The New gMLP Model and Its Implementation on Language Modeling.

Minghao Li
School of Computer Science and Engineering
Nanyang Technological University,
Singapore, 639798
MINGHAO002@e.ntu.edu.sg

Wong Siew Chien
School of Computer Science and Engineering
Nanyang Technological University,
Singapore, 639798
WONG1343@e.ntu.edu.sg

Abstract

The Transformer has been widely used in various natural language processing (NLP) tasks, including machine translation, language modeling, and question-answering. As a core component of the Transformer, self-attention has been doubted in its significance. The new **gMLP** (1) model was proposed recently, but it was only applied to BERT (2). In this report, we associate gMLP with generative pre-trained transformer (GPT) (3) to explore its effects on language modeling. The results indicate that gMLP can scale as well as Transformers on increased data and compute, but the mixed block with gMLP and self-attention has surprising effectiveness in terms of time consumption. The report also suggests that self-attention is still necessary for scaling up machine learning models, although gMLP works ideally in most scenarios. All the experimenting codes can be publically accessed at our GitHub repository[1].

1 Introduction

Natural Language Processing (NLP) is a subfield of artificial intelligence (AI) that focuses on the interaction between computers and humans using natural language. The goal of NLP is to enable computers to understand, interpret, and generate human language. This involves processing large amounts of text data and using machine learning algorithms to identify patterns, extract meaning, and derive insights from the text. NLP has a wide range of applications, including machine translation, sentiment analysis, speech recognition, chatbots, and information retrieval.

With the current surge in popularity of ChatGPT [2], the Generative Pretrained Transformer (GPT) (3) has gained widespread attention once more. The core functional component of GPT is the Transformer (4), which revolutionized NLP by using a self-attention mechanism that enables it to process sequences of variable length, making it highly effective for handling long-range dependencies in natural language.

However, Liu *et al.* doubted the necessity of self-attention modules in NLP applications of Transformers and proposed an MLP-based alternative to Transformers without self-attention, called **gMLP** (1). The gMLP simply consists of channel projections and spatial projections with static parameterization. They conducted experiments to explore various design options for this architecture, and the findings suggested that spatial projections were particularly effective with linear and paired multiplicative gating.

The new **gMLP** (1) model was proposed recently, but it was only applied to BERT (2). The report aims to understand whether gMLPs can perfectly replace the functionality of Tranformers in GPT

[1]https://github.com/liminghao0914/gpt-gmlp-experiments
[2]https://chat.openai.com/

and evaluate the features of gMLP announced by *Liu et al.* in (1). To achieve this goal, we plan to quantify the impact of incorporating the gMLP model into a basic version of GPT with the smallest parameter settings on the language modeling task. The result shows that gMLP can scale as well as Transformers over increased data and compute in GPT to deal with the language modeling task, which is similar to the outcomes in (1). However, the mixed block with gMLP and self-attention has a surprising effectiveness in terms of time consumption, which refutes the conclusion in (1). From our perspectives, self-attention is still necessary for scaling up machine learning models, although gMLP works ideally in most scenarios. With both Transformers and gMLP, models can have a better performance compared to only using the simpler spatial interaction mechanisms.

2 Related Work

2.1 Transformers

The Transformer model (4) has been instrumental in driving significant advancements in the field of natural language processing (NLP), as evidenced by breakthroughs in various NLP tasks (2; 5; 6). Moreover, Transformers have demonstrated remarkable performance in computer vision tasks (7; 8), making them an attractive alternative to Convolutional Neural Networks (9) for vision applications. This success has led to the widespread adoption of Transformer models as the default architecture in NLP, gradually replacing the Long Short-Term Memory Recurrent Neural Networks (10).

The Transformer model incorporates two key ideas: (1) an architecture that is free from recurrence, enabling it to compute the representations for each token in parallel, and (2) multi-head self-attention blocks that aggregate spatial information across the tokens. The attention mechanism (11) introduces an inductive bias that allows for the dynamic parameterization of spatial interactions based on input representations. However, it is also known that Multi-Layer Perceptrons (MLPs) with static parameterization are capable of representing any function (12). As a result, it is currently unclear whether the inductive bias introduced by self-attention is an essential factor contributing to the outstanding performance of the Transformer architecture.

2.2 Unsupervised pre-training in GPT

Unsupervised pre-training is a variant of semi-supervised learning that aims to discover an effective initialization point, rather than modifying the supervised learning objective. Earlier studies examined this approach in the context of image classification (13; 14; 15) and regression tasks (16). Later research (17) showed that pre-training serves as a regularization technique, improving generalization in deep neural networks. More recently, this method has been applied to aid in training deep neural networks for various tasks, such as speech recognition (18) and machine translation (19).

GPT is most closely related to pre-training a neural network using a language modeling objective and then fine-tuning it on a supervised target task. Dai et al. (20) and Howard and Ruder (21) employ this method to enhance text classification. However, their use of LSTM models in the pre-training phase limits their predictive capabilities to short-range linguistic structures, despite capturing some linguistic information. In contrast, GPT choice of transformer networks enables us to capture longer-range linguistic structures.

3 Approach

This section will introduce the structure and core components of gMLP and GPT. Further, we incorporate gMLP into GPT for language modeling tasks.

3.1 gMLP Model

In gMLP, a stack of L blocks with identical size and structure can be formulated as

$$Z = \sigma(XU), \quad \tilde{Z} = s(Z), \quad Y = \tilde{Z}V, \tag{1}$$

where $X \in \mathbb{R}^n$ is the token representations with sequence length n and dimension d, and σ is an activation function (e.g. GELU (22)). U and V are linear projections that operate along the channel dimension. These projections have the same shape as those found in the feed-forward networks of

Transformers, such as $\mathrm{BERT}_{\mathrm{base}}$, which are 768×3072 and 3072×768. For the sake of conciseness, we have left out any shortcuts, normalizations, and biases.

A key component in Equation 1 is $s(\cdot)$, which is a layer capturing spatial interactions. If s is an identity mapping, the transformation described above simplifies to a standard feed-forward network (FFN), where each token is processed independently without any communication or interaction between them. Hence, our primary objective is to devise an effective s that can capture intricate spatial relationships between tokens. Unlike Transformers, there is no position embeddings in gMLP because $s(\cdot)$ can capture such information. The gMLPs in the original paper strictly follow the same input and output protocol as BERT for NLP. Figure 1 shows the structure of gMLP with a spatial gating unit (SGU).

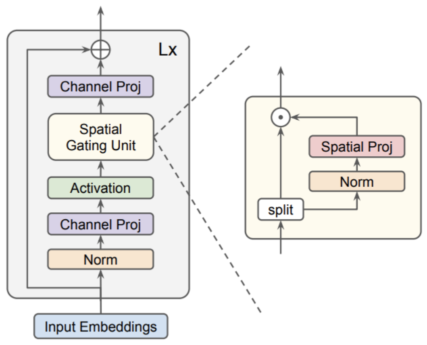


Figure 1: The gMLP architecture, which includes a Spatial Gating Unit (SGU), is presented in this overview. Unlike Transformers, gMLPs do not necessitate positional encodings and do not require the masking out of paddings during NLP fine-tuning.

3.1.1 Spatial Gating Unit

In order to facilitate interactions between tokens, it is imperative for the layer $s(\cdot)$ to include a contraction operation that spans the spatial dimension. A straightforward approach would be to use a linear projection:

$$f_{W,b}(Z) = WZ + b, \tag{2}$$

where $W \in \mathbb{R}^n$ is a matrix with the same size as the sequence length n, and b represents biases specific to each token.

In contrast to self-attention, where $W(Z)$ is created dynamically from Z, the projection matrix W in Equation 2 for spatial interactions is not dependent on the input representations. We apply the same setting as the paper does, which formulates layer $s(\cdot)$ as the output of linear gating

$$s(Z) = Z \odot f_{W,b}(Z) \tag{3}$$

where $\odot$ denotes element-wise multiplication. To ensure training stability, we discovered that initializing W with values close to zero and b with ones is crucial. This implies that $f_{W,b}(Z) \approx 1$ and consequently, at the start of training, $s(Z) \approx Z$. The initialization here guarantees that each token is processed independently in each gMLP block in Equation 1, which acts like the regular FNN at the early stage of training.

3.2 GPT Model

Similar to the policy deployed in BERT, we utilize gMLPs instead of Transformers in GPT. The GPT's architecture is illustrated in Figure 2.

In our experiments, we only employ GPT as our guide model, which has already been open-sourced at openai GitHub repository[3]. The original GPT training procedure consists of two stages. The first stage is learning a high-capacity language model on a large corpus of text. This is followed by a fine-tuning stage, adapting the model to a discriminative task with labeled data. In our approach, we only do the unsupervised pre-training to validate the effectiveness of gMLP-based GPT on language modeling procedures.

[3]https://github.com/openai/gpt-2

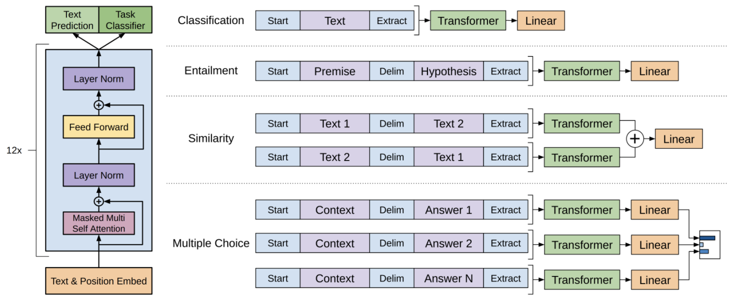


Figure 2: The **left** panel depicts the Transformer architecture and training objectives utilized in this study. The **right** panel illustrates the input transformations employed for fine-tuning on various tasks. GPT converts all structured inputs into token sequences that are processed by the pre-trained model, and this is followed by a linear+softmax layer.

3.3 Unsupervised pre-training

By utilizing an unsupervised corpus of tokens represented as $\mathcal{U} = u_1, \ldots, u_n$, we adopt a conventional language modeling approach to maximize the following likelihood:

$$L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta) \tag{4}$$

where context window has a size of k, and a neural network with parameters Θ is utilized to model the conditional probability P.

For our experiments, we employ a similar architecture as in (3), which uses a multi-layer *Transformer decoder* (4) for the language model. This transformer variant utilizes multi-headed self-attention operation over input context tokens and is followed by position-wise feedforward layers that generate an output distribution over target tokens:

$$\begin{aligned} h_0 &= UW_e + W_p \\ h_l &= \text{transformer\_block}(h_{l-1}), \forall i \in [1, n] \\ P(u) &= \text{softmax}(h_n W_e^T) \end{aligned} \tag{5}$$

where context vector of tokens is represented by $U = (u_{-k}, \ldots, u_{-1})$, n denotes the number of layers, W_e is the matrix for token embedding, and W_p is the matrix for position embedding.

3.4 gMLP-based GPT

Integrating gMLP with GPT structure, we finally obtain gMLP-based GPT as shown in Figure 3 below. The input embedding units and the output units are identical to GPT in (3). The difference here is just at the first layer in the Transformer, where we employ gMLPs with SGUs instead of masked multiple self-attention.

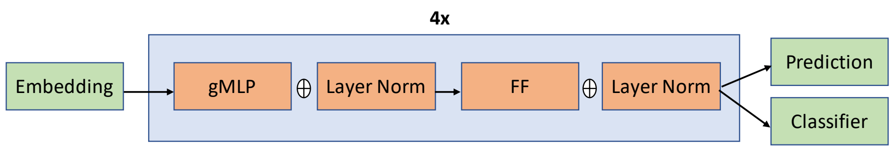


Figure 3: The gMLP-based GPT is applied in GPT_{base}.

4 Experiment

In this section, we first introduce the datasets in our experiments on doing language modeling tasks with evaluation methods (i.e. metrics). Then, we select several baseline models to validate the gMLP-based GPT. Finally, the results reveal the performance of gMLP compared to the baselines.

4.1 Task

The task of Language Modeling (23) involves predicting the next word or character in a document, which can be utilized for training language models to perform a variety of natural language tasks such as text generation, text classification, and question answering.

4.2 Data

Since we want to apply gMLP to GPT instead of using Transformers, we choose the language modeling tasks to profile the performance of the GPT models. In language modeling, there are some commonly used benchmark datasets, from which we select four in our experiemtns:

- Enwik8[4]: a benchmark dataset used in the field of natural language processing (NLP). It is a subset of the English Wikipedia, containing the first 100 million bytes of the Wikipedia dump from January 2006. The dataset consists of over 40 million words and over 800,000 unique words.
- Enwik9[5]: a similar dataset to Enwik8, but it provides a larger corpus of text data for language modeling tasks, which contains the first 1 GB of the English Wikipedia.
- Wikitext-2[6]: a collection of over 100 million tokens extracted from the set of verified Good and Featured articles on Wikipedia. Compared to the preprocessed version of Penn Treebank (PTB)[7], WikiText-2 is over 2 times larger.
- Wikitext-103[8]: a similar dataset to Wikitext-2, but with over 55 times larger data context.

4.3 Baselines

We establish baselines for our ablation studies on GPTs. Due to the limited time and the enormity of the parameters in the GPT-based neural network, we are not able to conduct a comprehensive well-designed set of experiments. These baselines are different combinations between gMLP and self-attention, which are listed in Table 1:

Table 1: Each gMLP model has $L = 4$ layers with $d_{model} = 768$, and $d_{ffn} = 2048$. ($\mathrm{GPT}_{\mathrm{origin}}$* is the model from (24), where results are used in subsection 4.6)

Types of gMLP	Parameter (M)
$\mathrm{GPT}_{\mathrm{base}}$	66M
$\mathrm{GPT}_{\mathrm{origin}}$*	112M
$\mathrm{GPT}_{\mathrm{base}}$ + gMLP	71M
$\mathrm{GPT}_{\mathrm{base}}$ + gMLP + attn	72M

4.4 Evaluation Method

We evaluate the performance of gMLP-based GPT on enwik8, enwik9, wikitext-2, and wikitext-103.

One key metric score is employed for the enwik8 and enwik9 datasets, *bits per character (BPC)*. BPC measures the number of bits needed to encode each character in a text sequence, given the probability distribution generated by the language model. BPC is commonly used in text compression tasks,

[4]http://prize.hutter1.net/dc/enwik8.zip
[5]http://mattmahoney.net/dc/enwik9.zip
[6]https://blog.salesforceairesearch.com/the-wikitext-long-term-dependency-language-modeling-dataset/
[7]https://catalog.ldc.upenn.edu/docs/LDC95T7/cl93.html
[8]https://blog.salesforceairesearch.com/the-wikitext-long-term-dependency-language-modeling-dataset/

Table 2: Training parameters for GPT models.

attribute	value	attribute	value
batch size	64	total steps	100000
hideen dropout	0.1	GELU dropout	0
optimizer	Adam	weight decay	0
β_1(Adam)	0.9	β_2(Adam)	0.999
Adam ϵ	1e-8	Learning rate	2e-4

where the goal is to compress a text sequence using as few bits as possible. A good language model can help to reduce the number of bits needed to encode the text sequence, leading to a more efficient compression scheme. The BPC score formulation is:

$$\begin{aligned} bpc(string) = \frac{1}{T}\sum_{t=1}^{T} H(P_t, \hat{P}_t) &= -\frac{1}{T}\sum_{t=1}^{T}\sum_{c=1}^{n} P_t(c)\log_2 \hat{P}_t(c), \\ &= -\frac{1}{T}\sum_{t=1}^{T} \log_2 \hat{P}_t(x_t). \end{aligned} \tag{6}$$

where T is the length of your input string.

The lower the BPC score, the better the language model's performance, as it indicates that the language model is better at predicting the next character in the text sequence.

For wikitext-2 and wikitext-103, the key metric for these datasets is perplexity.

Perplexity measures the level of uncertainty or "surprise" of a language model when predicting the next word in a sequence of words. A lower perplexity score indicates that the language model is better at predicting the next word, while a higher perplexity score indicates that the language model is less certain or accurate in its predictions. Perplexity is calculated as the exponential of the cross-entropy loss between the predicted probability distribution of the language model and the true probability distribution of the test data:

$$\begin{aligned} PP(W) &= \frac{1}{P\left(w_1, w_2, \ldots, w_N\right)^{\frac{1}{N}}} \\ &= \sqrt[N]{\frac{1}{P\left(w_1, w_2, \ldots, w_N\right)}} \end{aligned} \tag{7}$$

4.5 Model Settings

In our experiment settings, we set all four models with the same hyperparameters, which are shown as Table 2:

All experiments were conducted on a machine NVIDIA DGX-2 that consists of four graphic cards NVIDIA Tesla V100 SXM2 32GB, and CPU Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz. All experimental programs were implemented in PyTorch 1.12.0 with CUDA 10.2 enabled.

4.6 Results

Since we have three models across four datasets, there are 12 experiments in total. It only works for the comparison of GPTs, while we did not do any hyperparameter tunning jobs due to the tight deadline. The training results of these 12 experiments can be concluded in the following Table 3.

Besides, Table 4 illustrate several examples of the decoded output of the three models in subsection 4.4 and the difference along the increasing number of epochs.

5 Analysis

First, it was brought to our attention that gMLP is likely to make the overfitting of each gMLP-based model happen on the wikitext2, according to the Figure 4. It may be caused by the SGU's sigmoid

Table 3: GPT with different combinations of gMLP and attention. The influence of gMLP in GPT model during the unsupervised pretraining period.

Case	Model Type	Dataset	Metric Score (Best/Final)		Computation
			BPC	Perplexity	
#	T	D	bits per character	PP(W)	time per step
#1	GPT_{base}	enwik8	2.48/2.48	43.3	12.91
#2	GPT_{base}	enwik9	2.42/2.42	42.7	14.05
#3	GPT_{base}	wikitext2	2.14/2.14	40.6	14.35
#4	GPT_{base}	wikitext103	2.27/2.28	40.5	13.45
#5	GPT_{base} + gMLP	enwik8	1.10/1.11	33.6	35.01
#6	GPT_{base} + gMLP	enwik9	1.19/1.25	34.5	35.07
#7	GPT_{base} + gMLP	wikitext2	1.02/2.39	32.7	34.89
#8	GPT_{base} + gMLP	wikitext103	0.95/0.96	31.9	34.71
#9	GPT_{base} + gMLP + attn	enwik8	0.99/1.00	31.4	35.37
#10	GPT_{base} + gMLP + attn	enwik9	1.19/1.25	34.3	36.03
#11	GPT_{base} + gMLP + attn	wikitext2	1.01/2.30	32.5	35.13
#12	GPT_{base} + gMLP + attn	wikitext103	0.95/0.95	31.6	35.80

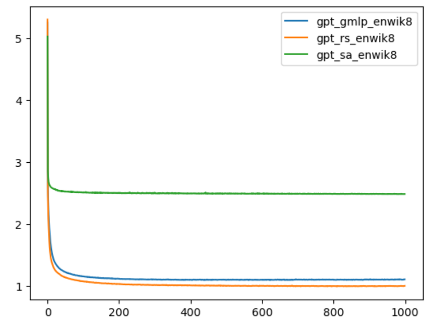
Table 4: The outputs of GPT_{base}, GPT_{base} + gMLP and GPT_{base} + gMLP + attn on textwiki2 dataset.

Epochs	GPT_{base}	GPT_{base} + gMLP	GPT_{base} + gMLP + attn
1	isheowcod d gomane lupe ushanta talmero <, tcing . wonidito " rr ily .	icoas te ftisowiof th h me e reank ag , ale ceg , watowes oninf nctseld forarg anshalens	e heralm . phinding <unk >unchevitheasy unantizy seallat bre orruth brled in wel <alors ud
10	it cofositovith @ s c stous d ar asescale des <eos><eam unk >thin the peen ,	e car a <unk >, games were all <unk > , included incomposed a noved in novision	e song was consructed to image . these defensal loymentaps of commaigneries city and martinally in the worldhotzam
100	en hend . ath 197 mancanoued m s runeve by tialesopann mp spe a	e face is little that were she distinguished several demonstrations of the frelimo , a mid	e scene made 29 cars fisa made in the michigan between the club and the <unk > series debuted and previously
500	e , or decontstu tin the wheghs ombe twe athe s , gns there a he , beoptoose	e motorway still hurricane convinced mi , john disapproved at the time of their meeting	e <unk >has <unk >rolled not a raft with carlsmacles that seated should be retro @-@ got ; he received
1000	ance cse 2 @- @ g ghincallotofal 20s as th thimoneatl at d sin	e hollywood reporter 's personalist <unk >, responded issues control of the sinai peninsula	e atlanta also on a record complete passes that ever technique were achieved at a <unk >. <eos>teibrary between mostly through

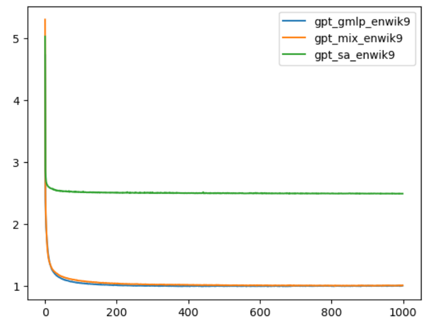
gating functions, which strongly affect the performance of gMLP-based GPTs. So, when we remove the SGU units and replace them with self-attentions in Transformers, there's no more overfitting as all loss curves steadily go down.

Second, the testing results of three models on the same dataset, enwik8 in Table 3, indicates that GPT with gMLP can perform much better than only using attention in GPT with 62M parameter. In (24), GPT with 117M parameters can achieve a 1.16 BPC score on enwik8 at best. While in our scenario, gMLP-based GPT with 71M parameters even achieves a 1.10 BPC score, which is far more lower than the original GPT with a larger network. Also, we can see from Table 3, when using mixed block type (i.e. GPT_{base} + gMLP + attn) with 1M more parameters can enhance the BPC score from 1.10 to 0.99 and only more 0.36s per hundred steps on enwik8 dataset, from 1.03 to 1.01 and only more 0.24s per hundred steps on wikitext2, and from 0.99 to 0.95 and only more 0.43s per hundred steps on wikitext103. In other words, the mixed block type of GPT can better handle language modeling tasks as the unsupervised pere-training learner with only subtle time consumption.

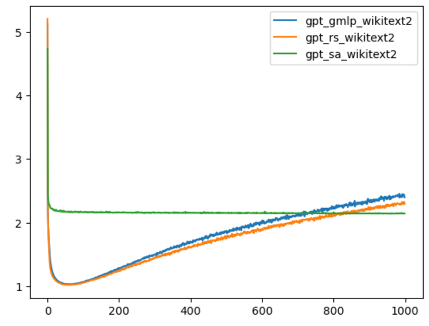
In addition, when training $\mathrm{GPT}_{\mathrm{base}}$ models, the **computational cost** is relatively low compared to other models with gMLP. The time per step metric was not included in (1). Although experiments have illustrated the effectiveness of gMLP in reducing the parameters in large language models, it somehow increases the training time. The side effects of gMLP, neglected by Liu et al., cause us to be suspicious of whether gMLP can perfectly replace the attention-based structures in Transformers.



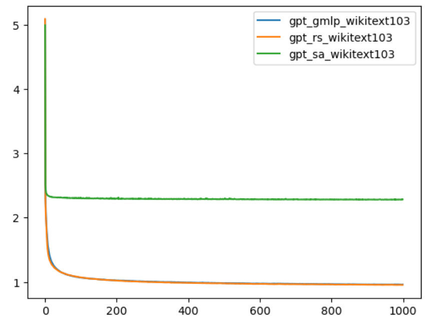
(a) Entropy loss on enwik8



(b) Entropy loss on enwik9



(c) Entropy loss on wikitext2



(d) Entropy loss on wikitext103

Figure 4: Testing losses of four datasets, including enwik8, enwik9, wikitext2, and wikitext3.

6 Conclusion

In this report, we associated gMLP with GPT to handle language modeling tasks on multiple datasets. The results showed that gMLP could scale as well as Transformers on increased data and compute. The mixed block with gMLP and attention was more effective than only gMLP-based GPT in terms of time consumption. Our report also concluded that gMLP seems like to be causing over-fitting in some scenarios that it cause the testing set loss to grew significantly after certain epochs. Further more, gMLP can also perform as good as self-attention with much lesser parameters in most scenarios. However, our experiments also found that gMLP has higher computational cost during training compare to self-attention. While gMLP works ideally in most scenarios, the report suggests that self-attention is still necessary for scaling up machine learning models. We doubt the significance of gMLP and emphasize that the effect of Transformers can not be replaceable. In conclusion, mixed blocks of gMLPs and attentions perform better in GPT for language modeling tasks. Future research could explore the other part, supervised fine-tuning, with gMLP-based models. Further, model tuning can also investigate the effects on different settings of GPT.

References

[1] H. Liu, Z. Dai, D. So, and Q. V. Le, “Pay attention to mlps,” in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 9204–9215. [Online]. Available: https://proceedings.neurips.cc/paper/2021/file/4cc05b35c2f937c5bd9e7d41d3686fff-Paper.pdf

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[3] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, “Improving language understanding by generative pre-training,” 2018.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.

[5] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, “Xlnet: Generalized autoregressive pretraining for language understanding,” *Advances in neural information processing systems*, vol. 32, 2019.

[6] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.

[8] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, “Training data-efficient image transformers & distillation through attention,” in *International conference on machine learning*. PMLR, 2021, pp. 10 347–10 357.

[9] M. Tan and Q. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

[10] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.

[12] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[13] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[14] M. Ranzato, C. Poultney, S. Chopra, and Y. Cun, “Efficient learning of sparse representations with an energy-based model,” *Advances in neural information processing systems*, vol. 19, 2006.

[15] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 1096–1103.

[16] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, “Greedy layer-wise training of deep networks,” *Advances in neural information processing systems*, vol. 19, 2006.

[17] D. Erhan, A. Courville, Y. Bengio, and P. Vincent, “Why does unsupervised pre-training help deep learning?” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 201–208.

[18] D. Yu, L. Deng, and G. Dahl, “Roles of pre-training and fine-tuning in context-dependent dbn-hmms for real-world speech recognition,” in *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning*. sn, 2010.

[19] P. Ramachandran, P. J. Liu, and Q. V. Le, “Unsupervised pretraining for sequence to sequence learning,” *arXiv preprint arXiv:1611.02683*, 2016.

[20] A. M. Dai and Q. V. Le, “Semi-supervised sequence learning,” *Advances in neural information processing systems*, vol. 28, 2015.

[21] J. Howard and S. Ruder, “Universal language model fine-tuning for text classification,” *arXiv preprint arXiv:1801.06146*, 2018.

[22] D. Hendrycks and K. Gimpel, “Gaussian error linear units (gelus),” *arXiv preprint arXiv:1606.08415*, 2016.

[23] R. Jozefowicz, O. Vinyals, M. Schuster, N. Shazeer, and Y. Wu, “Exploring the limits of language modeling,” *arXiv preprint arXiv:1602.02410*, 2016.

[24] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, “Language models are unsupervised multitask learners,” *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.