Research Paper Notes and Doubts

1. Large language models in machine translation

Abstract: This paper reports on the benefits of large- scale statistical language modeling in machine translation. A distributed infrastructure is proposed which we use to train on up to 2 trillion tokens, resulting in language models having up to 300 billion n-grams. It is capable of providing smoothed probabilities for fast, single-pass decoding. We introduce a new smoothing method, dubbed Stupid Backoff, that is inexpensive to train on large data sets and approaches the quality of Kneser-Ney Smoothing as the amount of training data increases.

Notes

Encoder decoder structure, trying to reduce computation time by Map Reduce techniques. The encoder creates n-grams (vocabulary) and the decoder uses a distributed language model to try and get the ideal sequence. Tries to guess a word based on the last n words. Problems in calculating probabilities due to sparse data. Can make n bigger in n-grams but the problem persists. Introduces a new smoothing algorithm called stupid backoff that uses relative frequencies. Average sentence length is 22. N-grams max n is 5 for experiments.

Doubts

Meaning of the sentence in introduction: “In particular, it proposes a distributed language model training and deployment infrastructure, which allows direct and efficient integration into the hypothesis-search algorithm rather than a follow-on re-scoring phase.”

In the BLUE score for n-grams, can we add some aggregation method to also match n-1, n-2, …n-m grams as well? Will that be a better metric to evaluate NLP results?

Not exactly related to this paper, but can we train a deep learning model on multiple systems using Map Reduce? Maybe we can try simulating a dropout mechanism for them and have aggregation points?

2. Attention is all you need

Abstract: The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English- to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Notes

Limitation of RNNs is not being able to parallelize due to being sequential, high complexity of each layer and can’t handle long sequence lengths. Attention mechanisms help form dependencies irrespective of their distance in input/output sequences. Self-attention layers are faster than recurrent layers when the sequence length n is smaller than the representation dimensionality d,

and have a constant number of sequential operations.
Transformers are unique since they rely only on self-attention, without any use of RNNs or CNNs. Scaled dot-product attention is used because of better performance in matrix multiplication. To prevent very small gradients, scaling is used (factor of square root (d_k)). Transformers use stacked self-attention sub-layers within the encoder and decoder, performing different roles for encoder-decoder, encoder and decoder layers.

Doubts
"Our model achieves 28.4 BLEU on the WMT 2014 English- to-German translation task". BLEU score is usually measured on a scale of 0-1, is 28.4 in terms of percentage, indicating score of 0.284?

Meaning of position-wise in: "position-wise fully connected feed-forward network".
What does it mean and how are we doing it: "We also modify the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions."

Not able to understand why we want to prevent this: "We need to prevent leftward information flow in the decoder to preserve the auto-regressive property." BERT tried bidirectional flow and it showed great results.

3. Improving language understanding by generative pre-training

Abstract: Natural language understanding comprises a wide range of diverse tasks such as textual entailment, question answering, semantic similarity assessment, and document classification. Although large unlabeled text corpora are abundant, labeled data for learning these specific tasks is scarce, making it challenging for discriminatively trained models to perform adequately. We demonstrate that large gains on these tasks can be realized by generative pre-training of a language model on a diverse corpus of unlabeled text, followed by discriminative fine-tuning on each specific task. In contrast to previous approaches, we make use of task-aware input transformations during fine-tuning to achieve effective transfer while requiring minimal changes to the model architecture. We demonstrate the effectiveness of our approach on a wide range of benchmarks for natural language understanding. Our general task-agnostic model outperforms discriminatively trained models that use architectures specifically crafted for each task, significantly improving upon the state of the art in 9 out of the 12 tasks studied. For instance, we achieve absolute improvements of 8.9% on commonsense reasoning (Stories Cloze Test), 5.7% on question answering (RACE), and 1.5% on textual entailment (MultiNLI).

Notes
Rather than training a model individually for each NLP task, pre-train a model on a large corpus of data and fine-tune for each use-case (transfer learning). Semi-supervised learning setting is used, wherein pre-training is done on an unlabelled corpus and fine-tuning is done for labeled training sets depending on the use-case. The pre-training part uses a decoder-only transformer with multi-headed self-attention creating a generative model that produces an output distribution over target tokens. The aim is to have the pre-trained model generate outputs, similar to the text in the initial corpus, depending on the task we fine-tune it for during the training stage.

Doubts
Why is the pre-training model called a decoder? What does the input and output look like in this scenario?

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

Abstract: We introduce a new language representation model called BERT, which stands for Bidirectional Encoder Representations from Transformers. Unlike recent language representation models (Peters et al., 2018a; Rad- ford et al., 2018), BERT is designed to pre- train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine- tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task- specific architecture modifications. BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answer- ing Test F1 to 93.2 (1.5 point absolute im- provement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement).

Notes

A pre-trained NLP model that can be fine-tuned for any task and by adding an additional layer at the end, we can perform different kinds of tasks. GPT-1 was unidirectional but BERT is bidirectional. Pre-training is for two tasks: masked language model (predicting the missing word in a sentence) and next sentence prediction (guessing whether two sentences are consecutive or not). Bi-directional attention mechanism helps it to learn contextual meaning and helps in tasks like question-answering.

Doubts

Why is the bidirectional Transformer often referred to as a “Transformer encoder” while the left-context-only version is referred to as a “Transformer decoder”?

“However, the feature-based approach, where fixed features are extracted from the pre- trained model, has certain advantages.” Why is the performance affected? Won't the new model need any training?

5. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

Abstract: Transformers have a potential of learning longer-term dependency, but are limited by a fixed-length context in the setting of language modeling. We propose a novel neural architecture Transformer-XL that enables learning dependency beyond a fixed length without disrupting temporal coherence. It consists of a segment-level recurrence mechanism and a novel positional encoding scheme. Our method not only enables capturing longer-term dependency, but also resolves the context fragmentation problem. As a result, Transformer- XL learns dependency that is 80% longer than RNNs and 450% longer than vanilla Transformers, achieves better performance on both short and long sequences, and is up to 1,800+ times faster than vanilla Transformers during evaluation. Notably, we improve the state-of- the-art results of bpc/perplexity to 0.99 on en- wiki8, 1.08 on text8, 18.3 on WikiText-103, 21.8 on One Billion Word, and 54.5 on Penn Treebank (without fine tuning). When trained only on WikiText-103, Transformer-XL manages to generate reasonably coherent, novel text articles with thousands of tokens. Our code, pretrained models, and hyperparameters are available in both Tensorflow and PyTorch.

Notes

Authors introduce recurrence into the self-attention models and use relative positional encodings rather than absolute ones. Recurrence is introduced by sending information through these

connections. One hidden state sends the information to the next segment's hidden state. Every two consecutive segments in the corpus share information like this, essentially creating recurrence in the hidden states. Relative positional encodings help avoid temporal confusions. All this helps the transformer to remember long range dependencies and maintain contextual information.

Doubts
Why are self-attention mechanisms less affected by the vanishing gradient problem compared to RNNs?

6. Language Models are Unsupervised Multitask Learners

Abstract: Natural language processing tasks, such as question answering, machine translation, reading comprehension, and summarization, are typically approached with supervised learning on task- specific datasets. We demonstrate that language models begin to learn these tasks without any explicit supervision when trained on a new dataset of millions of web pages called WebText. When conditioned on a document plus questions, the answers generated by the language model reach 55 F1 on the CoQA dataset - matching or exceeding the performance of 3 out of 4 baseline systems without using the 127,000+ training examples. The capacity of the language model is essential to the success of zero-shot task transfer and increasing it improves performance in a log-linear fashion across tasks. Our largest model, GPT-2, is a 1.5B parameter Transformer that achieves state of the art results on 7 out of 8 tested language modeling datasets in a zero-shot setting but still underfits WebText. Samples from the model reflect these improvements and contain coherent paragraphs of text. These findings suggest a promising path towards building language processing systems which learn to perform tasks from their naturally occurring demonstrations.

Notes
Very similar to GPT-1. Key differences include addition of layer normalization in sub-layers, bigger training set and a bigger model to train. The authors focus on providing SOTA results in zero-shot task-transfer, rather than fine-tuning the model for each task (like GPT-1). The model is supposed to guess the input and the task it has to perform based on the query.

Doubts
What do the authors mean by this: "A modified initialization which accounts for the accumulation on the residual path with model depth is used."

7. Language Models are Few-Shot Learners

Abstract: We demonstrate that scaling up language models greatly improves task-agnostic, few-shot performance, sometimes even becoming competitive with prior state-of- the-art fine-tuning approaches. Specifically, we train GPT-3, an autoregressive language model with 175 billion parameters, 10x more than any previous non- sparse language model, and test its performance in the few-shot setting. For all tasks, GPT-3 is applied without any gradient updates or fine-tuning, with tasks and few-shot demonstrations specified purely via text interaction with the model. GPT-3 achieves strong performance on many NLP datasets, including translation, question-answering, and close tasks. We also identify some datasets where GPT- 3's few-shot learning still struggles, as well as some datasets where GPT-3 faces methodological issues related to training on large web corpora.

Notes
GPT-2 mistakenly calls few-shot learning as zero-shot since they might have some examples of the task they intend to perform in the training set. They train the model on a huge corpus initially

in an unsupervised manner. Then for making sure that the model performs on different kinds of tasks, considering few-shot learning, they give k examples (where k is based on the context window and example is the context and the output) for a task. At k+1th example, we just give the context and expect it to produce the output.

GPT-1	**GPT-2**	**GPT-3**
117M parameters	117M, 345M, 762M, 1.5B parameters	125M - 13B, 175B parameters
Unsupervised pre-training and supervised fine-tuning	Unsupervised pre-training and zero-shot task transfer	Unsupervised pre-training and in-context learning for results in few-shot setting
Learning objective: P(output|input)	Learning objective: P(output|input, task)	Similar to GPT-2, but now focused on few-shot, one-shot and zero-shot learning

Doubts

"We partition each model across GPUs along both the depth and width dimension in order to minimize data-transfer between nodes." How does this minimize data transfer between nodes?

What exactly is the context window (k) here? Is it like the maximum length their transformer can take at a time without losing context information?

Why are the authors using "alternating dense and locally banded sparse attention patterns in the layers of the transformer"?

When OpenAI deployed ChatGPT, they're doing inference on a large variety of tasks. Did they give the examples for few-shot learning on all the tasks before deploying the model? When we enter the query in the chat box, is the model doing zero-shot learning at that time?

8. <u>On the Dangers of Stochastic Parrots: Can Language Models be Too Big</u>

Abstract: The past 3 years of work in NLP have been characterized by the development and deployment of ever larger language models, especially for English. BERT, its variants, GPT-2/3, and others, most recently Switch-C, have pushed the boundaries of the possible both through architectural innovations and through sheer size. Using these pretrained models and the methodology of fine-tuning them for specific tasks, researchers have extended the state of the art on a wide array of tasks as measured by leaderboards on specific benchmarks for English. In this paper, we take a step back and ask: How big is too big? What are the possible risks associated with this technology and what paths are available for mitigating those risks? We provide recommendations including weighing the environmen- tal and financial costs first, investing resources into curating and carefully documenting datasets rather than ingesting everything on the web, carrying out pre-development exercises evaluating how the planned approach fits into research and development goals and supports stakeholder values, and encouraging research directions beyond ever larger language models.

Talked about it in Assignment 2.C Deep Learning.

9. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter
   Abstract: As Transfer Learning from large-scale pre-trained models becomes more prevalent in Natural Language Processing (NLP), operating these large models in on-the-edge and/or under constrained computational training or inference budgets remains challenging. In this work, we propose a method to pre-train a smaller general purpose language representation model, called DistilBERT, which can then be finetuned with good performances on a wide range of tasks like its larger counterparts. While most prior work investigated the use of distillation for building task-specific models, we leverage knowledge distillation during the pre-training phase and show that it is possible to reduce the size of a BERT model by 40%, while retaining 97% of its language understanding capabilities and being 60% faster. To leverage the inductive biases learned by larger models during pre-training, we introduce a triple loss combining language modeling, distillation and cosine-distance losses. Our smaller, faster and lighter model is cheaper to pre-train and we demonstrate its capabilities for on-device computations in a proof-of-concept experiment and a comparative on-device study.

10. Distilling the Knowledge in a Neural Network
    Abstract: A very simple way to improve the performance of almost any machine learning algorithm is to train many different models on the same data and then to average their predictions [3]. Unfortunately, making predictions using a whole ensemble of models is cumbersome and may be too computationally expensive to allow deployment to a large number of users, especially if the individual models are large neural nets. Caruana and his collaborators [1] have shown that it is possible to compress the knowledge in an ensemble into a single model which is much easier to deploy and we develop this approach further using a different compression technique. We achieve some surprising results on MNIST and we show that we can significantly improve the acoustic model of a heavily used commercial system by distilling the knowledge in an ensemble of models into a single model. We also introduce a new type of ensemble composed of one or more full models and many specialist models which learn to distinguish fine-grained classes that the full models confuse. Unlike a mixture of experts, these specialist models can be trained rapidly and in parallel.

    Notes
    Distilling is the concept analogous to the physics concept where you raise the temperature and entropy increases. With increased entropy, there is an equal probability of any possible event happening. In classification models, with increased value of T, there is an equal probability of all the classes. We want the model to be able to relate to two classes, and say how different they are. We don't want absolute probabilities, we want to know how close two classes are.

    Doubts
    Meaning of 'raising the temperature' in: "Our more general solution, called "distillation", is to raise the temperature of the final softmax until the cumbersome model produces a suitably soft set of targets. We then use the same high temperature when training the small model to match these soft targets. We show later that matching the logits of the cumbersome model is actually a special case of distillation."

    What does it mean to have 'softer' probability and how does it affect the performance: "T is a temperature that is normally set to 1. Using a higher value for T produces a softer probability distribution over classes.".

    A question not related to the above papers specifically:
    "Though it is extremely popular in recent years, we find the attention mechanism does not always contribute to the performance of a model since the video captioning system may overfit on the

training set more easily.”

Is it true? This is related to the context of video captioning. Considering the case where we do not want very long range dependencies to be learnt, are we better off by not using the transformers? Do RNNs (gated) give better performance in some settings in practical experience?