Figure 2 shows the overview for pre-training of our model. Similar to the MAE (He et al., 2021), it uses a decoder to reconstruct the masked tokens at the corresponding positions during pre-training. Given blocks of sentences, we randomly mask a portion of input tokens. The unmasked tokens are first processed by a series of Transformer blocks, including a self attention layer followed by a feed forward layer. After the encoder, the masked tokens are concatenated with latent representations, then feed into a light-weight decoder. The decoder is designed to reconstruct the masked tokens bidirectionally from a full set of tokens.

Compared with vanilla encoder-decoder models (Raffel et al., 2020; Lewis et al., 2020), the decoder of MLAE utilizes bidirectional information to reconstruct the masked tokens. Besides, we adopt an asymmetric encoder-decoder architecture: the decoder has less layers than the encoder. Both of them prevent the model more rely on the decoder, thus provide a more challenging task for the encoder.

3.2 Fine-tuning for Understanding Tasks

For NLU tasks, we directly use the encoder part of MLAE as feature extractor. After the encoder, we add a linear layer followed by softmax classifier as the task layer. The encoder generates latent representations from source sentences. Then the task layer projects the representation corresponding to [EOS] into the label space.

3.3 Fine-tuning for Generation Tasks

For NLG tasks, we introduce two fine-tuning methods for MLAE, which are UniLM-style (Dong et al., 2019) fine-tuning and Seq2seq-style fine-tuning.

For UniLM-style fine-tuning, we modify the attention mask as Seq2Seq mask in each self-attention layer of the decoder. With Seq2Seq mask, a token in the source segment can attend to all the tokens within segment, while a token in the target segment can only attend to the leftward tokens. The encoder first generates latent representations from source sentences. After the encoder, we concatenate the latent representations with target tokens, then feed them into the decoder. The decoder autoregressively predicts target tokens conditioned on the leftward tokens.

For Seq2seq-style fine-tuning, we insert cross-attention layers into each layer of MLAE decoder, then fine-tune it as the vanilla encoder-decoder. The decoder autoregressively generates target tokens conditioned on the encoder output through cross-attention layers. To make full use of pre-trained modules, we initialize each cross-attention layer by the weights of self-attention layer.

The experiments in Section 5.1 demonstrate that Seq2seq-style fine-tuning is better for MLAE on generation task. We achieve greater gains with more pre-trained modules of MLAE.

Algorithm 1 Scheduled Masking

Input: source sentence $[x_1, x_2, ..., x_n]$ and target sentence $[y_1, y_2, ..., y_n]$, initial mask ratio m_0 and maximum training updates T.
for $t = 0$ to T **do**
  // feed source sentence into the encoder
  $[h_i]_{i\in[1,n]} \leftarrow Encoder([x_i]_{i\in[1,n]})$
  // Masking scheduler
  $m_t \leftarrow m_0 - \frac{m_0}{T} t$
  $[y_1, ..., M, ..., y_n] \leftarrow Mask([y_i]_{i\in[1,n]}, m_t)$
  // feed masked target sentence into the decoder
  $[y_i^*]_{i\in[1,n]} \leftarrow Decoder([y_1, ..., M, ..., y_n], [h_i]_{i\in[1,n]})$

  $Loss = \sum_i f(y_i^*, y_i)$
end for

3.4 Scheduled Masking

There are two major gaps between the MLM task and the AR generation. During the pre-training, MLAE decoder is designed to bidirectionally reconstruct the masked tokens from latent representations generated by the encoder. However, in the fine-tuning for AR generation, the decoder aims to predict the target tokens given the leftward tokens within target segment and source tokens. Besides, the input of decoder is blocks of target tokens without the masked tokens introduced during the pre-training.

To bridge the MLM task and AR generation, we design a simple yet effective strategy for the decoder input, named as masking scheduler. We deploy the linear masking decay for the scheduler to train the model from easier data to harder one.

We summarize the training process with masking scheduler in Algorithm 1. At the beginning of fine-tuning, we randomly mask a portion of input tokens for the decoder. Since the decoder has already learned to bidirectionally reconstruct the

Models	Arch.	Obj.	SQuAD2	MNLI-(m/mm)	QNLI	QQP	SST	Avg.
T5	Enc-dec	AR	- / -	85.56/85.20	89.30	84.76	93.54	87.67
T5 Encoder			77.87/80.83	85.54/85.40	92.32	87.99	93.27	88.90
RoBERTa	Enc-only	NAR	79.14/81.86	85.88/**86.10**	92.39	87.60	93.16	89.03
MLAE	**Enc-dec**	**NAR**	**79.93/82.84**	**86.15**/86.05	**93.05**	**88.08**	**93.62**	**89.39**

Table 1: Results for T5, T5 encoder, RoBERTa and MLAE on the dev set of SQuAD2 and GLUE benchmark. We report EM/F1 scores for SQuAD2.

masked tokens in the pre-training, masking scheduler creates easier samples for the decoder. As the training progresses, the mask ratio is linearly decayed to zero, which makes the decoder gradually adapt from NAR reconstruction to AR generation.

Similarly in the fine-tuning for NAR generation, the decoder non-autoregressively generates target tokens from a full set of the unknown tokens. We replace the unknown tokens with the masked tokens used in the pre-training. Then we adopt Masked-and-predict (Ghazvininejad et al., 2019) strategy to narrow the gap between NAR generation and the MLM pre-training. The number of the masked tokens is sampled from a uniform distribution between one and the maximum sequences' length.

4 Experiments

In this section, we first introduce the setup of pre-training, then conduct experiments on both NLU tasks (i.e., the GLUE benchmark and extractive question answering), and NLG tasks (i.e., abstractive summarization), including AR and NAR paradigms.

4.1 Setup

Models We pre-train RoBERTa, T5 and MLAE with the same corpus. RoBERTa has a 12-layer encoder. T5 is based on the vanilla encoder-decoder, which has a 12-layer encoder and 4-layer decoder. For a fair comparison, MLAE has the same depth for the encoder and the decoder, respectively. We adopt the BERT-base setting: the hidden dimension, intermediate dimension of feed-forward layers and attention heads for all models are 768, 3072 and 12 respectively.

Data Following Devlin et al. (2019), we use the BookCorpus (Zhu et al., 2015) and English-Wikipedia as the pre-training corpus. The BookCorpus is a large collection of free novel books written by unpublished authors, which contains 800M words. We remove non-text parts for English-Wikipedia, which leads to 2.5B words.

For all models, we set the mask ratio as 15%. The maximum length is 512 tokens. We randomly mask consecutive spans rather than tokens. The average length of span is 3 tokens. We adopt the masked language modeling as the pre-training task for RoBERTa and MLAE, span corruption for T5. The vocabulary is built from a SentencePiece (Kudo and Richardson, 2018) tokenizer with 64K tokens.

Training We train our model and the baselines with Adam (Kingma and Ba, 2015) optimizer for 125K steps. The batch size is set as 2,048. The whole training procedure takes about 2 days on 64 NVIDIA Tesla V100 GPUs. The other hyper-parameters used in pre-training are detailed in Table 7 of Appendix A.

4.2 Results of Understanding Tasks

We evaluate our model and the baselines on the large tasks of GLUE benchmark and SQuAD2 dataset. For T5 encoder and MLAE, we only use their encoder as feature extractor, then add a task layer for them. Besides, we reformat the text classification to text-to-text generation, and directly fine-tune T5 without any modifications. More details are in Table 8 and Table 9 of Appendix A.

GLUE benchmark (Wang et al., 2019) is a collection of nine language understanding tasks, including linguistic acceptability, question answering, sentiment analysis and textual entailment. We choose the large tasks of GLUE benchmark, namely MNLI, QNLI, QQP and SST.

SQuAD2 (Rajpurkar et al., 2018) is one of the most popular benchmarks for extractive question answering, which combines SQuAD (Rajpurkar et al., 2016) with unanswerable questions.

We report the results of our model and the baselines in Table 1. T5 encoder outperforms T5 by a gain of 1.23 average score on the large tasks of GLUE benchmark. It shows that reformatting text classification to generation leads to the degradation of performance. Besides, RoBERTa outperforms

	Models	Arch.	Obj.	RG-1	RG-2	RG-L
	RoBERTa (Liu et al., 2019)	Enc-only	NAR	40.19	17.50	38.83
	T5 (Raffel et al., 2020)	Enc-dec	AR	41.29	18.37	39.55
[1]	**MLAE (Ours)**	**Enc-dec**	**NAR**	**42.58**	**19.30**	**40.73**
[2]	[1] - scheduled masking			41.50	18.62	39.94
[3]	[2] - asymmetric architecture			40.50	17.58	38.83
[4]	[2] - pretrain w/o cross-attn			41.29	18.40	39.75

Table 2: Results of RoBERTa, T5 and MLAE for AR generation on the test set of XSum dataset.

Models	RG-1	RG-2	RG-L
RoBERTa	37.03	15.03	31.41
T5	40.43	17.50	34.21
MLAE	**40.74**	**17.78**	**34.61**

Table 3: Results of RoBERTa, T5 and MLAE for AR generation on the test set of CNN/DM dataset.

T5 encoder on both two dataset, which verifies our analysis that the vanilla encoder-decoder pre-training undermines the ability of its encoder.

Furthermore, our model achieves gains of 0.36 average score on the large tasks of GLUE benchmark, 1.28 EM and 1.27 F1 on SQuAD2 dataset compared with RoBERTa. It demonstrates that MLAE creates a more powerful encoder than RoBERTa and the encoder part of T5.

4.3 Results of AR Generation

For AR generation, we conduct experiments on two popular benchmarks, Extreme summarization (XSum) and CNN/Daily Mail (CNN/DM) dataset.

XSum (Narayan et al., 2018) is a collection of 227K online articles and single sentence summaries harvested from the British Broadcasting Corporation(BBC). The average input and output lengths are 359 and 21 respectively.

CNN/DM (Hermann et al., 2015; Nallapati et al., 2016) contains online news articles accompanying with multi-sentence summaries. The average tokens of input and output are 781 and 56 respectively.

For RoBERTa, due to lack of pre-trained decoder, we add a randomly initialized 4-layer decoder for it, and fine-tune the whole model as the vanilla encoder-decoder. For MLAE, we adopt Seq2seq-style fine-tuning and initialize the cross-attention layers by the weight of self-attention layers. The masking scheduler is also deployed for the decoder: we randomly mask 60% tokens of decoder input at beginning and linearly decay the mask ratio to 0 as the training progresses.

We fine-tune our model and the baselines for 30K updates on CNN/DM dataset, 50K updates on XSum dataset; and select the best checkpoint based on their validation loss. For a fair comparison, we use the same hyper-parameters for all models. More details can be found in Table 10 and Table 11 of Appendix A. For inference, we truncate the inputs to be 512 tokens and use beam search strategy to generate target sentences. We set beam size as 6, length penalty as 1.0. We use ROUGE (Lin, 2004) as the evaluation metric for all experiments.

Table 2 and Table 3 summarize the results of our model and the baselines on the test set of XSum and CNN/DM dataset, respectively. T5 and MLAE outperform RoBERTa by a large gain on both two datasets. It verifies that the pre-trained decoder can significantly improve the quality of generation. Furthermore, compared with T5, MLAE achieves improvements of 0.93 ROUGE-2 on XSum dataset, and has comparable performance on CNN/DM dataset. It shows the effectiveness of our model on AR generation.

4.4 Results of NAR Generation

For NAR generation, we choose iNAT (Lee et al., 2018), InsT (Stern et al., 2019), LevT (Gu et al., 2019) and CMLM (Ghazvininejad et al., 2019) as the baselines trained from the scratch.

To explore the impact of pre-training strategy, we pre-train the T5-NAR model on the 16G corpus. The only difference between T5-NAR and T5 lies on the design of the decoder. The input of T5-NAR decoder is a full set of the masked tokens. In the self-attention layer of T5-NAR's decoder, we remove the masking for the rightward tokens to allow bidirectional information. Above all, the decoder of T5-NAR is designed to non-autoregressively reconstruct the target tokens from the masked tokens.

We pre-train MLAE with cross-attention layers on the same 16G corpus, then directly load our

Models	RG-1	RG-2	RG-L
iNAT (Lee et al., 2018)	20.71	4.39	22.94
InsT (Stern et al., 2019)	21.44	6.77	24.66
LevT (Gu et al., 2019)	25.02	7.41	27.15
CMLM (Ghazvininejad et al., 2019)	29.24	7.70	28.93
T5-NAR	31.48	9.02	30.80
MLAE (Ours)	**39.08**	**14.81**	**37.25**

Table 4: Results of MLAE and the baselines for NAR generation on the test set of XSum dataset. All models are trained with a 12-layer encoder and a 4-layer decoder for a fair comparison.

Models	Layers	# Params	AR			NAR		
			RG-1	RG-2	RG-L	RG-1	RG-2	RG-L
BANG (Qi et al., 2021)	6L-6L	100M	41.09	18.37	33.22	34.71	11.71	29.16
MLAE (Ours)	6L-6L		41.69	18.63	39.93	38.56	14.41	36.76
	9L-4L		**42.10**	**18.76**	**40.22**	**39.25**	**14.74**	**37.21**

Table 5: Results of BANG (Qi et al., 2021) and MLAE on the test set of XSum dataset, including AR and NAR generation paradigm. AL-BL refers to A-layer encoder and B-layer decoder.

model into the CMLM. For masking scheduler, we adopt Masked-and-predict (Ghazvininejad et al., 2019) strategy.

We evaluate our model and the baselines on XSum dataset. Despite knowledge distillation can improve the quality of NAR generation, the results highly depend on the distilled dataset. To enable other researchers to reproduce our results more easily, we do not perform knowledge distillation for our model and the baselines. Since NAR models need more updates to converge, we set the maximum updates as 300K. The NAR baselines trained from the scratch are integrated into Fairseq library (Ott et al., 2019). We implement these baselines with default settings[2]. For a fair comparsion, all models have a 12-layer encoder and a 4-layer decoder.

For inference, we truncate the inputs to be 512 tokens and use iterative decoding strategy. The tokens with low confidence will be masked and re-generated in the next cycle until the iterations reaches a manually set number. Following Qi et al. (2021), the maximum iteration is set as 10. We merge consecutive repeated tokens to ease the problem of repeated tokens.

Table 4 presents the results of MLAE and the baselines. T5-NAR and MLAE achieve a gain of over 1 ROUGE-2 compared with the other baselines trained from the scratch. It shows that via pre-training the performances for NAR generation are significantly improved. Further, with a more powerful encoder, MLAE outperforms T5-NAR by improvements of 5.79 ROUGE-2 on XSum dataset. It shows the effectiveness of MLAE pre-training on the NAR generation paradigm.

[2]Fairseq NAR baselines

4.5 Comparison with BANG

We compare our model with BANG (Qi et al., 2021) on XSum dataset. BANG is based on vanilla encoder-decoder, which fuses AR and NAR objectives through different attention mechanisms. It has a 6-layer encoder, 6-layer decoder and 768 hidden dimension.

MLAE is trained with a 9-layer encoder, a 4-layer decoder and the same hidden dimension, resulting in up to 100M backbone parameters for a fair comparison. Besides, we train MLAE with symmetric architecture. The pre-training corpus is the same as BANG. The fine-tuning hyper-parameters and methods of MLAE on AR and NAR generation are consistent with the experiments presented in Section 4.3 and Section 4.4 respectively. We adopt the same evaluation scripts following Qi et al. (2021).

We report the results of our models and BANG on Table 5. It demonstrates that 6L-6L and 9L-4L MLAE both has consistently better performance than BANG on AR and NAR generation paradigm. Especially, 9L-4L MLAE outperforms BANG by an improvement of 3.03 ROUGE-2 on the NAR generation.

	MLAE fine-tuning	SM	RG-1	RG-2	RG-L
	UniLM-style	✗	40.46	17.75	39.18
[1]	Seq2seq-style	✗	40.22	17.64	38.94
[2]	[1] + MLAE decoder	✗	40.92	18.19	39.52
[3]	[2] + shared self & cross attn	✗	41.50	18.62	39.94
[4]	[3] + Const. scheduler	✓	41.46	18.50	39.99
[5]	[3] + Linear decay scheduler	✓	**42.58**	**19.30**	**40.73**

Table 6: Comparisons between different fine-tuning strategies and implementations of the masking scheduler for MLAE. SM indicates whether to use scheduled masking for fine-tuning. For [4] and [5], the initial mask ratio is set as 60%.

5 Ablation Study

In this section, we conduct the ablation studies on the fine-tuning strategies for NLG tasks in Section 5.1, model architecture and scheduled masking in Section 5.2 and Section 5.3, respectively.

5.1 Fine-tuning Strategies

We present comparisons between UniLM-style and Seq2seq-style fine-tuning for AR generation on XSum dataset. For a fair comparison, we do not apply masking scheduler for all the experiments.

For UniLM-style fine-tuning, we use the bidirectional mask for the encoder and Seq2seq mask for the decoder. With the Seq2seq mask, MLAE decoder can auto-regressively generate the target tokens conditioned on the representations of source tokens and leftward tokens within the target segment. For Seq2seq-style fine-tuning, we insert cross-attention layers into each layer of MLAE decoder, then fine-tune the model as a vanilla encoder-decoder.

To explore the impact of pre-trained decoder, we conduct experiments for Seq2seq-style fine-tuning with a randomly initialized decoder. As shown in Table 6, the model with MLAE decoder outperforms it with random decoder by a gain of 0.55 ROUGE-2. However, the newly inserted cross-attention modules are still randomly initialized, which provides the room to use more pre-trained modules to improve the performance. Therefore, we further use each self-attention layer's weights of MLAE decoder to initialize correspond cross-attention layer's weights. The results show that with more pre-trained modules, we achieve greater gain for the quality of generation.

Besides, Table 6 shows that Seq2seq-style fine-tuning with a "fully" pre-trained decoder is a better way for MLAE on AR generation task: it outperforms UniLM-style fine-tuning by an improvement of 0.87 ROUGE-2 on XSum dataset.

5.2 Ablation on the Architecture

We compare MLAE based on symmetric and asymmetric architecture, with and without cross-attention layers for the decoder during the pre-training. The masking scheduler is not applied for all the experiments for a fair comparison.

We first compare MLAE with symmetric and asymmetric architecture for AR generation on XSum dataset. For symmetric architecture, we train MLAE with an 8-layer encoder, an 8-layer decoder and 768 hidden dimension, which leads to the same amount of parameters. We report the results of 8L-8L, 12L-4L MLAE in Table 2. It shows that asymmetric architecture is preferred for AR generation. This results from that the ability of the encoder is more crucial for the quality of AR generation.

Further we present comparisons for MLAE with and without cross-attention layers for the decoder during the pre-training. For a fair comparison, we pre-train a Base-size, 12L-4L MLAE with cross-attention layers on the same corpus. As shown in Table 2, MLAE pre-training without cross-attention layers slightly outperforms it with cross-attention layers by an improvement of 0.22 ROUGE-2 for AR generation.

5.3 Effect of Scheduled Masking

We first compare different implementations of the masking scheduler: the mask ratio is set as a constant, and linearly decayed to 0 as the training progresses. As shown in Table 6, linear decay is a better scheduler function for AR generation.

Further, we explore the impact of different initial mask ratio for the masking scheduler. We vary the initial mask ratio from 0% to 90% with an interval of 15%. Figure 3 shows the ROUGE-2 scores for AR generation on XSum dataset. It demonstrates

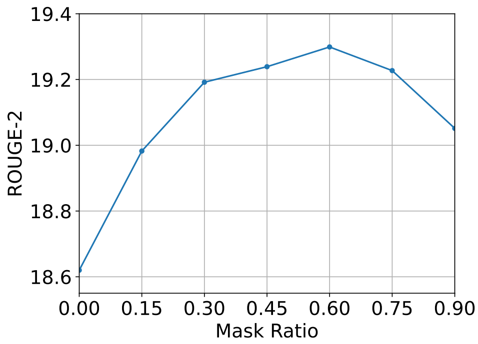

Figure 3: ROUGE-2 scores of MLAE for AR generation on the test set of XSum dataset varying the initial mask ratio of masking scheduler from 0% to 90%.

that replacing a small portion (15%) of decoder input with the masked token can significantly improve the performance of generation. As shown in Figure 3, a relatively high initial mask ratio, approximately 45% to 75%, is preferred for AR generation.

6 Related Work

Language model pre-training. The performance of downstream tasks benefits from the large-scale pre-trained models. BERT (Devlin et al., 2019) introduces MLM to pre-train the encoder-only Transformer, which allows the model to use bidirectional information to generate latent representations. A greater gain can be achieved by pre-training longer with more training data (Liu et al., 2019) and masking consecutive spans rather than tokens (Joshi et al., 2020).

However, although the encoder-only models achieve great success on NLU tasks, due to lack of pre-trained decoder, they are not effectively fine-tuned for NLG tasks. Besides, BERT reconstructs the masked tokens bidirectionally rather than auto-regressively, which broadens the gap between pre-training and fine-tuning for AR generation. To address these issues, UniLM (Dong et al., 2019) pre-trains BERT with different mask mechanisms for attention layers. With their proposed Seq2seq mask, UniLM can generate target tokens autoregressively with the encoder-only architecture.

Another line of research is to adopt vanilla encoder-decoder framework for pre-training. MASS (Song et al., 2019) randomly masks consecutive tokens for the input sentences. The encoder takes the corrupted sentences as input, including the masked and unmasked tokens; its decoder reconstructs the masked tokens. Different from MASS, BART (Lewis et al., 2020) feeds the corrupted sentences into the encoder, the uncorrupted sentences into the decoder, which reduces the mismatch between pre-training and fine-tuning. T5 (Raffel et al., 2020) aims to unify all text-based language problems into text-to-text format, which adopts vanilla encoder-decoder framework with span corruption.

NAR generation. Gu et al. (2017) first introduce vanilla Transformer encoder-decoder for NAR machine translation. NAR generation removes the assumption that each output word is conditioned on previously generated outputs. Although this parallel generation largely speeds up the inference, it is troubled by the repeated tokens problem. A lot of efforts are proposed to ease this issue (Lee et al., 2018; Gu et al., 2019; Stern et al., 2019; Ghazvininejad et al., 2019). Qi et al. (2021) introduces BANG to bridge AR and NAR generation with large-scale pre-training, which fuses AR and NAR objectives by different attention mechanisms.

Masked autoencoders. He et al. (2021) first introduces masked autoencoder for self-supervised vision pre-training. With a light-weight decoder and high masking ratio, MAE avoids wasting the model capacity on short-range dependencies, creates a more powerful encoder from reconstructing unsemantic pixels of the masked patches. After that, masked autoencoders are adopted for video pre-training (Tong et al., 2022; Feichtenhofer et al., 2022) and vision-language pre-training (He et al., 2022; Geng et al., 2022).

7 Conclusion

We propose MLAE, a new pre-training paradigm based on masked autoencoders. With MLAE, we not only have a pre-trained decoder for NLG tasks, but also a more powerful encoder compared with the encoder part of vanilla encoder-decoder, even the encoder-only model. Besides, we design a simple yet effective method, named as masking scheduler, to bridge MLM pre-training and generation. The proposed MLAE combines the best of two worlds: the encoder-only models' good performance on NLU tasks and the vanilla encoder-decoders' good performance on NLG tasks, including AR and NAR paradigms. It shows that MLAE is a preferred alternative compared with vanilla encoder-decoder.

8 Limitations

While this work empirically finds that non-autoregressive modeling improves language model pre-training, the mechanism behind this inductive bias needs more in-depth analysis. In addition, we do not explore the multilingual pre-training of MLAE in the paper, which will be left as future work. Like most of the existing pre-trained models, our method may have some potential bias originating from the pre-training data.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

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. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13042–13054.

Christoph Feichtenhofer, Haoqi Fan, Yanghao Li, and Kaiming He. 2022. Masked autoencoders as spatiotemporal learners. *CoRR*, abs/2205.09113.

Xinyang Geng, Hao Liu, Lisa Lee, Dale Schuurams, Sergey Levine, and Pieter Abbeel. 2022. Multimodal masked autoencoders learn transferable representations. *CoRR*, abs/2205.14204.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6111–6120. Association for Computational Linguistics.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2017. Non-autoregressive neural machine translation. *CoRR*, abs/1711.02281.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11179–11189.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. 2021. Masked autoencoders are scalable vision learners. *CoRR*, abs/2111.06377.

Sunan He, Taian Guo, Tao Dai, Ruizhi Qiao, Chen Wu, Xiujun Shu, and Bo Ren. 2022. VL-MAE: vision-language masked autoencoder. *CoRR*, abs/2208.09374.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Trans. Assoc. Comput. Linguistics*, 8:64–77.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *CoRR*, abs/1808.06226.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1173–1182. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 280–290. ACL.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1797–1807. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *CoRR*, abs/1904.01038.

Weizhen Qi, Yeyun Gong, Jian Jiao, Yu Yan, Weizhu Chen, Dayiheng Liu, Kewen Tang, Houqiang Li, Jiusheng Chen, Ruofei Zhang, Ming Zhou, and Nan Duan. 2021. BANG: bridging autoregressive and non-autoregressive generation with large scale pre-training. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8630–8639. PMLR.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *CoRR*, abs/1806.03822.

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, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: masked sequence to sequence pre-training for language generation. *CoRR*, abs/1905.02450.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5976–5985. PMLR.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. 2022. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *CoRR*, abs/2203.12602.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yukun Zhu, Ryan Kiros, Richard S. 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. *CoRR*, abs/1506.06724.

A Hyper-parameters

Hyperparameters	Value
Hidden size	768
FFN inner hidden size	3,072
Attention heads	12
Peak Learning rate	5e-4
Learning rate schedule	Polynomial decay
Warm-up updates	10,000
Warm-up init learning rate	1e-7
Sequence length	512
Batch size	2,048
Mask ratio	15%
Adam β	(0.9, 0.98)
Training updates	125K
Gradient clipping	2.0
Dropout	0.1
Weight decay	✗

Table 7: Hyperparameters for MLAE and the baselines pre-training.

Hyperparameters	Value
Peak Learning rate	{1e-5, 2e-5, 3e-5, 4e-5}
Learning rate schedule	polynomial decay
Adam β	(0.9, 0.98)
Warm-up	{10%, 20%}
Batch size	32
Training epochs	3
Seed	{1, 2, 3}
Gradient clipping	✗
Dropout	0.1
Weight decay	0.01

Table 8: Hyperparameters for MLAE and the baselines fine-tuning on the large tasks of GLUE benchmark.

Hyperparameters	Value
Peak Learning rate	{2e-5, 3e-5, 4e-5}
Learning rate schedule	polynomial decay
Adam β	(0.9, 0.999)
Warm-up	10%
Batch size	32
Training epochs	3
Seed	{1, 2, 3}
Gradient clipping	✗
Dropout	0.1
Weight decay	0.01

Table 9: Hyperparameters for MLAE fine-tuning on the SQuAD2 dataset.

Hyperparameters	AR	NAR
Peak Learning rate	{7e-5, 1e-4}	
Learning rate schedule	inverse sqrt	
Warm-up	500	10,000
Maximum tokens	8×4096	
Training updates	30K	300K
Adam β	(0.9, 0.999)	
Gradient clipping	1.0	
Dropout	0.1	
Weight decay	0.01	

Table 10: Hyperparameters for MLAE fine-tuning for AR and NAR generation on the XSum dataset.

Hyperparameters	Value
Peak Learning rate	{7e-5, 1e-4}
Learning rate schedule	inverse sqrt
Warm-up	500
Maximum tokens	16×4096
Training updates	30K
Adam β	(0.9, 0.999)
Gradient clipping	1.0
Dropout	0.1
Weight decay	0.01

Table 11: Hyperparameters for MLAE fine-tuning on the CNN/DM dataset.