ATTENTIVE MLP FOR NON-AUTOREGRESSIVE GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive (AR) generation almost dominates sequence generation for its efficacy. Recently, non-autoregressive (NAR) generation gains increasing popularity for its efficiency and growing efficacy. However, its efficiency is still bottlenecked by softmax attention of quadratic complexity on computational time and memory cost. Such bottleneck prevents non-autoregressive models from scaling to long sequence generation and few works have been done to mitigate this problem. In this paper, we propose a novel MLP variant, **A**ttentive **M**ulti-**L**ayer **P**erceptron (AMLP), to produce a generation model with linear time and space complexity. Different from classic MLP with static and learnable projection matrices, AMLP leverages adaptive projections computed from inputs in an attentive mode. And different from softmax attention, AMLP uses sample-aware adaptive projections to enable communications among tokens in a sequence, and models the measurement between the query and key space. Furthermore, we marry AMLP with popular NAR models, deriving a highly efficient NAR-AMLP architecture with linear time and space complexity. The empirical results show that such marriage architecture NAR-AMLP surpasses competitive efficient NAR models, by a significant margin on text-to-speech synthesis and machine translation. We also test AMLP's self- and cross-attention ability separately with extensive ablation experiments, and find them comparable or even superior to the other efficient models. The efficiency analysis further shows that AMLP speeds up the inference and extremely reduces the memory cost against vanilla non-autoregressive models. All the experiments reveal that NAR-AMLP is a promising architecture in both of efficiency and efficacy.

1 INTRODUCTION

Attention-based sequence generation methods have achieved great success and gained increasing popularity in machine learning (Vaswani et al., 2017; Li et al., 2019a; Liu et al., 2021b; Dosovitskiy et al., 2021). A large body of research in neural architectures has been devoted to the autoregressive (AR) method (Peng et al., 2022; 2021), where tokens are generated one after another in an iterative manner. The computational overhead in decoding can thus be prohibitive, especially for long sequences. Recently, non-autoregressive (NAR) generation attracts more attention for its efficiency and growing efficacy (Gu et al., 2018; 2019; Qian et al., 2021a;b; Ren et al., 2021; Chang et al., 2022). In a non-autoregressive model, the decoder generates the target sequence all at once, significantly reducing its computational overhead at the inference stage.

Nevertheless, relatively little research has been done on the attention architecture in non-autoregressive models. In particular, the conventionally adopted softmax attention comes with a quadratic time and memory cost. It is therefore still difficult to scale up non-autoregressive models to long sequence generation tasks.

In this paper, we propose Attentive Multi-Layer Perceptron (§2.2; AMLP) to integrate the attention mechanism with the multi-layer perceptron (MLP) in non-autoregressive architecture, resulting in a fully parallelizable sequence generation model with linear complexity. Unlike the widely-used MLP whose weights are invariant across different sequences, we compute the weights in AMLP through adaptive projections from (multiple) input tokens and model their interactions in an attentive manner. Specifically, we put forward two methods (§2.3) to compute the adaptive projections in

AMLP, which implicitly model the association between the query and key space. We utilize the simplicity and efficiency of MLP while obtaining the strong modeling capability of AMLP for input tokens' communication. Finally, we present a hybrid NAR-AMLP model (§2.4) to achieve both linear complexity and high parallelism.

We evaluate the AMLP architecture on text-to-speech synthesis for a relatively long sequence scenario and machine translation for a relatively short sequence scenario. Experiments show that AMLP achieves more superior scores with objective measurements compared with the strong softmax attention counterpart (§3.2) on text-to-speech synthesis, with less computational cost (§3.4). On machine translation, AMLP performs slightly behind vanilla attention (by ˜1 BLEU) but achieves the best result among efficient NAR models with linear complexity (§3.2). Further, we test the self- and cross-attention ability of AMLP on super resolution and long sequence time-series forecasting tasks, respectively. Empirical results show that AMLP is on par with other efficient attention in self-attention and achieves the best performance in cross-attention scenarios (§3.3). Additionally, when scaling to long sequence, AMLP reduces the memory footprint substantially and further improves the inference speed in NAR models (§3.4).

2 Non-Autoregressive Generation with Attentive MLP

In this section, we first give a brief introduction to autoregressive (AR) and non-autoregressive (NAR) generation, and then present the AMLP architecture to model the communication among sequence tokens. Finally, we build up an NAR-AMLP architecture with linear time and space complexity.

2.1 Background: Autoregressive and Non-Autoregressive Generation

Given a source sequence $X_{1:m}$, conditional sequence generation targets to predict a target sequence $Y_{1:n}$ by modeling the conditional probability $p(Y|X)$. Autoregressive generation decomposes the probability $p(Y|X)$ as:

$$p(Y|X) = \prod_{i=1..n} p(Y_i|Y_{<i}, X), Y_{<1} = \emptyset. \tag{1}$$

Although such decomposition is proved effective, it suffers from two main drawbacks: efficiency and exposure bias. On the one hand, the autoregressive decoding process, where each token depends on the previous predicted ones, prevents the model from fast inference in usage. On the other hand, teacher-forcing exposes ground truth tokens in network inputs during the training process, where the exposed tokens are unable to observe in inference. Such exposure creates an inconsistency between the training and inference, and harms the prediction quality.

Recently, non-autoregressive generation shows its capability of sequence modeling in terms of both efficiency and efficacy, which decomposes the conditional probability $p(Y|X)$ via a Naïve Bayes assumption:

$$p(Y|X) = \prod_{i=1..n} p(Y_i|X) \tag{2}$$

The NAR decomposition enables parallel decoding for each token, and speeds up the inference process substantially. Although NAR generation is much faster than AR generation, its speed is still limited by the $O\left((n+m)^2\right)$ time complexity of the softmax attention module. This is especially problematic in modeling long sequences.

2.2 Attentive Multi-Layer Perceptron

Modeling interactions between tokens is crucial and challenging in sequence generation. Transformer (Vaswani et al., 2017) stacks the MLP, which aims to learn features of individual tokens, on top of the attention block, which is responsible for modeling the communication within the sequence. In AR generation, the attention needs to be recomputed for each time step through the recurrent process, as the key and value set is changing. However, this procedure is non-causal in NAR generation. We therefore are able to integrate the modeling of token interactions into the MLP architecture and make the whole architecture fully parallelizable and more efficient.

Given a sequence representation $\mathbf{X} \in \mathbb{R}^{n \times d}$, where n is the sequence length and d is dimensionality of the feature space, the conventional MLP models the feature of individual token $\mathbf{X}_i \in \mathbb{R}^d$ as:

$$\text{MLP}(\mathbf{X}_i) = \sigma(\mathbf{X}_i W_1) W_2 \tag{3}$$

where $W_1 \in \mathbb{R}^{d \times d_h}$, $W_2 \in \mathbb{R}^{d_h \times d}$ are learnable parameters d_h is the dimensionality of hidden space. $\sigma(\cdot)$ is a non-linear activation function such as $\text{ReLU}(\cdot)$. However, it disables the communication between tokens in the sequence, and prevents the model from learning contextualized token representations.

A widely-used approach to enable communication between each token in a sequence is the attention mechanism (Vaswani et al., 2017). Vanilla attention learns to incorporate source sequence features $\mathbf{K}, \mathbf{V} \in \mathbb{R}^{m \times d}$ into target $\mathbf{Q} \in \mathbb{R}^{n \times d}$ with an attention matrix

$$\text{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\mathbf{Q}\mathbf{K}^\top)\mathbf{V} \tag{4}$$

where m, n are the source and target length respectively. Here we omit the input projections for $\mathbf{Q}, \mathbf{K}, \mathbf{V}$, the output projection, and the scaling factor $1/\sqrt{d}$ for simplicity.

The motivation of Attentive Multi-Layer Perceptron (AMLP) starts from the fact that the vanilla softmax attention can be viewed as a projection function as $\text{SA}(\cdot|\mathbf{K}, \mathbf{V}) : \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times d}$ which projects the original $\mathbf{Q} \in \mathbb{R}^{n \times d}$ with $\mathbf{K}$ and $\mathbf{V}$ features as its context while preserving $\mathbf{Q}$'s shape. Thus we propose an alternative solution by fusing key $\mathbf{K} \in \mathbb{R}^{m \times d}$ and value $\mathbf{V} \in \mathbb{R}^{m \times d}$ information into query $\mathbf{Q} \in \mathbb{R}^{n \times d}$, via a symmetric and positive semi-definite distance matrix $\mathbf{\Sigma} \in \mathbb{R}^{d \times d}$ on $\mathbf{Q}$ and $\mathbf{K}$ space. The contextualizing process on $\mathbf{Q}$ can be formulated as:

$$f(\mathbf{Q}; \mathbf{K}, \mathbf{V}) = \mathbf{Q}\mathbf{\Sigma}\mathbf{K}^\top\mathbf{V} \tag{5}$$

where $\mathbf{\Sigma}$ is computed from $\mathbf{Q}$ and $\mathbf{K}$, representing the distance between target sequences and source sequences in each embedding component. We add a more detailed explanation for the transition from attention to AMLP in Appendix B.

Previous work on efficient attention (Xiong et al., 2021) has shown that the softmax attention matrix $\text{softmax}(\mathbf{Q}\mathbf{K}^\top)$ could be decomposed into two low-rank matrices. With similar functionality, the matrix $\mathbf{Q}\mathbf{\Sigma}\mathbf{K}^\top$ can also be treated as a low-rank matrix. Without taking any low-rank assumptions on input $\mathbf{Q}, \mathbf{K}$, we decompose the distance matrix as:

$$\mathbf{\Sigma} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top = \mathbf{U}\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{U}^\top \approx \mathbf{U}\hat{\mathbf{\Lambda}}^{\frac{1}{2}}\hat{\mathbf{\Lambda}}^{\frac{1}{2}}\mathbf{U}^\top = (\mathbf{U}\hat{\mathbf{\Lambda}}^{\frac{1}{2}})(\mathbf{U}\hat{\mathbf{\Lambda}}^{\frac{1}{2}})^\top = \mathbf{L}\mathbf{L}^\top \tag{6}$$

where $\mathbf{U}$ is the orthogonal eigenvector of matrix and $\mathbf{\Lambda}$ is the diagonal eigenvalues matrix. $\hat{\mathbf{\Lambda}}$ here is an approximation to $\mathbf{\Lambda}$ by keeping largest-c eigen-values and masking the others with 0, where c is a hyper-parameter in AMLP. Thus we derive a decomposition equation $\mathbf{\Sigma} \approx \mathbf{L}\mathbf{L}^\top$ where $\mathbf{L} = \kappa(\mathbf{Q}, \mathbf{K})^\top \in \mathbb{R}^{d \times c}$ indicates a low-rank matrix. We will show two different methods for parameterization of $\mathbf{L}$, resulting in two different AMLP variants. We rewrite Eq. 5 by decomposing the distance matrix $\mathbf{\Sigma}$ as:

$$f(\mathbf{Q}, \mathbf{K}, \mathbf{V}) \approx \mathbf{Q}\mathbf{L}\mathbf{L}^\top\mathbf{K}^\top\mathbf{V} \tag{7}$$

Now Eq. 5 could be approximated with Eq. 7 by linearly projecting the original $\mathbf{Q}$ with adaptive weights twice. By reordering the computation and adding nonlinearity into Eq. 5, we derive a general form of AMLP model as:

$$\text{AMLP}(\mathbf{Q}; \mathbf{K}, \mathbf{V}) = \sigma_1(\mathbf{Q}W_{\mathbf{Q},\mathbf{K}})W_{\mathbf{Q},\mathbf{K},\mathbf{V}} \tag{8}$$

where the nonlinear function $\sigma_1(\cdot)$ can be adjusted arbitrarily. Following the form of Eq. 8, we will further introduce two AMLP variants in the rest of this section, by specifying $\mathbf{L} = W_{\mathbf{Q},\mathbf{K}} = \kappa(\mathbf{Q}, \mathbf{K})$, computational order and nonlinear function. The computation of weights in AMLP fuses token-level communication, while MLP models tokens in a sequence independently. Therefore, AMLP enables the communication between tokens in a sequence. And different from vanilla softmax attention, AMLP utilizes a distance matrix $\mathbf{\Sigma}$ between $\mathbf{Q}$ and $\mathbf{K}$ spaces to fuse information among their contexts and outputs a contextualized $\mathbf{Q}$. Through this distance matrix, AMLP computes the similarity between $\mathbf{Q}$ and $\mathbf{K}$ like softmax attention, and leverages it to aggregate $\mathbf{V}$.

2.3 Parameterization of Adaptive Weights

In this section, we describe two methods for the parameterization of two adaptive weight matrices $W_{\mathbf{Q},\mathbf{K}}$ and $W_{\mathbf{Q},\mathbf{K},\mathbf{V}}$. Fig. 1 illustrates the computation graph of these two methods. [1]

[1] AMLP is implemented with multiple heads (Vaswani et al., 2017), but for simplicity and without loss of generality, we will discuss our AMLP computation process in a single-head setting.

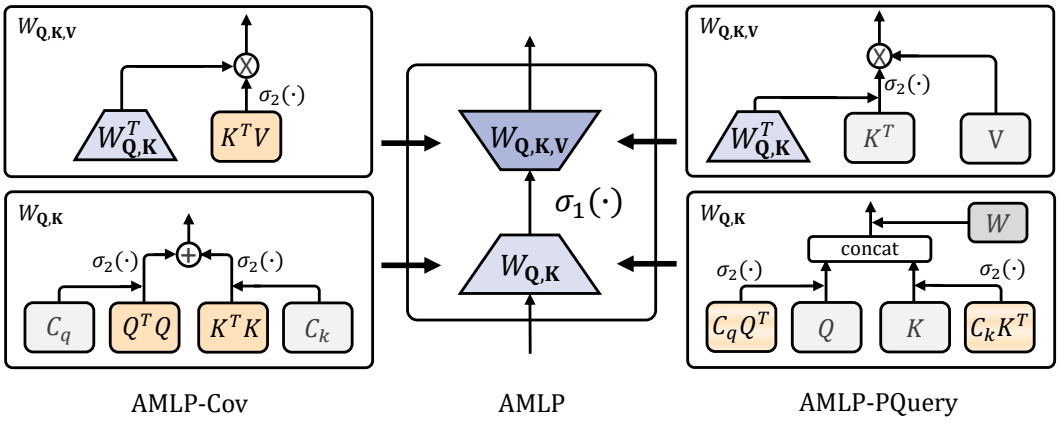


Figure 1: Computation diagram of two AMLP variants. The middle part shows the computation of basic AMLP. The Left and right figures show the detailed computation of two adaptive weight matrics in AMLP-Cov and AMLP-PQuery.

(Cross-)Covariance Here we present AMLP-Cov, a variant that adopts (cross-)covariance to parameterize $W_{\mathbf{Q},\mathbf{K}}$ and $W_{\mathbf{Q},\mathbf{K},\mathbf{V}}$. One challenge of AMLP is to fuse information of $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ of different shapes into static-shaped projection matrices $W_{\mathbf{Q},\mathbf{K}}$ and $W_{\mathbf{Q},\mathbf{K},\mathbf{V}}$. Inspired by (Ali et al., 2021), we propose to use $\mathbf{Q}, \mathbf{K}$'s covariance and the cross-covariance between $\mathbf{K}$ and $\mathbf{V}$ in AMLP. To obtain $\mathbf{L} = \kappa(\mathbf{Q}, \mathbf{K})^\top$, we separately compute $\mathbf{Q}$'s and $\mathbf{K}$'s covariance matrices and combines them with learned down-sampling projection matrices $C_q \in \mathbb{R}^{c\times d}$ and $C_k \in \mathbb{R}^{c\times d}$:

$$\kappa(\mathbf{Q}, \mathbf{K}) = C_q \left(\sigma_2(\mathbf{Q}^\top \mathbf{Q})\right) + C_k \left(\sigma_2(\mathbf{K}^\top \mathbf{K})\right) \tag{9}$$

where $\sigma_2(\cdot)$ is set to softmax function as Ali et al. (2021) suggest. The covariance matrices of $\mathbf{Q}, \mathbf{K}$ are of the same shape and can be directly fused. We add the softmax function as a non-linear activation to enhance the expressiveness. For $W_{\mathbf{Q},\mathbf{K},\mathbf{V}}$, we notice the shapes of $\mathbf{K}$ and $\mathbf{V}$ are usually identical, and we hence use their cross-covariance $\mathbf{K}^\top\mathbf{V}$ for computation in Eq. 8. $W_{\mathbf{Q},\mathbf{K},\mathbf{V}}$ is then formulated by transforming the cross-covariance $\mathbf{K}^\top\mathbf{V}$ to query space by $\mathbf{L}$ as:

$$W_{\mathbf{Q},\mathbf{K},\mathbf{V}} = \mathbf{L}^\top \sigma_2(\mathbf{K}^\top \mathbf{V}) \tag{10}$$

Pseudo-Queries To further improve the communication between target and source sequences, we propose AMLP-PQuery, which treats learnable C_q, C_k and $\mathbf{L}^\top$ as pseudo attention queries. Specifically, AMLP-PQuery estimates $W_{\mathbf{Q},\mathbf{K}}$ by fusing features from query and key to the hidden space with an extra learnable weight $W \in \mathbb{R}^{2d\times d}$:

$$W_{\mathbf{Q},\mathbf{K}} = \mathbf{L}^\top = \left[\sigma_2(C_q\mathbf{Q}^\top)\mathbf{Q}; \sigma_2(C_k\mathbf{K}^\top)\mathbf{K}\right] W \tag{11}$$

where $\sigma_2(\cdot)$ is set to softmax as AMLP-Cov. For $W_{\mathbf{Q},\mathbf{K},\mathbf{V}}$, we notice that $\mathbf{L}^\top$ has fused features from $\mathbf{Q}$. So we again treat $\mathbf{L}^\top$ as a pseudo query to fuse features from the source sequence:

$$W_{\mathbf{Q},\mathbf{K},\mathbf{V}} = \sigma_2(\mathbf{L}^\top\mathbf{K}^\top)\mathbf{V} \tag{12}$$

With explicit communication between $\mathbf{Q}$ and $\mathbf{K}$ in $W_{\mathbf{Q},\mathbf{K},\mathbf{V}}$, the alignment between different sequences is enhanced; therefore, AMLP-PQuery is more adaptive to cross-attention.

2.4 Hybrid Architecture: NAR-AMLP

We combine AMLP with NAR for lower memory costs, faster inference speed and higher parallelism because AMLP and NAR are mutually reinforcing. On one hand, NAR parallelizes the inference process, but its efficiency is still hindered by vanilla attention. AMLP, as a plug-in efficient attentive module, mitigates the inefficiency effortlessly. On the other hand, the non-autoregressive pipeline provides a non-causal encoding framework, with which the computation of AMLP avoids fine-grained operations. However, in autoregressive modeling, the dedicated operations similar to Peng

et al. (2021) are indispensable to maintain the causality. These computation steps increase heavy memory costs and large time consumption in the training phase, which is illustrated in Appendix A with more details. Therefore, we decide to incorporate AMLP into NAR to produce an efficient model in both training and inference stages.

2.5 Complexity Analysis

Without loss of generality, we focus on the complexity in the typical decoder architecture and omit the independent factor *w.r.t.* target length n and source length m for simplicity.

AMLP-Cov & AMLP-PQuery Note that the inner dimension c is a constant to both m and n. The sequential computation of two adaptive projection matrices and the overall MLP computation in Eq. 8 are all of $O(n+m)$. Therefore, the time and memory complexity of AMLP (both AMLP-Cov and AMLP-PQuery) is $O(n+m)$.

NAR-AMLP Due to the quadratic complexity of softmax attention, traditional non-autoregressive models still take up $O(n^2+nm+m^2)$ time and memory in the whole architecture. Therefore, we replace softmax modules in non-autoregressive models with AMLP, deriving an NAR-AMLP architecture with linear time and space complexity.

3 Experiments

We conduct extensive experiments, covering the fields of speech, natural language processing, time-series and computer vision.[2] Specifically, we first apply our hybrid architecture NAR-AMLP in two tasks: Text-to-Speech Synthesis and Machine Translation. Then we assess AMLP's self-attention and cross-attention abilities on super resolution and long sequence time-series forecasting tasks, respectively. Finally, we conduct ablation studies to show the hidden philosophy of AMLP and explore how efficient if AMLP scales to long-sequence modeling.

3.1 Baselines

We compare AMLP with the following efficient architectures.

gMLP (Liu et al., 2021a): gMLP remains W_1 the same as vanilla MLP, but uses a special gating unit to parameterize W_2 in Eq. 3. Similar to vanilla MLP, gMLP could not integrate features from other sequences to the sample-aware matrix. Therefore, it is incapable of cross-attention.

XCA (Ali et al., 2021): XCA approximates $\mathbf{QK}^\top$ by replacing it with the cross-covariance $\mathbf{K}^\top\mathbf{Q}$. Although this substitution achieves linear complexity, it also forbids XCA from generalizing to cross-attention due to different target and source sequence lengths.

ABC (Peng et al., 2022): ABC is an efficient attention that uses two learnable matrices to compress $\mathbf{K}$ and $\mathbf{V}$ to bounded memory respectively. It can replace both vanilla self and cross attention.

local attention (Luong et al., 2015b): Local attention intuitively forces each query token to focus on its neighborhood tokens within a fixed window size. It is applicable to self-attention. Given prior sequence alignment information, it can also work as cross-attention.

3.2 Main Results of NAR-AMLP

Text-to-Speech We select LJSpeech (Ito & Johnson, 2017) dataset for this task, and use FastSpeech 2 (FS2) (Ren et al., 2021) and Transformer-TTS (Tr-TTS) (Li et al., 2019a) as the backbone models for NAR and AR, respectively. For both backbones, we replace all softmax attention modules with efficient ones to achieve linear complexity. We use AMLP-Cov variant and $\mathrm{ReLU}(\cdot)$ as $\sigma_1(\cdot)$ in Eq. 8. The alignment tool "g2pE" (Wang et al., 2021) is applied to train FastSpeech 2. For reproducibility, we use two widely-used objective evaluation metrics, Mel Cepstral Distortion (MCD) and Mel Spectral Distortion (MSD), to assess the quality of synthesized audio clips.

[2] In experiments, we take $\mathrm{softmax}(\cdot)$ as the nonlinear function $\sigma_1(\cdot)$ unless otherwise specified.

Table 1: Automatic evaluation metric on LJSpeech dataset. All models are trained by ourselves. n, m are the target and source sequence lengths. Colored rows represent NAR models.

Arch	Model	LJSpeech	
		MCD↓	MSD↓
Complexity: $O(n^2)$ *or* $O\left((n+m)^2\right)$			
AR	Tr-TTS	4.0953	2.1985
NAR	FS2	3.4748	1.9735
Complexity: $O(n)$ *or* $O(n+m)$			
AR	Tr-TTS (ABC)	5.1302	2.5957
NAR	FS2 (local)	3.4189	1.9704
	FS2 (ABC)	3.3925	1.9658
	FS2 (XCA)	3.5003	2.0239
	FS2 (gMLP)	3.4025	1.9641
	FS2 (AMLP)	**3.3274**	**1.9396**

Table 2: BLEU4 scores on WMT14 EN-DE and WMT14 DE-EN dataset. All models for comparison are implemented by ourselves. n, m are the target and source sequence lengths. Colored rows represent NAR models. Subscript figures $\boldsymbol{\Delta}$ denote performance drop comparing with backbone models.

Arch	Model	WMT' 14	
		En-De$_{\Delta}$	De-En$_{\Delta}$
Complexity: $O\left((n+m)^2\right)$			
AR	Tr	**27.38**	**31.26**
NAR	GLAT	26.24	30.10
Complexity: $O(n+m)$			
AR	Tr (local)	$24.77_{2.61}$	$28.21_{3.05}$
	Tr (ABC)	$25.86_{1.52}$	$29.09_{2.17}$
NAR	GLAT (local)	$18.19_{8.05}$	$21.36_{8.74}$
	GLAT (ABC)	$21.98_{4.26}$	$24.78_{5.32}$
	GLAT (AMLP)	$25.00_{\mathbf{1.24}}$	$28.42_{\mathbf{1.68}}$

We demonstrate the results in Table 1. AMLP substantially lowers the MCD and MSD values by a great margin up to 0.15 MCD with even lower complexity compared to vanilla models. Additionally, AMLP also outperforms other efficient models. Notably, we have significantly lower MCD than XCA which also leverages (cross-)covariance matrices.

Machine Translation For Machine Translation (MT), we launch our experiments on WMT 2014 English-German (WMT'14 En-De) and German-English (WMT'14 De-En) datasets (Bojar et al., 2014). We adopt AMLP-PQuery variant to GLAT (Qian et al., 2021a), which is a powerful fully NAR architecture without extra decoding algorithms. We exclude the AR-reranking process to make a fully linear-complexity generation process. Similar to TTS, we replace self/cross-attention modules in Transformer and GLAT to obtain their efficient variants. For completeness, we include widely-used AR architecture Transformer (Tr) (Vaswani et al., 2017) with competitive linear attentions. We report BLEU-4 (Papineni et al., 2002) scores as the performance metric.

Results in Table 2 indicate that the NAR-AMLP architecture achieves the best result among efficient NAR models with linear complexity. Among the NAR models, AMLP surpasses a strong linear attention variant ABC, by 3.02 BLEU on the en-de dataset and 3.64 BLEU on the de-en dataset. Compared with an even stronger ABC-based Transformer model in an autoregressive manner, our NAR-AMLP achieves less performance drop with 1.24 and 1.68 BLEU on en-de and de-en translation correspondingly.

3.3 Self- and Cross-Attention Ablation

Self-attention We evaluate the self-encoding ability of AMLP on Super Resolution (SR) task. SR aims to convert low-resolution (16×16) images into high-resolution (128×128) ones. We use a powerful backbone — SR3 (Saharia et al., 2022) and replace the softmax self-attention with the efficient architectures.[3] Following Saharia et al. (2022), we use the Flickr-Faces-HQ (FFHQ) dataset (Karras et al., 2019) for the training set and CelebA-HQ dataset (Karras et al., 2018) for the evaluation set. We use Peak Signal-to-Noise Ratio (PSNR) and Structural SIMilarity (SSIM) (Wang et al., 2004) to measure efficient models. Experiment results are shown in Table 3. AMLP improves the performance of SR3 to 23.28 (+0.10) on PSNR and 0.684 (+0.09) on SSMI against the vanilla baseline, indicating that AMLP has a strong self-encoding ability. When compared to gMLP, AMLP also has a slight performance gain. AMLP outperforms covariance-based architecture XCA by 0.20 and 0.14 on PSNR and SSMI, respectively.

[3]We use the SR3 with attention layers added after the residual blocks.

Table 3: PSNR and SSMI on CelebA-HQ dataset. n is the pixel number of the images.

Complexity	Model	Super Resolution	
		PSNR↑	SSMI↑
$O(n^2)$	vanilla	23.18	0.675
$O(n)$	local	**23.33**	0.682
	gMLP	23.24	0.679
	XCA	23.08	0.670
	ABC	22.54	0.635
	AMLP	23.28	**0.684**

Table 4: Cross-attention ablation on ETT dataset. The results are average on ETT-h1, ETT-h2 and ETT-m1 datasets. n, m are the target and source sequence lengths.

Complexity	Model	ETT	
		MSE↓	MAE↓
$O(nm)$	Vanilla	1.138	0.775
$O(n+m)$	ABC	1.147	0.809
	Performer	1.254	0.819
	cosFormer	1.219	0.823
	AMLP	**1.006**	**0.750**

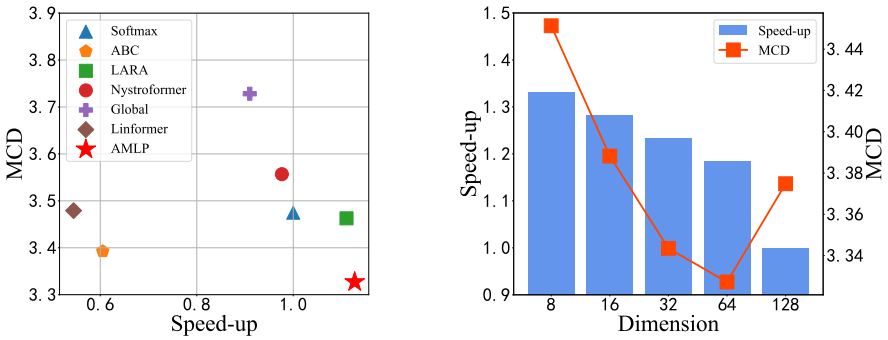
Cross-Attention We test the cross-attention ability on the long sequence time-series forecasting (LSTF) task. We take Informer (Zhou et al., 2021) as the backbone neural networks and evaluate efficient models on Electricity Transformer Temperature (ETT) dataset, which contains three sub-datasets ETT-h1, ETT-h2, and ETT-m1. We follow Zhou et al. (2021) to conduct univariate and multivariate evaluations on three sub-datasets and average their Mean Square Error (MSE) and Mean Absolute Error (MAE) to obtain final scores. Except for vanilla attention, we also compare AMLP with other three efficient models with strong cross-alignment abilities: ABC (Peng et al., 2022), Performer (Choromanski et al., 2021) and cosFormer (Qin et al., 2022). We exclude local attention as it does not work for cross attention without explicit token alignment in the time-series forecasting task. The detailed results are shown in Table 6 and the overall results on ETT are shown in Table 4. AMLP, in contrast to the vanilla counterpart, achieves lower MSE and MAE as well as more efficient complexity. Moreover, we notice that all other efficient models perform poorly compared to vanilla attention. It suggests that AMLP has a solid ability to model non-homologous information.

3.4 Ablation Study

In this section, we conduct substantial ablation experiments to dig out the efficiency and superiority of our AMLP mechanism. We first present our analysis in comparison with other efficient attention modules on the TTS task. Then we show that our approximation $c < d$ in Eq. 6 does not deteriorate the performance of speech generation. Finally, we elucidate the outstanding generation speed and GPU peak usage of our AMLP in the NAR scenario.

Comparison with Efficient Attention Recently, a surge of efficient attention algorithms also explore efficient architectures for sequence modeling, and are also proved to be effective in real-world tasks. MLP itself is efficient for vanilla attention while there have existed various kinds of novel attention mechanisms that have proved to be effective to replace self-attention modules and efficient to achieve a linear time complexity as well. We compare our AMLP with 5 efficient attention architectures of strong performance on text-to-speech synthesis task, including LARA (Zheng et al., 2022), Linformer (Wang et al., 2020), ABC (Peng et al., 2022), global attention (Luong et al., 2015a), and Nystromformer (Xiong et al., 2021). All the efficient attentions adopt reported hyper-parameters of their papers, and our AMLP uses the same setting as §3.2.

Fig. 2a shows the performance-speedup tradeoff of compared methods. Among these counterparts, AMLP and LARA are much faster than other attention mechanisms, while AMLP is slightly faster than LARA. In synthesizing speech, AMLP gains $1.15\times$ speed-up than vanilla attention, which is the most efficient module among the competitors. This fact also indicates that when sequences become longer, AMLP with linear time complexity comes to show great efficiency. When it comes to efficacy, efficient attentions like LARA and ABC can also outperform vanilla softmax attention in the TTS task. It verifies that efficient attentions clean redundant information or noise in the attention matrix, and further enhance the model representation ability. AMLP reduces the MCD with a significant margin and shows its effectiveness in generating sequences.



(a) (b)

Figure 2: (a) Performance-speedup tradeoff of various efficient architectures. Lower MCD and higher speedup indicate better performances. (b) Performance-speedup tradeoff of various intermediate dimension c values. Lower MCD and higher speedup indicate more superior models.

Intermediate Dimension Analysis The approximation of eigenvalues in Eq. 6 prompts us to know whether such approximation is feasible and whether the exorbitant approximation will deteriorate the generation performance. To this end, we test several values of c in AMLP and report each corresponding performance on TTS and the decoding speed when adopted to FastSpeech 2, in Fig. 2b. Except for c value, we adopt the same setting in §3.2.

From Fig. 2b, we can see that AMLP with approximation rank c can achieve as well as no approximation setting ($c = d = 128$) and does not impact the performance greatly. But with a lower c value, AMLP can achieve better decoding speed. Specifically, in contrast to $c = 64$, a higher MCD when setting c to d also indicates that maintaining the whole eigenvalues in Eq. 6 may even lead to overparameterization and impair the overall decoding efficacy. It verifies the feasibility to approximate $\mathbf{\Sigma}$ with fewer eigenspectrums in AMLP.

Efficiency Analysis To further understand the performance of NAR-AMLP architecture in inference, we set up a simulation experiment to test its efficiency. Fig. 3a shows that NAR-AMLP extremely speeds up the inference process. To generate a long sequence with $8,192$ tokens, vanilla NAR is $116\times$ faster than AR while NAR-AMLP is even $590\times$ faster. For sequences with more than 1500 tokens, both variants of AMLP are more efficient than vanilla attention; otherwise, the vanilla attention is faster. Fig. 3b shows that NAR-AMLP significantly reduces memory consumption in NAR generation. It saves 89% memory usage of NAR model when generating a sequence with $8,192$ tokens. Note that AR models cost fewer memory resources because of incremental decoding, which caches previous states and processes only one token at each step. But AR models still suffer from huge memory usage as NAR models in training, since they are usually implemented with a causal mask on the attention matrix. Thus it is reasonable to infer that NAR-AMLP is more efficient than AR and NAR models in training. The detailed experiment settings are present in Appendix G.

4 Related Work

Non-Autoregressive Generation Gu et al. (2018) first proposes a non-autoregressive model to generate all the tokens within a sequence in parallel, which extremely speeds up the inference process but is inferior in generation quality. To mitigate the quality degradation, many researchers devote to improve the model performance with iterative decoding (Lee et al., 2018; Ghazvininejad et al., 2019; Gu et al., 2019; Guo et al., 2020b; Huang et al., 2022), curriculum learning (Guo et al., 2020a; Liu et al., 2020; Qian et al., 2021a;b; Bao et al., 2022), latent variable modeling (Ma et al., 2019; Ran et al., 2021; Bao et al., 2019; 2022), imitation learning (Li et al., 2019b; Wei et al., 2019) and learning objective (Saharia et al., 2020; Ghazvininejad et al., 2020; Liu et al., 2022; Du et al., 2021). These previous works focus on pursuing the high efficacy of non-autoregressive generation, but few works are presented to improve NAR's efficiency in long sequence modeling. We target to further improve its efficiency and scale non-autoregressive models to long sequences.

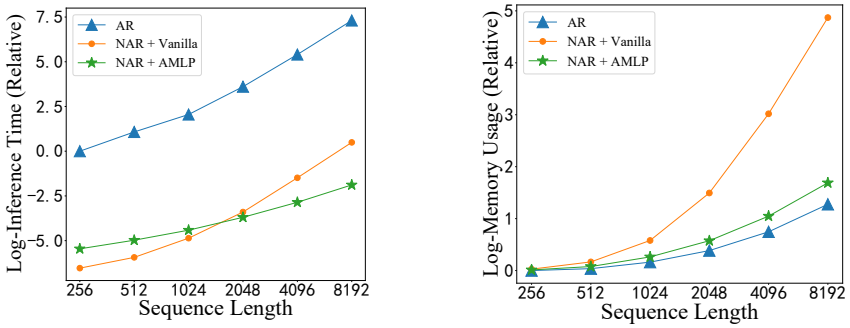


Figure 3: Empirical running time (a) and memory cost (b) with sequence length. Logarithms of relative measurement to the AR model are reported.

MLP Architecture Multi-layer perceptron (Gardner & Dorling, 1998) is a classic neural network architecture and has been widely used. Recently, novel variants of MLP architectures are proposed for text and image processing, achieving impressive results on image classification (Tolstikhin et al., 2021; Liu et al., 2021a), text classification (Tay et al., 2021), multilingual parsing (Fusco et al., 2022), and intent classification (Fusco et al., 2022). MLP-Mixer (Tolstikhin et al., 2021) is proposed by leveraging a token-mixing and a channel-mixing MLP to enable token-wise and channel-wise communication. MLP-Mixer is further improved to pNLP-Mixer with locality sensitive hashing (Indyk & Motwani, 1998) projection at the bottom calculating non-trainable fingerprints (Fusco et al., 2022). Liu et al. (2021a) propose gMLP by introducing a spatial gating unit to enhance the communication between neighboring tokens. CycleMLP (Chen et al., 2022) leverages a local window to achieve linear time complexity on dense prediction. Besides, previous studies focus on encoding text/image features with MLP, but we explore the possibility to leverage an MLP architecture for sequence generation.

Attention Mechanism Attention is first proposed to align the target and source sequence in neural machine translation (Bahdanau et al., 2015), and is further improved to multi-head self/cross/-causal attention (Vaswani et al., 2017). Due to its quadratic time complexity and memory cost with sequence length, a surge of efficient attention is proposed to improve the efficiency of softmax attention. Due to the the sparsity of attention matrix, many researchers propose to explicitly model a sparse attention mechanism to obtain fast computation without harming performance (Ho et al., 2019; Tay et al., 2020; Kitaev et al., 2020; Beltagy et al., 2020; Zaheer et al., 2020; Roy et al., 2021). The low-rank property of attention matrix also brings out matrix decomposition-based methods (Xiong et al., 2021; Chen et al., 2021). The softmax attention can also be linearized via exponential kernel decomposition (Choromanski et al., 2021; Peng et al., 2022; 2021; Zheng et al., 2022; Qin et al., 2022). These attention variants are exploring an efficient way to approximate softmax attention, but we focus on MLP architecture, which is naturally an efficient architecture.

5 Conclusions

In this work, we introduced Attentive Multi-Layer Perceptron (AMLP), an efficient plugin alternative to vanilla attention for non-autoregressive generation tasks. AMLP uses adaptive weights to learn inter-token interactions as done in attention. And we also put forward two methods adopting different philosophies to parameterize the adaptive weight matrices in AMLP. Substantial experiments on generation tasks verify that AMLP surpasses attention in most tasks and achieves similar performances with other strong efficient models in other tasks. Besides, efficiency analysis indicates that AMLP combined NAR model could save time compared to AR models, and save space compared to vanilla NAR models in long sequence settings.

REFERENCES

Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *Advances in neural information processing systems*, 34:20014–20027, 2021.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

Yu Bao, Hao Zhou, Jiangtao Feng, Mingxuan Wang, Shujian Huang, Jiajun Chen, and Lei Li. Non-autoregressive transformer by position learning. *arXiv preprint arXiv:1911.10677*, 2019.

Yu Bao, Hao Zhou, Shujian Huang, Dongqi Wang, Lihua Qian, Xinyu Dai, Jiajun Chen, and Lei Li. latent-GLAT: Glancing at latent variables for parallel text generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8398–8409, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.575. URL https://aclanthology.org/2022.acl-long.575.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pp. 12–58, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-3302.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11315–11325, 2022.

Shoufa Chen, Enze Xie, Chongjian GE, Runjian Chen, Ding Liang, and Ping Luo. CycleMLP: A MLP-like architecture for dense prediction. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=NMEceG4v69Y.

Ziye Chen, Mingming Gong, Lingjuan Ge, and Bo Du. Compressed self-attention for deep metric learning with low-rank approximation. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 2058–2064, 2021.

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J Colwell, and Adrian Weller. Rethinking attention with performers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Ua6zuk0WRH.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Cunxiao Du, Zhaopeng Tu, and Jing Jiang. Order-agnostic cross entropy for non-autoregressive machine translation. In *International Conference on Machine Learning*, pp. 2849–2859. PMLR, 2021.

Francesco Fusco, Damian Pascual, and Peter Staar. pnlp-mixer: an efficient all-mlp architecture for language. *arXiv preprint arXiv:2202.04350*, 2022.

Matt W Gardner and SR Dorling. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15):2627–2636, 1998.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 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)*, pp. 6112–6121, 2019.

Marjan Ghazvininejad, Vladimir Karpukhin, Luke Zettlemoyer, and Omer Levy. Aligned cross entropy for non-autoregressive machine translation. In *International Conference on Machine Learning*, pp. 3515–3523. PMLR, 2020.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K. Li, and Richard Socher. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*, 2018.

Jiatao Gu, Changhan Wang, and Junbo Zhao. Levenshtein transformer. *Advances in Neural Information Processing Systems*, 32, 2019.

Junliang Guo, Xu Tan, Linli Xu, Tao Qin, Enhong Chen, and Tie-Yan Liu. Fine-tuning by curriculum learning for non-autoregressive neural machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7839–7846, Apr. 2020a. doi: 10.1609/aaai.v34i05.6289. URL https://ojs.aaai.org/index.php/AAAI/article/view/6289.

Junliang Guo, Linli Xu, and Enhong Chen. Jointly masked sequence-to-sequence model for non-autoregressive neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 376–385, 2020b.

Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*, 2019.

Chenyang Huang, Hao Zhou, Osmar R Zaïane, Lili Mou, and Lei Li. Non-autoregressive translation with layer-wise prediction and deep supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 10776–10784, 2022.

Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pp. 604–613, 1998.

Keith Ito and Linda Johnson. The lj speech dataset. https://keithito.com/LJ-Speech-Dataset/, 2017.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkgNKkHtvB.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1173–1182, 2018.

Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. Neural speech synthesis with transformer network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01): 6706–6713, Jul. 2019a. doi: 10.1609/aaai.v33i01.33016706. URL https://ojs.aaai.org/index.php/AAAI/article/view/4642.

Zhuohan Li, Zi Lin, Di He, Fei Tian, Tao Qin, Liwei Wang, and Tie-Yan Liu. Hint-based training for non-autoregressive machine translation. 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)*, pp. 5708–5713, Hong Kong, China, November 2019b. Association for Computational Linguistics. doi: 10.18653/v1/D19-1573. URL https://aclanthology.org/D19-1573.

Guangyi Liu, Zichao Yang, Tianhua Tao, Xiaodan Liang, Junwei Bao, Zhen Li, Xiaodong He, Shuguang Cui, and Zhiting Hu. Don't take it literally: An edit-invariant sequence loss for text generation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2055–2078, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.150. URL https://aclanthology.org/2022.naacl-main.150.

Hanxiao Liu, Zihang Dai, David So, and Quoc V Le. Pay attention to mlps. *Advances in Neural Information Processing Systems*, 34:9204–9215, 2021a.

Jinglin Liu, Yi Ren, Xu Tan, Chen Zhang, Tao Qin, Zhou Zhao, and Tie-Yan Liu. Task-level curriculum learning for non-autoregressive neural machine translation. In *IJCAI*, pp. 3861–3867, 2020.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021b.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421, 2015a.

Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421, Lisbon, Portugal, September 2015b. Association for Computational Linguistics. doi: 10.18653/v1/D15-1166. URL https://aclanthology.org/D15-1166.

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard Hovy. FlowSeq: Non-autoregressive conditional sequence generation with generative flow. 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)*, pp. 4282–4292, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1437.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pp. 48–53, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-4009. URL https://aclanthology.org/N19-4009.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318, 2002.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. Random feature attention. In *International Conference on Learning Representations*, 2021.

Hao Peng, Jungo Kasai, Nikolaos Pappas, Dani Yogatama, Zhaofeng Wu, Lingpeng Kong, Roy Schwartz, and Noah A. Smith. ABC: Attention with bounded-memory control. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7469–7483, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.515. URL https://aclanthology.org/2022.acl-long.515.

Lihua Qian, Hao Zhou, Yu Bao, Mingxuan Wang, Lin Qiu, Weinan Zhang, Yong Yu, and Lei Li. Glancing transformer for non-autoregressive neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the*

11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 1993–2003, Online, August 2021a. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.155.

Lihua Qian, Yi Zhou, Zaixiang Zheng, Yaoming Zhu, Zehui Lin, Jiangtao Feng, Shanbo Cheng, Lei Li, Mingxuan Wang, and Hao Zhou. The volctrans GLAT system: Non-autoregressive translation meets WMT21. In *Proceedings of the Sixth Conference on Machine Translation*, pp. 187–196, Online, November 2021b. Association for Computational Linguistics.

Zhen Qin, Weixuan Sun, Hui Deng, Dongxu Li, Yunshen Wei, Baohong Lv, Junjie Yan, Lingpeng Kong, and Yiran Zhong. cosformer: Rethinking softmax in attention. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=Bl8CQrx2Up4.

Qiu Ran, Yankai Lin, Peng Li, and Jie Zhou. Guiding non-autoregressive neural machine translation decoding with reordering information. In *AAAI*, pp. 13727–13735, 2021.

Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *International Conference on Learning Representations*, 2021.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68, 2021.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1098–1108, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.83.

Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse sinkhorn attention. In *International Conference on Machine Learning*, pp. 9438–9447. PMLR, 2020.

Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. Synthesizer: Rethinking self-attention for transformer models. In *International Conference on Machine Learning*, pp. 10183–10192. PMLR, 2021.

Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Changhan Wang, Wei-Ning Hsu, Yossi Adi, Adam Polyak, Ann Lee, Peng-Jen Chen, Jiatao Gu, and Juan Pino. fairseq s^2: A scalable and integrable speech synthesis toolkit. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 143–152, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-demo.17.

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Bingzhen Wei, Mingxuan Wang, Hao Zhou, Junyang Lin, and Xu Sun. Imitation learning for non-autoregressive neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1304–1312, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1125.

Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. Nyströmformer: A nyström-based algorithm for approximating self-attention. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14138–14148, May 2021. doi: 10.1609/aaai.v35i16.17664. URL https://ojs.aaai.org/index.php/AAAI/article/view/17664.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297, 2020.

Lin Zheng, Chong Wang, and Lingpeng Kong. Linear complexity randomized self-attention mechanism. *arXiv preprint arXiv:2204.04667*, 2022.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):11106–11115, May 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17325.

A AR-BASED AMLP

We here present the detailed reason why we do not fuse AMLP with AR models.

AMLP is not suitable for AR In AR models, the self/cross attention is different from NAR models and they require to maintain causality due to teacher forcing training. To enable causal (decoder) AMLP, the overall time consumption and memory costs in the training phase are hugely increased. Such drawback is also revealed in most efficient attention (like Nyströmformer (Xiong et al., 2021), Local, cosFormer (Qin et al., 2022), Performer (Choromanski et al., 2021)) that also adopt softmax kernel decomposition for speed-up. To replace decoder self attention, they also need to pay extra complexity during training (Peng et al., 2021). Therefore, these efficient attention and AMLP are also not straightforward to be adopted for AR models.

AMLP suits NAR In contrast to AR models where typical efficient models and AMLP all suffer from inefficiency, the training objective and identically independent distribution (i.i.d.) assumption of NAR models avoid teacher forcing. NAR thus suits the application of efficient attention and AMLP. As a result, AMLP is dedicated to NAR models for even higher efficiency in both training and inference.

Example of AR-AMLP We present the specific computation steps of AMLP in AR scenario and explain the drawbacks of AR-AMLP. We take AMLP-Cov as an example. Given an query token $\boldsymbol{q}_t$, the covariances $\mathbf{S}_t^{\mathbf{Q}}$ and $\mathbf{S}_t^{\mathbf{K}}$ of $\mathbf{K}_t$ and $\mathbf{Q}_t$, and the cross-covariance $\mathbf{z}_t$ of $\mathbf{K}_t$ and $\mathbf{V}_t$, $W_{\mathbf{Q},\mathbf{K}}$ and $W_{\mathbf{Q},\mathbf{K},\mathbf{V}}$ are formulated as follows:

$$W_{\mathbf{Q}_t,\mathbf{K}_t} = \mathbf{L}_t^\top = C_q(\sigma_2(\mathbf{S}_t^{\mathbf{Q}})) + C_k(\sigma_2(\mathbf{S}_t^{\mathbf{K}})) \tag{13}$$

$$W_{\mathbf{Q}_t,\mathbf{K}_t,\mathbf{V}_t} = \mathbf{L}_t^\top \sigma_2(\mathbf{z}_t) \tag{14}$$

where $\mathbf{S}_t^{\mathbf{Q}} = \mathbf{S}_{t-1}^{\mathbf{Q}} + \boldsymbol{q}_t^\top \boldsymbol{q}_t$, $\mathbf{S}_t^{\mathbf{K}} = \mathbf{S}_{t-1}^{\mathbf{K}} + \boldsymbol{k}_t^\top \boldsymbol{k}_t$ and $\mathbf{z}_t = \mathbf{z}_{t-1} + \boldsymbol{k}_t^\top \boldsymbol{v}_t$. These computation steps increase heavy memory costs and large time consumption in the training phase, with an additional $O(ncd)$ costs beyond the overall computation. Recurrent computation also harms the parallelism and further slows down the training process.

B FROM ATTENTION TO AMLP

We here show that AMLP fuses attentive ability through kernel function approximation, matrix decomposition and reformulation.

1. In vanilla attention, $\mathrm{softmax}(QK^\top)$ is a softmax kernel which can be decomposed into a multiplication of two kernel functions: $\phi(Q)\cdot\phi(K)^\top$, which is verified in Performer (Choromanski et al., 2021), cosFormer (Qin et al., 2022) and LARA (Zheng et al., 2022). Hence, we here use a distance matrix Σ to serve as a bridge which enables $Q\Sigma K^\top$ to be decomposed with fewer eigenspectrums like Performer.

2. The low rank approximation of the attention matrix, $\mathrm{softmax}(QK^\top)$, does not impact the performance much, which is verified by Nyströmformer (Xiong et al., 2021). Based on their findings, the kernel function $Q\Sigma K^\top$ can also enjoy lower computation costs from low rank approximation.

3. Combined with two results, AMLP reformulates the attention $\mathrm{softmax}(QK^\top)$ with $Q\Sigma K^\top$ and uses Nystrom's findings to decompose Σ into two matrices. Finally, AMLP reparameterizes the decomposed matrices and maintains the attentive functionality as $\mathrm{Attn}(Q,K,V)$.

C TASK DESCRIPTION

We provide the statistics of tasks in Table 5.

Table 5: Task statistics of evaluation metrics, input sequence lengths and backbone neural networks. For encoder-decoder architectures, we show both source and target lengths. (†) denotes the sequence lengths of encoders and decoders of Tr-TTS. The number of final audio phonemes after the vocoder is 559 as well.

Task	Dataset	Length	Model	Metric
TTS	LJSpeech	559 100/141†	FastSpeech 2 Transformer-TTS	MCD/MSD
MT	WMT'14 EN-DE	23/21	Transformer & GLAT	BLEU
SR	FFHQ & CelebA-HQ	16384	SR3	PSNR/SSMI
LSTF	ETT	336/720	Informer	MSE/MAE

D HYPERPARAMETERS OF TASKS

We provide the hyperparameters of each task in Table 7. We adopt the same hyperparameter sets for Transformer-TTS and FastSpeech 2, which is referred from Wang et al. (2021). Following Vaswani et al. (2017) and Qian et al. (2021a), we use the same hyperparameters to train Transformer-base and GLAT, respectively. We implement Transformer-TTS, FastSpeech 2, GLAT and Transformer with fairseq (Ott et al., 2019). In "Evaluation Checkpoint" of Table 7, "last" denotes that we use the last saved checkpoint to test models. "best" denotes that we use the best checkpoint with the lowest MCD value in the validation set for evaluation. "average last n" denotes that we create a new checkpoint whose parameters are averaged on the parameters of last n ones and use this new checkpoint for performance comparison.

E DETAILS OF ATTENTION

Attention hyperparameters We provide the hyperparameters of attentions applied in different tasks in Table 8. local attention has one hyperparameter `wsize` to control the number of tokens around query one to be attended. ABC has one hyperparameter `landmark` to control the size of bounded memory. AMLP has one hyperparameter `ffn_dimension` to control the inner dimensionality. Performer has one hyperparameter `approx_attn_dim` to control the dimensionality of random feature matrices. For vanilla, gMLP, XCA and cosFormer, they do not adopt hyperparameters.

Implementation details We implement ABC and local attention by ourselves because we cannot find open-sourced official implementation for them. For Performer, cosFormer, and vanilla attention, we use their official implementation and rewrite them with PyTorch (Paszke et al., 2019) if necessary. These attention implementations in PyTorch will be released in `https://github.com/Anonymous`.

F EXPERIMENTAL RESULTS ON ETT DATASET

We report the complete results on ETT-h1, ETT-h2 and ETT-m1 in Table 6.

G EXPERIMENT SETTINGS FOR EFFICIENCY ANALYSIS

The simulation experiment evaluates NAR-AMLP efficiency from running time and memory usage with respect to sequence length from 256 to 8,192, compared with AR model and vanilla NAR model. We simulate the generation process with a single efficient module. For AR, we test its causal attention, which is its bottleneck in generation. For AMLP, we use 64 as the inner dimension with ReLU activation function. AMLP-Cov andAMLP-PQuery shares the same complexity, so we use

Table 6: Results on ETT-h1, ETT-h1, and ETT-m1 datasets. n, m are the target and source lengths.

Complexity	Model	ETTh1		ETTh2		ETTm1	
		MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓
Multivariate							
$O(m^2 + nm)$	vanilla	1.257	0.905	3.548	1.652	1.028	0.813
$O(n + m)$	ABC	1.332	0.934	3.430	1.585	1.037	**0.784**
	Performer	1.455	0.966	4.018	1.770	1.203	0.832
	cosFormer	1.384	0.963	3.912	1.707	1.021	0.792
	AMLP	**1.247**	**0.894**	**2.748**	**1.312**	**1.015**	0.792
Univariate							
$O(m^2 + nm)$	vanilla	**0.251**	**0.240**	0.266	0.419	0.479	0.619
$O(n + m)$	ABC	0.357	0.522	0.294	0.44	0.430	0.586
	Performer	0.267	0.440	**0.255**	**0.411**	**0.325**	**0.493**
	cosFormer	0.311	0.483	0.276	0.426	0.408	0.568
	AMLP	0.346	0.509	0.260	0.415	0.420	0.575

Table 7: Hyperparameters of different tasks.

Task	TTS	MT	SR	LSTF
Backbone	FastSpeech 2/ Transformer-TTS	Transformer/GLAT	SR	Informer
Batch Size	48	–	4	32
Number of Steps (epochs)	20K	100K/300K	1M	6 (epochs)
Warmup Steps	4K	4K	–	–
Peak Learning Rate	5e-4	5e-4	1e-4	1e-4
Scheduler	Inverse Sqrt	Inverse Sqrt	Linear	Exponential Decay
Optimizer	AdamW	AdamW	AdamW	AdamW
Adam	(0.9, 0.98)	(0.9, 0.98)	(0.9, 0.999)	(0.9,0.999)
Clip Norm	5.0	5.0	0	0
Attention Dropout	0.1	0.3	0.2	0.05
Weight Decay	0.01	0.0001	0	0
Max Tokens	–	65536	–	–
Iteration	–	–	–	5
Evaluation Checkpoint	best	average last 10	average last 5	last

"AMLP" to denote the two variants. The experiments are performed with batch size 12 on a single A100 GPU, and the results are repeated with 100 runs. We remain running latency data ranging from the first quatile and the third quatile among the 100 runs to remove noise. Finally, the remaining figures are averaged to serve as the final time consumption.

Table 8: Hyperparameters of each attention architecture.

Architecture	**Hyperparameter**	**TTS**	**MT**	**SR**	**LSTF**
local	`wsize`	15	5	15	15
ABC	`landmarks`	64	16	16	16
AMLP	`ffn_dimension`	64	16	16	16
Performer	`approx_attn_dim`	64	16	–	16

H EXPERIMENT RESULTS ON MT AND TTS

We report the performance of cosFormer and Performer on TTS (Table 9) and MT (Table 10) tasks. We do not include these results in Table 1 and Table 2 because these two models perform far behind other efficient models. We show their results here for completeness and fair comparison.

Table 9: Automatic evaluation metric on LJSpeech dataset. All models are trained by ourselves.

Arch	Model	LJSpeech	
		MCD↓	MSD↓
Complexity: $O(n^2)$ *or* $O\left((n+m)^2\right)$			
AR	Tr-TTS	4.0953	2.1985
NAR	FS2	3.4748	1.9735
Complexity: $O(n+m)$			
AR	Tr-TTS (ABC)	5.1302	2.5957
NAR	FS2 (local)	3.4189	1.9704
	FS2 (cosFormer)	3.3998	1.9561
	FS2 (Performer)	3.4374	1.9830
	FS2 (ABC)	3.3925	1.9658
	FS2 (XCA)	3.5003	2.0239
	FS2 (gMLP)	3.4025	1.9641
	FS2 (AMLP)	**3.3274**	**1.9396**

Table 10: BLEU4 scores on WMT14 EN-DE and WMT14 DE-EN dataset. All models for comparison are implemented by ourselves. Subscript figures Δ denote performance drop comparing with backbone models. "–" denotes the attention fails in this dataset.

Arch	Model	WMT' 14	
		En-De$_\Delta$	De-En$_\Delta$
Complexity: $O\left((n+m)^2\right)$			
AR	Tr	**27.38**	**31.26**
NAR	GLAT	26.24	30.10
Complexity: $O(n)$ *or* $O(n+m)$			
AR	Tr (local)	$24.77_{2.61}$	$28.21_{3.05}$
	Tr (ABC)	$25.86_{1.52}$	$29.09_{2.17}$
	w/ conv	$26.38_{1.00}$	$30.04_{1.22}$
NAR	GLAT (local)	$18.19_{8.05}$	$21.36_{8.74}$
	GLAT (cosFormer)	–	–
	GLAT (Performer)	$17.81_{8.43}$	–
	GLAT (ABC)	$21.98_{4.26}$	$24.78_{5.32}$
	GLAT (AMLP)	$25.00_{1.24}$	$28.42_{1.68}$
	w/ conv	$25.98_{\mathbf{0.26}}$	$29.61_{\mathbf{0.49}}$