Generative Deep Models (Northwestern University) **Fall 2022**

Paper 1: Attention Is All You Need

Authors: Ashish Vaswani et al. from Google Brain **Published at:** NIPS 2017

Brief Summary

In this paper [13], The authors first discussed the limitations of the existing RNN-based Encoder-Decoder models. The main point is that the recurrent network models have a huge bottleneck of information loss. Even with the attention mechanism, the inherent architecture of the recurrent models encourages the network to lose a lot of information. To alleviate this performance constraint, the authors of this paper proposed a new technique where attention will be used, but without recurrent networks. They named it 'Transformer'. The Transformer model is based on two fundamental design principles/architecture blocks as discussed below.

First, the authors mentioned that across sequential models, the most important part is putting attention to the surroundings (at least what happened in the seq2seq model at that time). However, having a recurrent network to propagate the sequence is not that important. Instead, the sequence should look at 'all surroundings' every time. The authors named this attention as a 'self-attention' block. Here, the Key K, Query Q, and Value V are projected using weight vectors, and then Q and K are multiplied to compute the attention matrix. Then, softmax is applied to that matrix, followed by multiplication with V to find the most important information. Then the output is further passed through the feed-forward layer for output. This strategy is used to design the basic attention block.

Second, they utilized this attention block to build the encoder and decoder. Basically, an encoder is a stack of self-attention blocks I discussed in the earlier paragraph. The decoder is also built with the same block. Output from each encoder layer is added to the respective decoder layers to pass all attention information to the decoder. The decoder finally provides sequence output.

With all these methods, there was a major problem: maintaining sequence. To gain context, sequence input is necessary. For that reason, they introduced sinusoid positional encoding that produces multiple curves of different wavelengths to feed the data. They further extended the architecture with multi-head attention, which is basically the identical transformers with different attention weights and works as a voting mechanism when selecting the best value.

Strengths and Critiques

The paper has done a great job of replacing recurrent networks and building an attention-only model. This method revolutionized deep learning, mainly natural language processing (NLP). Having longer context became easier than RNNs. That advantage eventually resulted in higher performance in NLP benchmarks. Multi-Head attention was also a remarkable innovation.

Critique: The position encoding mechanism is not that intuitive. No guarantee of context capture.

Generative Deep Models (Northwestern University) **Fall 2022**

Paper 2: Self-Attention with Relative Position Representations

Authors: Peter Shaw et al. from Google and Google Brain **Published at:** NAACL 2018

Brief Summary

Usually, the vanilla Transformer models [13] use positional encoding to propagate a sequence using a series of sinusoidal waves. There are multiple periodic waves with different wavelengths. The polarity of the wavelengths is used to pass the sequence for self-attention. However, this is a fragile process for two fundamental reasons. First, there is no clear reason why this encoding will do well in any kind of sequence learning task. Second, this encoding process has no guarantee of learning 'all' context surroundings at the same time. To solve these problems, the authors of this paper [10] proposed a new mechanism of representing positions to the self-attention blocks as follows.

The first solution is in the input to the self-attention block. The authors proposed adding a list of input embeddings that will be represented during the transformer output representation. The embeddings are a group of vectors that mainly estimate the relative positioning distance between two sequences, say p and q. They name this process Relative Position Representation (RPR). For example, in a sequence of 5, a total of 9 input vectors will be provided. 1 will be for the current word, 4 to the left, and 4 to the right. One important property of RPR embeddings is that these are not re-projected like the key, query, or value vectors. They are shared across all attention heads. But, they are not shared across the layers.

The second trick they introduced is to limit the context window. In their RPR, they say that to save the memory; they clip the RPR window to a fixed number of length k. So, the number of total sequences surrounding a word is $2K + 1$. The authors pointed out two main reasons behind choosing this mechanism. First, they said that position information is not useful enough after a certain distance. Second, if they clip in this way, the model can normalize to sequence length not seen during training. They also discussed their efficient RPR implementation technique.

Strengths

Some authors of this paper and Transformer were the same. They compared their technique with the standard Transformer model. They performed remarkably in the BLEU score evaluation (for the translation tasks). I also loved the clear explanation of the proposed method.

Major Critiques

I don't have any major critique. But it still has the issue of memory overflow. Because new vectors are needed to enable relative positioning, which takes more memory and CPU, can this be optimized even better?

Generative Deep Models (Northwestern University) **Fall 2022**

Paper 3: Music Transformer: Generating Music with Long-Term Structure

Authors: Cheng-Zhi Anna Huang et al. from Google Brain **Published at:** ICLR 2019

Brief Summary

This paper [6] is about generating state-of-the-art music using the Transformer [13] architecture. The authors of the paper mainly focused on two major problems.

First, the problem of tracking the regularity in a music sequence. In earlier research, it was harder to track the ordering, periodicity, etc. The authors of the paper utilized the 'Relative Attention' technique that focuses on the relation between the music tokens. However, the existing relative attention technique takes higher memory for longer sequences. The authors solved the problem by proposing a better 'skewing' algorithm. More, specifically, the existing relative attention takes $O(L^2D)$ memory complexity, where L is the length of the sequence and D is the feature vector dimension. Their proposed approach reduces the space to $O(LD)$. In addition, they are the first ones to use relative local attention to the music domain.

Second, the ability to generate long music sequences. The prior transformer models used higher memory, thus, could not generate long-range music. This implementation trick enabled training on minute-range audios. Thus, the newly generated audios are more meaningful compared to the vanilla Transformer model. They grounded this by evaluating their output using expert people in the music domain.

Strengths

The paper has a number of good points according to my opinion. First, they explored a massive amount and variations of music data to claim their performance improvement. Second, the long-sequence generation was a major problem. Their memory optimization trick became very handy for relative attention, which is also a big breakthrough according to my opinion. In addition, they performed a qualitative evaluation by experts to make their claims on better music generation even stronger. I personally went through their webpage and found their model's generated music very pleasing compared to the original Transformer model.

Major Critiques

Although the paper has performed really well on music, I am more curious about generating real songs. For example, can their approach handle generating realistic songs in a certain language, e.g., English? Also, will it be able to generate complex vocabulary, i.e., mixture of multiple instrument sounds? These are some instant thoughts that came to my mind.

Generative Deep Models (Northwestern University) **Fall 2022**

Paper 4: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Authors: Jacob Devlin et al. from Google AI Language **Published at:** NAACL 2019

Brief Summary

This paper [4] extended the existing uni-directional Transformer [13] by introducing Bi-Directionality, which eventually helped to learn richer natural language context. To be more specific, the paper focused on two major problems we see in the natural language.

First, learning the embedding of sentences has been a very challenging task before. The state-of-the-art language models use an encoder-decoder architecture to encode sentence information and decode the next word(s) in an autoregressive fashion. However, this is computationally expensive. The authors of the BERT paper proposed a solution to tackle this problem. First, they removed the decoding layer of the vanilla Transformer model. Instead, they build a bi-directional self-attention strategy to train on the language data without supervision.

Second, there have been works on how to represent inputs for unsupervised language model learning. The authors of the paper came up with two interesting approaches. First, instead of predicting the next word, they randomly masked a few words from sentences and expected the model to predict them. The main intuition behind this is to ensure that their proposed bi-directional self-attention learns the context words surrounding that target word. Then, they trained their model on next-sentence prediction tasks. In that case, they used question-answer (QA) datasets.

Solving these two problems, they claimed that their method is very general in learning sentence representations and can be used in any downstream tasks using the embeddings.

Strengths

I found this paper with very strong motivations. Most unsupervised language models use next-word predictions. This paper first came up with another novel method. This method is also very much connected with the bi-directional self-attention mechanism they proposed. I believe, they first came up with the training strategy, which eventually led them to design the bi-directional self-attention technique. Overall, their approach is very powerful and they proved it by strong benchmark results.

Major Critiques

I have no particular critiques of this model except for the computational complexity and data requirements. There is room for biases in the model as it is solely trained only on the internet data. Also, the context it is learning is also based on how much data it gets in one scenario. Making sure the models don't get biased to contexts is a major challenge.

Generative Deep Models (Northwestern University) **Fall 2022**

Paper 5: GPT-3: Language Models are Few-Shot Learners

Authors: Tom B. Brown et al. from OpenAI **Published at:** NeurIPS 2020

Brief Summary

This paper [1] proposed a high-performing language model. The main contribution of this paper is two-fold.

First, the authors proposed a novel learning technique. Their method enabled the model to learn from less labeled data. They introduced zero-shot, one-shot, and few-shot learning techniques to capture the core information from the data. In zero-shot, no context info was given. In the one-shot, only a single sentence output was given for a single sentence input, e.g., a translation task. For a few-shot, an answer (consisting of one or a few sentences) is provided for a given question prompt. For their model, these data are portrayed as 'tasks'. This multi-task setting encourages the model to be more general across tasks, which GPT-3 does! It can be applied to most common natural language QA systems.

Second, a MASSIVE model size. To compare. the earlier discussed language model, RoBERTa [9], a variation of BERT had 355 Million parameters, whereas, GPT-3 has 175 Billion total parameters. Compared to RoBERTa, it is a massive model. This was, so far, the largest neural network ever built in early 2021. The big advantage of having more parameters is more memorization of the contexts. For this reason, GPT-3 model could learn about the tasks very easily.

In addition, they added 'word manipulation' in the training strategy. This method can be helpful in many ways. For example, it can handle typing mistakes directly from the model.

Strengths

The biggest strength of GPT-3 is its versatility. GPT-3 was trained on zero-shot, one-shot, and few-shot data. For this reason, this model had a very competitive performance in many language generation tasks. Before GPT-3, the language models were not so versatile across the tasks. However, GPT-3 can answer questions, do translations, generate big paragraphs, etc. I believe, this is the biggest advantage of this massive model.

Major Critiques

There are a number of concerns about this model. First, it is trained on public data sources. As the model is overparameterized, it is very likely that it will generate plagiarized content, i.e., content directly from trained data. Second, I am not sure how realistic the sentences would be because it doesn't have a connection with real word facts. Overlapping sentences can bias this model easily.

Generative Deep Models (Northwestern University) **Fall 2022**

Paper 6: Quantifying Memorization Across Neural Language Models

Authors: Carlini et al. from Google Research, UPenn, and Cornell **Published at:** NA

Brief Summary

Deep Learning with modern architecture, e.g., Transformer, enabled the language models to learn representations about sentences robustly. This improvement is very promising as we can widely use language models now in life. However, most of the models are overparameterized. They have a massive amount of data and a massive parameter to 'memorize' the data. Some latest models, e.g., GPT-3 have 175 Billion parameters to memorize the language data. The authors of this recent paper [2] raised concern about investigating the memorization of the latest SOTA language models. They performed the experiment in two steps.

First, they have defined what 'memorization' implies. To do that, they defined a formal procedure of token matching. We know that the language model generates the next word(s) given a prompt, i.e., a set of previous words. Formally, $[p||s]$ means, s is being generated based on p context words. The authors, pick both s and p randomly (they discussed the process in the paper) and prompted the model with p. If the model generates s, which is the identical data in the training set, they call it 'memorization'. They have selected different number in p and s to show how lengthy memorizations can be for the models. With this setup, they benchmarked the SOTA language models

In the second step, they claimed that within the same model family, e.g., GPT-3, larger models memorize more data than the smaller models. To prove their claim, they picked the models with different parameters and evaluated them with their pre-generated evaluation data mentioned in the first step. Then, they have shown that larger models (with billion parameters), e.g., GPT-Neo tend to extract more information from the training data. Moreover, larger models tend to memorize very high sequence lengths. More importantly, the more repetition in the data, the higher the memorization rate among the models.

Strengths

I personally found this paper very interesting as this paper addressed the concern that I had in my mind. They evaluated the models from various angles, e.g., fraction extractable, how long sequence the models can memorize, and the impact of data repetitions. This paper will surely drive further research direction on designing privacy-preserving language models to protect personal information.

Major Critiques

I don't have any critiques on the paper. However, I believe, they could incorporate more language models in their experiments so that readers could get more insights about them as well.

Generative Deep Models (Northwestern University) **Fall 2022**

Paper 7: Image GPT: Generative Pretraining from Pixels

Authors: Mark Chen et al. from OpenAI **Published at:** ICML 2020

Brief Summary

GPT models have been very successful for natural language processing tasks. GPT-3 with around 175 Billion parameters is very general and can almost answer any queries, do translations, etc. The authors of this paper [3] tried to build GPT for images too. Language and pixels have one fundamental difference. Language data are sequential (mostly temporal), however, image data has spatial correlation, which Convolutional Neural Network captures very well. However, the authors tried to apply the GPT training strategy using pixels. Their main motivation was to learn unsupervised image representation. They followed very interesting steps that I will discuss below.

First, Current CNN-based models require many training samples to do transfer learning. The authors' motivation behind unsupervised learning was to enable Image GPT as a unified image representation system to do many downstream tasks, which is very difficult for CNN-based models. The first burden they faced is the image size. A regular color image with 200×200 dimensions will result in 120000 values, which is pretty big to represent a single image. So, they performed context reduction that took the new image dimension to 32×32, thus total value to 3072. They additionally, clustered the color space of the dataset and picked 9 bit custom color space to reduce the final dimension to have 1024 values.

Second, they looked at image generation as an autoregressive problem. The inputs will be the prior pixels and the output will be the next pixel. The initial pixel is the top-left, and from that further images are generated. To design the loss function, first, they flattened the image, passed through the decoder, then reconstructed it back to the original dimension. The second loss they focused on is similar to BERT [4]. Similar to BERT, they masked some pixels randomly and expected the model to fill up those pixels with the right values. Combining autoregressive loss and BERT loss they trained the model.

They further introduced linear probing to initiate the generation of images based on the pixel distribution of each image type. This method helped them to evaluate the performance of the models. They also discussed context reduction to reduce memory usage.

Strengths and Critiques

The paper is powerful in terms of method. They generated images without the CNN method, e.g., GANs. They also did a detailed evaluation. This is surely a remarkable breakthrough.

No direct critiques. There are still a few issues to re-study the model. For example, they found out that immediate layers tend to have the best results based on linear probe accuracy.

Generative Deep Models (Northwestern University) **Fall 2022**

Paper 8: Reformer: The Efficient Transformer

Authors: Kitaev et al. from U.C. Berkeley and Google Research **Published at:** ICLR 2020

Brief Summary

Transformer [13] architecture is very successful mostly in Natural Language Processing for holding robust attention in the documents. However, calculating the attention vectors are very time-consuming. Vanilla Transformers take $O(N^2)$ for a sequence of length N. The memory issue is also a big burden for Transformers because currently, it has to hold all activation outputs. So, it increases by $O(N^2)$ for each layer increment. These issues limit transformer models to work on limited sentences even with low batching. To solve this problem, the authors of this paper [8] proposed two techniques to get rid of the computation burden and make Transformers efficient.

To solve the first burden, they first looked at the current attention matrix calculation strategy. They first claimed that key-query self-attention mechanism performance is almost similar to query-query self-attention. Because of this phenomenon, they introduced hashing the attention matrix. They pointed out that, query-query attention generates a very sparse matrix. But, we only need the top values for learning about the context. Thus, if we can hash the attention matrix into multiple 'buckets', then we don't have to scan through the full matrix. For this, they proposed a technique for the Locality-Sensitive Hashing (LSH) mechanism to have multiple buckets to hold the attention vectors. The close vectors are grouped together, so, it is highly probable that the vectors with higher softmax values will fall into a single bucket. This approach made the complexity into $O(N \times b)$, where b is the bucket size. As the bucket size is constant, the time complexity is $O(N)$. However, $O(NlogN)$ is needed to build the hash bucket.

The second burden was about memory. They utilized existing Reversible Residual Layers (RevNets) [5] that can generate both forward and backward weights by only using the last activation layer weights. The same strategy is used in their Reformer architecture, which no longer required to store all activation data in the memory. They are computed on demand.

They further discussed 'chunking' to parallelize the model training across GPUs. Here, they split Q in such a way that shared computations became possible.

Strengths and Critiques

The biggest strength of the paper is to bring two separate concepts into Transformer to make it way more efficient. The immediate result is that now Transformers can be used to train on a very large number of tokens. The authors said that is can learn an entire novel with a single GPU. The efficiency will surely motivate people to use Transformers on low-end devices.

Critique: Hashing does not guarantee that all similar softmax vectors will be in the same group. Coming up with an idea that can guarantee they fall into the same group would be very helpful.

Generative Deep Models (Northwestern University) **Fall 2022**

Paper 9: Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention

Authors: Katharopoulos et al. **Published at:** ICML 2020

Brief Summary

Transformer models [13] are very effective in many Machine Learning tasks. However, the only is computational complexity. Reformer [8] tried to optimize Transformer from $O(N^2)$ to $O(NlogN)$. The authors of this paper [7] looked at Transformers differently. First, they proposed a solution to linearize the Transformers, followed by claiming that Transformers work like RNNs.

In the first stage, the authors split the $O(N^2)$ attention vector calculation strategy into multiple parts. They generalized the calculation using $sim()$ function. In Transformer's case, it is exp (more details in the paper). After splitting, they focused on linearizing the attention procedure. To do that, they kernelized the vectors using $\varphi(x)$ and showed that the computation can be performed in $O(N)$ instead of $O(N^2)$. The Kernel is essentially a feature mapping function. They claim that their way of splitting and kernelization enables reusing the key K and value V with each row of Q. This helped reduce memory and over-computation during attention calculation. They then discussed how this mechanism can also be applied to causal masking, followed by linear-time and constant-memory gradient calculation techniques.

After designing Linear Transformer, they claimed that the property of Transformers follows RNNs if causal masking is used. To demonstrate, they formulated Transformers as RNNs, e.g., design input, memory, and output computation strategy. Using that formula, they trained the model in multiple tasks and compared the benchmark results with the Reformer paper. They showed that their runtime computations are lesser than the Reformer while having almost similar performance.

Strengths

The biggest strength of this paper is their derivation of the model. They have re-designed the original Transformer paper and got rid of the explicit Softmax computation with Kernels. This method let them train Transformers in almost linear time. Their method will surely be helpful to popularize Transformers and training in even bigger sequences than Reformer did.

Major Critiques

How the authors built the positional encoding strategy is not clear. Also, I am not sure how their RNN structure derivation will work for multiple transformer layers. For example, it is not clear from their paper how a third layer of an RNN Transformer can attend to the sequences of the first layer in a 3 layer architecture. In fact, the authors have not provided any evaluation results for multi-layer Transformer RNNs. It will be worth exploring this more.

Generative Deep Models (Northwestern University) **Fall 2022**

Paper 10: WaveNet: A Generative Model for Raw Audio

Authors: Aaron van den Oord et al. from Google DeepMind **Published at:** Arxiv

Brief Summary

This paper proposes WaveNet [11], a deep learning architecture for audio synthesis. The paper generates realistic sound/music based on speaker conditioning (identity). The fundamental architecture of their network is discussed below.

First, their model follows an autoregressive model. In fact, their architecture is very close to PixelCNN [12] model. PixelCNN is based on stacked convolutional layers without the pooling layers. The output dimension of PixelCNN is the same as the input. WaveNet made one modification to PixelCNN in masking. As their model has conditional dependence, e.g., speaker conditioning, instead of masking, they use causal dilated convolutions. Causal convolutions are better than RNNs as they are faster to compute due to having no recurrent connections. However, they require larger receptive fields, thus dilated convolution (skipping some nodes) becomes helpful.

Second, they have kept the activation function calculation the same as PixelCNNs, however, they modifier Softmax computation. As audio is 16-bit integer value, $2^{16} = 65,536$ will be possible for softmax. The authors reduced the dimension to 256 by applying the transformation formula.

To add conditioning, WaveNet uses two steps: Global and Local. In global conditioning, h influences the output distribution by being an embedding for each speaker. In local conditioning, the authors use transposed convolutional network to map to a new time series with the same audio signal resolution. With this approach, the authors benchmark the model for multi-speaker speech and text-to-speech generation.

Strengths

The success of the model is generating natural sound. In the benchmarks, WaveNet did a remarkable performance generating natural speech from text. Also, the dataset it was trained on had 109 speakers. WaveNet's conditioned model did a remarkable job of generating different speaker voices. Back then, it was a revolution in the speech domain.

Major Critiques

Although the paper did a fantastic job of generating natural speech and music, it emphasized the higher receptive field. However, according to dilated casual convolution technique, the receptive field growth is exponential with the number of layers, a major drawback for deeper network design. Also, the music generation was not long-term coherent, probably due to the lack of attention.

References

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[2] N. Carlini, D. Ippolito, M. Jagielski, K. Lee, F. Tramer, and C. Zhang. Quantifying memorization across neural language models. *arXiv preprint arXiv:2202.07646*, 2022.

[3] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR, 2020.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[5] A. N. Gomez, M. Ren, R. Urtasun, and R. B. Grosse. The reversible residual network: Backpropagation without storing activations. *Advances in neural information processing systems*, 30, 2017.

[6] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck. Music transformer. *arXiv preprint arXiv:1809.04281*, 2018.

[7] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5156–5165. PMLR, 13–18 Jul 2020.

[8] N. Kitaev, Ł. Kaiser, and A. Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

[9] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[10] P. Shaw, J. Uszkoreit, and A. Vaswani. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[11] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. In *Arxiv*, 2016.

[12] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.

[13] 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*, 30, 2017.