Chapter 26
Ascent of Pre-trained State-of-the-Art Language Models

Keval Nagda, Anirudh Mukherjee, Milind Shah, Pratik Mulchandani and Lakshmi Kurup

1 Introduction

Applications involving natural language processing (NLP) have become significantly easier, faster and more economical to build these days largely due to the immense computational power in our hands which can be used to develop pre-trained models that can be used to come up with a number of solutions to a wide array of tasks via transfer learning and fine-tuning. Language modeling is a fundamental problem of NLP that needs to be addressed appropriately in order to come up with proper solutions for a number of NLP tasks. A language model works toward estimating the probability of various linguistic units such as characters, words, sentences and paragraphs. In order for such words to be processed by these models, they need some form of numeric representation. Using traditional embeddings wherein such words are represented as vectors has always been a popular approach but they have a major limitation. They fail to consider the context behind a word and assume its meaning to be the same across all sentences. For example, the word "rose" in "Debbie rose to give her speech." and "Debbie gave a rose to her mother." have different meanings,

K. Nagda · A. Mukherjee · M. Shah (✉) · P. Mulchandani · L. Kurup
Department of Computer Engineering, Dwarkadas J Sanghvi College of Engineering, Mumbai, India
e-mail: mlndshh63@gmail.com

K. Nagda
e-mail: knagda008@gmail.com

A. Mukherjee
e-mail: anirudhmk130@gmail.com

P. Mulchandani
e-mail: pratikm1910@gmail.com

L. Kurup
e-mail: lakshmidkurup@gmail.com

H. Vasudevan et al. (eds.), *Advanced Computing Technologies and Applications*, Algorithms for Intelligent Systems,
https://doi.org/10.1007/978-981-15-3242-9_26

but are represented via the same vector in such traditional embedding models which are thus trained on shallow representations. This is where we bring in pre-trained state-of-the-art models which have been highly successful in tackling this issue.

Language modeling has shown to capture many aspects of language relevant for downstream tasks such as hierarchical relations, question answering, paraphrasing and machine translation. Such language models have seen countless developments in recent days, the freshest of them being the use of pre-trained models to perform NLP tasks. A few of these models are task-specific, while the others are more generalized. Using such pre-trained models is highly beneficial due to the fact that it can be fine-tuned to solve any specific task doing away the need to train the model from scratch. It thereby reduces the amount of labeled data needed to train it in order to solve that task and also reducing the computational load which leads to faster results and an efficient model. Researchers have used a number of different methodologies in order to pre-train such models, almost all of them being trained on huge datasets. This approach of using pre-trained models leads to the model being able to learn both the lower-level and higher-level features of the language, which leads to a much better performance in almost all standard NLP tasks. In the following sections, we review some of the most popular cutting-edge state-of-the-art models, namely ULMFiT, ELMo, Transformer-XL, BERT and OpenAI's GPT-2.

2 ULMFiT

Computer vision (CV) has achieved a drastic enhancement using transfer learning methods, and this approach can be used to tackle a major set of NLP tasks too. Several state-of-the-art models used to solve CV problems are obtained by fine-tuning of models pre-trained on heavy datasets like ImageNet and MS-COCO.

Early NLP research majorly consisted of the transductive transfer learning. Universal Language Model Fine-Tuning (ULMFiT) [1] introduces an inductive transfer learning approach with different fine-tuning methods which tackles the problem of overfitting of language models on small datasets and can be used for any NLP task.

Pre-trained word embeddings [2] are also used in most of the state-of-the-art models, but this technique adds a constraint where models using pre-trained embeddings consider them as fixed parameters. The ULMFiT proposes various fine-tuning approaches such as discriminative fine-tuning, slanted triangular learning rates and gradual unfreezing techniques to maintain long-term dependencies in the model.

2.1 Working

ULMFiT consists of two major tasks:

General-domain LM pre-training: Pre-training the language model (LM) on a large general-domain corpus helps with generalization even with small datasets.

Target task LM fine-tuning: Fine-tuning the pre-trained LM model on the target task using the following proposed novel techniques.

- *Discriminative fine-tuning*: It is observed that different layers of a model are activated according to different types of information [3], and thus, discriminative fine-tuning should be applied in order to tune each layer with different learning rates rather than using the same learning rate for all layers of the model.
- *Slanted triangular learning rates*: To obtain task-specific feature learning, slanted triangular learning rates (STLR) can be used by the model to initially converge to an appropriate region in the parameter space by a linear increase in learning rates and then optimize its learning rate by refining the parameters.
- *Gradual unfreezing*: In this approach, instead of fine-tuning all the layers of a model in a single shot, a layer-wise unfreezing and training occur. Starting from unfreezing the last layer as this layer contains the least general knowledge [3], fine-tuning of all unfrozen layers for one epoch is carried out. Then, the next lower frozen layer is unfrozen and the process is repeated until all the layers of the model are fine-tuned as required.

2.2 Dataset

Six widely studied datasets, with a varying number of documents and varying document lengths, are used. The datasets are TREC-6 (6 classes, 5.5 k examples), IMDb (2 classes, 25 k examples), Yelp-bi (2 classes, 560 k examples), Yelp-full (5 classes, 650 k examples), AG (4 classes, 120 k examples) and DBpedia (14 classes, 560 k examples). The language model is pre-trained on Wikitext-103 [1].

2.3 Architecture

ULMFiT consists of the state-of-the-art language model AWD-LSTM, as a base LSTM model with tuned dropout hyperparameters. It consists of an embedding size of 400 with three layers, 1150 hidden activations per layer and a backpropagation through time (BPTT) batch size of 70. A dropout of 0.4 is applied to layers, 0.3 to RNN layers, 0.4 to input embedding layers, 0.05 to embedding layers and 0.5 of weight dropout to RNN hidden-to-hidden matrix. Adam optimizer is used with $\beta_1 = 0.7$ instead of $\beta_1 = 0.9$ and $\beta_2 = 0.99$. A batch size of 64 with a base learning rate of 0.004 and 0.01 for fine-tuning the LM is chosen. The LM is fine-tuned for 15 epochs on small datasets while for 50 epochs on large datasets.

3 ELMo

Pre-trained word representations [2, 4] are very crucial elements in many state-of-the-art language models. However, these elements lack the representation of complex word characteristics (syntax and semantics) and are often used across varied linguistic contexts. ELMo proposes a deep contextualized word representation approach that tackles various challenges faced by traditional word embeddings. In ELMo, vectors are extracted from a bidirectional LSTM model coupled with language model (LM) that is pre-trained on a huge textual corpus. These context-dependent representations derived from the internal layers of the deep bidirectional language model (biLM) outperform models which use a neural machine translation encoder model to extract contextualized representations.

3.1 Working

ELMo can be used for any target-specific task by just combination of intermediate layers. Initially, the supervised target model forms context-independent token representation using pre-trained word embeddings and later context-sensitive representations are formed. A set of $2L + 1$ representations are computed by a L-layer biLM for each token. Usage of ELMo for any downstream supervised model can be done by concatenating all the layers into a single vector after freezing the weights of the biLM. Activations of each layer have variance in distribution, and thus, layer normalization aids if applied before weighting. A moderate amount of dropout addition is observed to be beneficial in few scenarios.

3.2 Dataset

The bidirectional language model (biLM) is pre-trained on 1B Word Benchmark and can be fine-tuned for most of the target-specific downstream tasks [4].

3.3 Architecture

The pre-trained bidirectional language model (biLM) is similar to the architecture in [5] with an additional connection between LSTM layers of the model. Embedding and hidden dimensions are halved as compared to the CNN-BIG-LSTM model [5].

The model consists of 2 biLSTM layers with 4096 units and 512-dimensional projections where the first and second layers are connected. The model is trained

for 10 epochs. The context-independent representation follows 2048 character n-gram convolutional filters, two highway layers and a linear 512 projections. The traditional word embedding methods provide only one layer of representation for tokens, whereas the biLM provides three layers of representations for each input token.

4 Bert

BERT, or Bidirectional Encoder Representations from Transformers [6], is a language representation model developed by Google's AI team which was able to achieve state-of-the-art performance in a number of natural processing tasks. Unlike traditional models [7] which take the previous n tokens for its prediction, BERT takes into consideration both the previous and the next token. BERT thus takes into consideration both the left and right contexts of a token in all of its layers, unlike the standard models which only take the left context into consideration. BERT is also trained on the next sentence prediction in order to handle tasks that require a certain degree of knowledge about the relationship between sentences. BERT makes use of a fine-tuning approach which introduces a minimum number of task-specific parameters and is trained on downstream tasks by fine-tuning all of the pre-trained parameters. Thus, with a little bit of fine-tuning, the BERT model can be used to create state-of-the-art models for a number of tasks.

4.1 Working

BERT makes use of a "masked language model" (MLM) pre-training objective in order to do away with the unidirectional constraint of traditional language models. MLM randomly masks or hides some of the tokens in the input and predicts the masked word based only on its context. The BERT framework includes two steps, namely pre-training and fine-tuning.

In the pre-training stage, BERT is trained on unlabeled data over a large number of different tasks. Pre-training of BERT occurs in two steps—masked LM and next sentence prediction (NSP).

Masked LM. In this step, a certain percentage of input tokens are masked at random and then the original vocabulary of these masked tokens is then predicted.

NSP. This step ensures that the model understands the relationships between sentences. To accomplish this, two sentences A and B are chosen for each pre-training example, 50% of the time B is the actual sentence after A and 50% of the time it is a random sentence from the corpus.

For the fine-tuning step, the task-specific inputs and outputs are plugged into the model and all parameters are fine-tuned end to end to obtain the fine-tuned model.

4.2 Dataset

BERT uses the BookCorpus (800 M words) and English Wikipedia (2,500 M words) [6]. Only the text passages are extracted for the Wikipedia corpus, thereby ignoring lists, tables and headers.

4.3 Architecture

BERT makes use of a unified architecture along all tasks which basically means that its architecture remains the same irrespective of the task. BERT's model architecture is a multi-layer bidirectional Transformer encoder based on the original implementation described in [8]. The Transformer architecture is a model that does not use recurrent connections at all and uses attention over the sequence instead. Unlike RNNs, the Transformer architecture cannot take the order of the inputs into account. Hence, to overcome this problem, BERT makes use of positional embeddings to express the position of a word in a sentence.

BERT was developed on two model sizes, namely BERT base and BERT large. BERT base has 12 layers or Transformer blocks, 768 hidden layers, 12 self-attention heads and 110 M parameters in total, whereas BERT large has 24 layers or Transformer blocks, 1024 hidden layers, 16 self-attention heads and 240 M parameters. BERT base was chosen to have the same model size as OpenAI's GPT for comparison purposes.

BERT makes use of bidirectional self-attention, whereas GPT uses constrained self-attention and thus can only take into consideration the left context of a certain token, whereas BERT because of its bidirectionality can make use of both the left and right contexts. BERT and GPT are similar in terms of model architecture apart from the attention masking.

5 OpenAI's GPT

GPT, by OpenAI, was released on June 2018. GPT is a task-agnostic system with very good results on various language tasks [7]. GPT is an amalgamation of Transformers and unsupervised pre-training. GPT shows that large gains on various tasks like textual entailment, question answering, document classification and semantic similarity assessment can be seen by generative pre-training of a LM on a varied corpus of unlabeled text, continued then by discriminative fine-tuning on each specific task. A Transformer [8] model is first trained on large amounts of data in an unsupervised manner, with language modeling as a training signal. This model is then fine-tuned on smaller supervised datasets, so that it can solve specific tasks.

GPT2, the successor to generative pre-training (GPT) was announced by OpenAI in February of 2019 and as they have put it, is a direct scale-up of GPT, with more than 10× the parameters and trained on 10× the amount of data. As noted by Radford et al. [9], ML systems do great at tasks that they are trained for by a large number and variety of datasets, high-capacity models and supervised learning.

5.1 Working

The goal of GPT is to learn a common representation that carries forward with little adaptation to a varied variety of tasks. The setup does not need the target tasks to be in the same domain as the unlabeled text. A language modeling objecting is first used on the unlabeled data to learn the initial parameters of the neural network model. Later, these parameters are adjusted to target tasks using respective supervised objectives. Using the Transformer model gives the choice of more structured memory to handle long-term dependencies in text compared to other options like recurrent networks. During the transfer, task-specific input adaptations are used which process structured text as a single continuous sequence of tokens. These adaptations allow for effective fine-tuning with very little to no changes to the architecture of the pre-trained model.

GPT2 works on similar principles as GPT, with modifications in the input representation. GPT2 uses byte pair encoding (BPE), which interpolates between word-level inputs for frequent word sequences and character-level inputs for infrequent word sequences. Since BPE may include many variations of common words, they prevent BPE from merging across character categories for any byte sequence. A special case is added for spaces which significantly improves the compression efficiency while adding only minimal fragmentation of words across multiple vocab tokens. This input representation combines the empirical benefits of word-level LMs with the generality of byte-level approaches.

5.2 Dataset

GPT used various datasets for the different tasks. For natural language inference, SNLI, MultiNLI, Question NLI, SciTail, RTE datasets were used. For question answering, RACE, Story Cloze were used, for sentence similarity, MSR Paraphrase Corpus, Quora Question Pairs, STS Benchmark were used and for classification, Stanford Sentiment Treebank-2, CoLA were used [7].

The approach used for GPT2 involved building as big and varied of a dataset as possible so that texts from varied domains and contexts could be gathered, by scraping all outgoing links from Reddit with Karma score of more than three was used. The final dataset known as WebText after de-duplication and heuristic cleaning contains over 8 million documents and over 40 gigabytes in text. All Wikipedia links

were removed since it is a common source for other datasets which would have led to complications including overlapping training data.

5.3 Architecture

A Transformer [8]-based architecture is used for the LMs. The model adopts the details of the OpenAI GPT model [7] with a few changes. Layer normalization is moved to the input of each sub-block, similar to a pre-activation residual network [10], and additional layer normalization was added after the final self-attention block. The initialization is changed, where a residual path with depth model is used which accounts for the accumulation. The weights of residual layers are scaled at initialization by a factor of $1/\sqrt{N}$ where N is the number of residual layers. The vocabulary is expanded to 50,257. The context size is also increased from 512 to 1024 tokens, and a larger batch size of 512 is used.

6 XLNet

Two of the most commonly used language models are BERT [6] and Transformer-XL [11], which use techniques such as autoencoding (AE) and autoregressive (AR) language modeling, respectively, during the pre-training process.

BERT, the earlier SoTA model, allows for the use of bidirectional context to construct sequences from a given input, which helps reduce the risk of context fragmentation. However, it requires certain tokens to be converted into an artificial symbol [MASK] in order to predict these tokens during training. Due to this mechanism, all of the tokens in the real data cannot be used, as some of them are masked in the input. This may cause discrepancies, and BERT assumes that the predicted tokens are completely independent of each other and ignores their joint probability.

To overcome this shortcoming of BERT, XLNet was autoencoding (AE) proposed, which uses a generalized AR method, similar to that used in Transformer-XL.

6.1 Working

XLNet maximizes the log-likelihood of a sequence for all the available permutations of the factorization order. It captures bidirectional context and learns from it. To ensure that the input sequence order is kept intact, it uses positional encodings.

Since it is an AR-based model, it does not use the data corruption mechanism that BERT uses, thus neglecting the independence assumption. This helps increase its accuracy, as 100% of the input data is being used as context.

The mechanisms borrowed from Transformer-XL are:

- *Relative positional encoding*, which helps improve the performance during pre-training. Since this encoding is sequential, it allows for XLNet's permutation operation.
- *Segment recurrence mechanism*, which helps cache the previous segment and uses it as context for the next segment.

6.2 Dataset

Similar to BERT, XLNet was pre-trained using English Wikipedia dataset (13 GB of plain text), as well as CommonCrawl, Giga5 and ClueWeb 2012-B datasets [11]. The large variant of the model has a sequence and memory length of 512 and 384, respectively. The model was trained on 512 TPU v2 chips for 500,000 steps for about two and a half days.

6.3 Architecture

A two-stream self-attention mechanism is implemented in order to overcome the requirements presented by the permutation system; i.e., only the position, not the content, of a token must be used during predictions, and previous tokens should also be encoded to preserve contextual information.

The last tokens are predicted in a factorization order, in order to make convergence faster. While trying to maximize the log-likelihood of the target sequence, only a select few (K) tokens are selected. The unselected tokens' query representations are not calculated, which helps speed up computation (Table 1).

7 Results

7.1 GLUE

The General Language Understanding Evaluation (GLUE) [12] benchmark is a collection of resources which includes a certain set of tasks based on which a certain natural language model can be evaluated. GLUE consists of a set of nine different sentences or sentence-pair based tasks which get evaluated over a diagnostic dataset. Models are thus evaluated as per their average accuracies across all tasks. Currently, the XLNet model stands at the top of the GLUE leaderboard with an average GLUE score of 88.4 beating previous state-of-the-art models in a number of tasks.

Table 1 Observed experimental scores

System	MNLI (m/mm)	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STSB	SNLI	SQuAD (EM/F1)	GLUE
BiLSTM + ELMo + Attn	74.1/74.5	79.8	63.1	58.9	90.4	84.4	33.6	74.2	88.7	78.58/85.833	70.0
BERT	86.7/85.9	92.7	72.1	70.1	94.9	89.3	60.5	86.5	–	87.4/93.2	80.5
GPT	82.1/81.4	88.1	70.3	56.3	91.3	82.3	45.5	82.0	89.9	–	72.8
XLNet (Multi-task ensembles on test)	90.2/889.7	98.6	74.2	86.3	96.8	93.0	67.8	91.8	–	86.35/89.13	88.4

7.2 SNLI

The Stanford Natural Language Inference (SNLI) [13] corpus is a large collection of natural language inference problems. Each problem exists as a pair of two sentences, the premise and the hypothesis, along with a label. The model thus has to predict the label correctly based on the premise and the hypothesis. Based on the models reviewed in this paper, OpenAI's GPT has attained the highest accuracy on the SNLI corpus with a score of 89.9.

7.3 SQuAD

The Stanford Question Answering Dataset (SQuAD) [14] is a set of crowdsourced question–answer pairs obtained from Wikipedia articles. The model is provided with a question and a passage from Wikipedia, and the goal of the model is to predict the answer text given in the passage. The exact match (EM) score and the F1 score are used for evaluation. Among the models reviewed in this paper, BERT stands at the top with the highest EM and F1 score at 87.4 and 93.2, respectively.

8 Conclusion

Language models can be used to solve a number of trivial and non-trivial NLP tasks with state-of-the-art accuracies which have started to exceed the accuracy with which humans can solve the same set of tasks. These tasks involve text generation, machine translation and question answering to name a few. We have reviewed a number of ensemble multi-task learning models which have given state-of-the-art performances in a number of benchmark tests and have set the bar higher with each new discovery. All the performance measures of the models reviewed in this paper belong to a common subset of tests performed on the same by the original authors of the respective models and duly cited in their respective papers. Each model is evaluated based on the scores obtained in this subset of common tests. The most recent model, the XLNet, has surpassed the previous state-of-the-art scores on 18 tasks. From bidirectionality to self-attention, we have seen language models evolve in a number of innovative ways. With every new model besting the previous state-of-the-art scores, this trend of advancements shall not stop anytime soon.

References

1. Howard J, Ruder S (2018) Universal language model fine-tuning for text classification
2. Mikolov T, Sutskever I, Chen K, Corrado G, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems
3. Yosinski J, Clune J, Bengio Y, Lipson H (2014) How transferable are features in deep neural networks? In: Advances in neural information processing systems, pp 3320–3328
4. Pennington J, Socher R, Manning CD (2014) Glove: global vectors for word representation. In: EMNLP
5. Jozefowicz R, Vinyals O, Schuster M, Shazeer N, Wu Y (2016) Exploring the limits of language modeling. CoRR abs/1602.02410
6. Devlin J, Chang M-W, Lee K, Toutanova K (2018) BERT: pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805
7. Radford Alec, Narasimhan Karthik, Salimans Tim, Sutskever Ilya (2018) Improving language understanding with unsupervised learning. Technical report, OpenAI
8. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I: Attention is all you need. In: Advances in neural information processing systems, pp 6000–6010
9. Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I (2019) Language models are unsupervised multitask learners
10. He K, Zhang X, Ren S, Sun J (2016) Identity mappings in deep residual networks. In: European conference on computer vision. Springer, pp 630–645
11. Dai Z, Yang Z, Yang Y, Cohen WW, Carbonell J, Le QV, Salakhutdinov R (2019) Transformer-xl: attentive language models beyond a fixed-length context. arXiv preprint arXiv:1901.02860
12. Wang A, Singh A, Michael J, Hill F, Levy O, Bowman S (2018) Glue: a multi-task benchmark and analysis platform for natural language understand-ing. In: Proceedings of the EMNLP workshop BlackboxNLP: analyzing and interpreting neural networks for NLP, pp 353–355
13. Bowman SR, Angeli G, Potts C, Manning CD (2015) A large annotated corpus for learning natural language inference. In: Proceedings of the Conference on empirical methods in natural language processing (EMNLP). Association for Computational Linguistics
14. Rajpurkar P, Zhang J, Lopyrev K, Liang P (2016) Squad: 100,000 + questions for machine comprehension of text. In: Proceedings of the conference on empirical methods in natural language processing, pp 2383–2392