Attention is All You Need

https://arxiv.org/abs/1706.03762

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin

The paper presents the Transformer, a neural network architecture used to process sequential data (natural language processing, NLP) that uses only the attention mechanism (without convolutional neural nets (ConvNets, for non-sequential) or recurrent neural nets (RNN, for sequential)). This architecture presents a state-of-art for NLP tasks (language translation), overcomes performance bottlenecks of previous architectures and can be generalized to other tasks.

1. Introduction. RNN types of NN have a disadvantage that they are not easily parallelized as they are sequential in essence, hence can't be processed non-sequentially in parallel. Also, they are limited in how far two positions in the input can be, as the whole sequence between the positions should be taken. Attention mechanism used in RNN can be used separately by itself, which is the know-how of the Transformer.

2. Background. This section shows that convnets were limited in that time and memory increased linearly or logarithmically with the increase of distance between two positions in input. They don't know any other architecture that uses only the transformer.

3. Model Architecture. The architecture follows RNN - it has an encoder and a decoder. Encoder produces representations, $ z $ based on input $ x $. Decoder generates output $ y $ based on $ z $, in an auto-regressive manner (consumes its own output but shifted). Encoder and decoder consist of sub-layers.

   Encoder:

   ```
   input -> Multi-head attention -> resid + norm -> FF -> resid + norm -> (decoder)
   ```

   Decoder:

   ```
   Masked multi-head attention -> resid + norm -> Mutli-head att. (input from encoder and from self) -> resid + norm -> FF -> resid + norm -> linear transformations for better output
   ```

   where

   - Multi-head attention is an attention block (self-attention or not) with parallelization. Masked multi-head is used to emulate sequential nature (at i-th position we don't see all input).
   - Resid + norm is residual connection (addition of input to output for a sublayer) before normalization: $ norm (x + layer(x)) $.
   - FF is a feed-forward fully connected layer, i.e. two layers with ReLU.

   Attention mechanism is a way to assign attention weights to an input so that NN learns more on those parts of the input that have higher weights and ignores more of those that have lesser weights. The attention weights are computed from input and query vectors, and then they are normalized (using softmax). The intuition behind this is to make the model 'focus' on different

parts of input, similar to the human eye that focuses only on a fraction of the picture (fovea in retina).

Encoder uses self-attention sublayer, i.e. it uses only its input. A decoder uses output from the encoder in its second multi-head attention layer along with output of its first attention layer which uses shifted output of the decoder.

Parallelization is achieved by using scaled dot-product (matrix multiplications).

4. Why Self-Attention. Three ways to compare to other architectures:
    1. Distance between learnable positions. Transformer is the best as it has constant complexity.
    2. Parallel computation. Transformer can do all computations in parallel while RNN can't process all examples asynchronously. But ConvNets also can do it in parallel.
    3. Total complexity. Transformer requires more time and memory overall to train a model.
5. Training. Training was done in batches on 8 NVidia P100 GPUs in 3.5 days. Adam optimizer was used for learning (computing gradient and the rate). Dropout was used (dropping random weights in neural networks so that an average of different NN results in better performance).
6. Results. The transform presents new state-of-the-art scores for several tests in language translation. Several modifications of their architecture were tested to get the resulting one. Also they evaluated the Transformer on English constituency parsing.
7. Conclusion. They present the Transformer, a new type of NN architecture for a sequence transduction model (sequence to sequence processing). They achieved new state-of-the-art results in various tests and proposed to apply the Transformer for other tasks.

Appendix shows visualizations for the attentions between words.

Pros:

- The Transformer architecture can achieve better results with less total complexity compared to ConvNets and RNN for the tasks that require far separated input signals to be learned by the model (consider translation of a book).
- The Transformer can process data in parallel which allows training large models in much shorter time compared to RNN which processes the data sequentially.

Cons:

- Requires more memory compared to RNN or ConvNets. Transformer total complexity per layer grows quadratically to number of examples.
- Unclear performance for other tasks compared to ConvNets with attention mechanisms.

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

https://arxiv.org/abs/1810.04805 Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova

This paper presents new NN architecture based on attention mechanism: Bidirectional Encoder Representations from Transformers (BERT). It aims to overcome limitations of the previous architectures, namely their unidirectional nature, thus pre-training in both directions (left and right). After fine tuning (transfer learning and training only a subset of layers), it achieved new state-of-the-art results for the eleven NLP tasks such as question answering, language inferring (guessing a paragraph idea), etc.

1. Introduction. Current models (esp. GPT) uses fine-tuning (using pre-trained models fully) or feature-based (uses pre-trained models as features) have the limitation in that they are unidirectional. GPT from OpenAI uses the Transformer architecture (based on an attention mechanism with masked attention that hides next chars for the current one). They argue that this greatly limits models in their power as models need to learn from both directions as it happens in question answering. They present a BERT that uses masked language model (MLM) (hiding random tokens in input so that model should guess their meaning) and predict next sentence objectives at the pre-train stage. Their architecture is still based on a Transformer. They use two types of their model different by size: base one (110M params) and large one (340M params).
2. Related work includes a review of many attempts in NLP tasks since 1992 for unsupervised (doesn't have labels) and supervised (has labels) data, fine-tuned and feature-based.
3. BERT
   - They first pre-train the model, modify it for a specific task (add layers, etc.) and then again they train all models (fine-tuning) thus having a specific model for each task (they did 11 tasks). But those modifications are minor.
   - It is the same Transformer architecture described in Vaswani et al. (All You Need is Attention, 2017).
   - They tokenize input text (token is a number or index of a word in a list of words used for a task).
   - 3.1 Pre-training. They train on two unsupervised (no targets or labels) tasks. First is masked language model (MLM), i.e. hiding random tokens in input so that model should guess their meaning. And the other is NSP, Next Sentence Prediction, that is important for such supervised tasks where it should link sentences.
   - 3.2 Fine-tuning For that they supply task specific inputs and outputs, e.g. for paraphrasing tasks (input is two sentence pairs) it matches the pre-training sentences (NSP objective).
4. Experiments. These are those 11 tasks they trained at. It includes tasks from these benchmarks: GLUE (several tasks on language understanding), SQuAD (question and answer pairs), SWAG (sentence completion)
5. Ablation Studies (removing some part(s) in order to detect interdependencies in a large system).
   - They prove that a bidirectional pre-trained model is important for the performance with removal of NSP (see above) and/or MLM (only left part available then).
   - Also they tried to change their model size to see how it affects performance. Varaying
   - If they replace fine-tuning with a feature-based way (instead of using all weights in a pre-trained model they only use final features) then they can achieve results of other best performing models at the time but those results are weaker than if they use fine-tuning.

6. Concultion. They conclude that for NLP it is best to use pre-trained un-supervised models first with bidirectional architectures. Appendix contains illustrations for the tasks, their setup details and ablation studies.

Pros:

- They try to prove empirically that NLP models should be uni-directional and use fine-tuning of pre-trained unsupervised models.

Cons:

- They didn't try with a larger number of parameters (max 340M). Compared to other NLP models with billions of params.
- Empirical evidence doesn't finally prove their theory and mathematical approach should be more trustful.

Unsolved Problems in ML Safety

https://arxiv.org/abs/2109.13916

Summary

Paper urges to change focus from ML capabilities to ML safety by providing attack directions for four interconnected areas: robustness, monitoring, systemic safety and objective alignment. It enumerates hazards that arise from ML systems and their urgency. Motivated directions of research, engineering, and advocating are given which are prioritized in appendixes. The paper is based on several many related works.

1. Introduction. It argues that we need to make clear what problems to address because the following hazards become more critical as capabilities of ML increases.
   - Hazards:
     - They argue that we should not prolong doing ML Safety work, we should do it now as it requires time, effort and iterations to get good results in safety. Postponing the work increases the likelihood of accidents.
     - ML systems does not generalize from the previous research on safety (e.g. from electroengineering).
     - We can't rely on corporations to make the safety work as they have little incentives. ML Safety is a public good.
   - We should start work as now it is a critical time to catch up until it is too late.
2. Robustness. About how ML systems will withstand unpredicted or carefully crafted events.
   - 2.1 Black Swan and Tail Risk Robustness.
     - Motivation is that black swan events happen as history depicts, i.e. long tail scenarios.
     - Directions:
       - Creating environments to emulate those events.
       - Creating benchmarks with unusual long tail events.
       - Adaptable systems are hard (impossible?) to prepare for black swan events. 2.2 Adversarial Robustness. About how ML systems should defend themselves from attacks.

- Motivation. Attackers use more and more sophisticated tools and methods and defenders need to keep pace with attackers.
- Directions.
  1. Conducting research on more broader robustness definitions. Particularly research should focus on more general attack specifications.
  2. Addressing attacks that target ML systems as black box, i.e. inserting malicious examples.
  3. Work on handling more data from more sensors of ML systems, like those self-driving cars have. Finding inconsistencies in them.
  4. Research on robust representations.
  5. Robust training models, better loss functions.
  6. Work on verifiable certifications for models.
  7. Creating environments with random worse cases.

3. Monitoring. This part addresses how ML systems can be monitored from outside.
   - 3.1. Anomaly Detection. This section is about getting operators to know about potential hazards, anomalies. This is like those systems we have for nuclear plants. Directions of work here are 1) detecting unseen anomalies; 2) monitoring strange changes in data distributions; 3) also addressing the causes of those anomalies.
   - 3.2 Representative Model Outputs.
     - 3.2.1 Calibration. This is about the question to what degree could we trust ML systems. Uncertainty of the models can be made more representative (direction 1), systems can tell themselves about their uncertainties (direction 2) in various ways.
     - 3.2.2 Honest outputs and truthfulness. Current language systems don't provide a way to tell if their outputs are honest. Directions here are work on detecting inconsistencies, deceptions (giving wrong answers when they know a better one). Work on creating environments to train such models.
   - 3.3 Hidden Model Functionality. This is addressing backdoors and unforeseen dangerous functionality hidden inside a model.
     - 3.3.1. This is about if models have 'backdoors', e.g. a hacker can train a model that controls building access by facial recognition to give her access if she is wearing a specific thing. Directions: detections of those backdoors, creating competitions for research (like capture the flag).
     - 3.3.2 Emergent Hazardous Capabilities. This is if ML models are trained to do something that teachers didn't foresee, like GPT-3 learned to do arithmetics, but such capabilities to address would be dangerous. Directions: techniques and tools to detect those capabilities with blackbox (environments) and whitebox approaches, teach ML models to forget those capabilities.
4. Alignment. There are mainly two types of issues here, i.e. societal and technical. Biggest worries here are that now development of ML systems are in the hands of people that might not prioritize safety hence those systems become mis-aligned with high probability.
   - 4.1. Difficult Objectives. Motivation is we need a philosophical research of our global human values in our brave new world. Directions are giving better approximations for our values, aligning current systems with those approximations, teaching models the law, etc.
   - 4.2 Optimizing those objectives. How to put those values into action? This is difficult for humans and here we want to make such a system. This is to optimize wellbeing on a long timescale, for example. Directions for such a hard problem: constructing systems that train those models to behave morally in artificial environments.
   - 4.3 Brittle Objectives. As we approximate objectives they can be easily gamed or dropped hence this is a work on a technical side to make sure ML systems nevertheless pursue the objectives.
     - Work on the system trying to game those objections. Directions are interleaved with the work on robustness and monitoring (see above) but here we should focus on optimizers being robust.
     - Work on value clarification. Here is interdisciplinary work to make those values clear and consistent with future events. Directions of work are for philosophers, law makers, and others.
   - 4.4 Unintended Consequences. This is like not to repeat our errors with developing technologies that make our life easier in short-term but worse in long-term, e.g. climate change. Similarly using ML systems might have such undesired consequences. Directions are making minimally invasive agents, reversible actions, etc.

5. Systemic Safety. Now in this section they address cybersecurity and help with good decisions as ML systems are not isolated from other worlds.
   - 5.1 ML for Cybersecurity. As ML systems will take more resources their security should be addressed as we handle current systems security. Here ML systems can help with that. On the other side attackers will use ML to become more sophisticated. This field is now neglected. Directions are creating better defensive techniques for defense from automated cyberattacks, modeling behavior and other.
   - 5.2 Improving decision making. This is to help those in charge of ML systems to make better decisions by improving forecasting, and having a better world view. Directions are two fold: forecasting and getting right questions for forecasting.
6. Related Research Agendas. This references work done so far on ML safety.
7. Conclusion. They presented four problems for research and those are interconnected. It is crucial now to shift focus from capabilities to safety as it might be too late as systems become more and more sophisticated. In appendices they present specific vectors of attack to those problems, how they are interconnected and their urgency.

Judge

Pros:

- This work stresses the urgency of ML safety work which is neglected somewhat now.
- The alignment part gives the impression that it is very high stakes and we will have mis-aligned systems. Is it true?

Cons:

- It is unclear to me how we can prepare for black swans. The paper didn't make it easier for me to understand it.
- Addressing ML systems with human concepts such as ML systems can believe, can "produce outputs honestly" (3.2.2), etc. This might be dangerous to assume such concepts in ML systems as it is an open question if they can believe, be honest, etc. Also how to teach a ML model the law?