Transformer-based Pre-trained Models for Natural Language Processing

Eric Wang, Jacky Chen, Ziyang Guo, Zirong Huang[1]

Abstract

Recently, the prosperity of transformer-based pre-trained models (PTMs) has greatly promoted the progress in solving complicated natural language processing (NLP) tasks. In this project, we provided a comprehensive review and evaluation of some pre-trained models. Specifically, we firstly introduced the required knowledge for language representation learning and transfer learning and discussed the structural differences and advantages of these different PTMs. Then we performed experiments applying these PTMs to different downstream tasks for comprehensive evaluation. This project aims to serve the future learners and the community as in-depth theoretical analysis and useful benchmark test results.

1. Introduction

The emergence of deep learning has revolutionized the natural language processing (NLP) field. Specifically, different techniques and architectures that hire neural networks, including convolutional neural network (CNNs) (Kalchbrenner et al., 2014), recurrent neural network (RNNs) (Liu et al., 2016), graph neural network (GNNs) (Marcheggiani et al., 2018), long short-term memory networks (LSTMs) (Tai et al., 2015) and attention mechanisms (Vaswani et al., 2017), provide promising solutions to large scale complicated NLP tasks by solving the feature engineering problem to some extent. Neural network-based methods usually leverage low-dimensional and dense vectors (embeddings) to implicitly learn the syntactic or semantic features of the speech and text in languages, thus providing great representational power and adaptability.

Though neural network-based models establish a more complete semantic structure representation space to achieve better understanding of text, they often have an extremely large number of parameters and deep structures, which makes it difficult for these models to generalize between different NLP tasks. In addition, it has always been a huge challenge how to use massive unlabeled dataset to allow the networks to obtain a general understanding of the syntactic or semantic features in texts.

Recent substantial work shows that pre-trained models (PTMs), trained on large-scale unlabeled data, are able to learn universal language representation. Applying transfer learning without training the models from scratch, one can simply fine-tune the PTMs to better accomplish the downstream tasks including sequence labeling, documentation classification, named entity recognition, document summarization, question answering and language modelling. In addition, since large scale dataset is necessary for the deep neural models to prevent from over-fitting and a randomly initialized deep neural network is easy to overfit to small dataset, pre-training is an effective regularization method to avoid overfitting on small dataset for the downstream tasks.

A large number of language models proposed recent years that use recurrent structures and encoder-decoder architectures, including RNNs and LSTMs, have achieved pleasant progress in many NLP tasks. However, the inherent sequential essence of recurrent architectures hinders parallelization across training samples. Instead, Transformer, a model structure that avoids recurrent cycles, relies entirely on the attention mechanism to model the global dependencies of input and output. Transformer not only breaks through the limitation that the RNNs cannot be calculated in parallel, but also produces the more interpretable model. In other words, the attention distribution can be examined from the model and individual attention heads can learn to perform different tasks. So, to discuss the state-of-the-art techniques, architectures and performances of frontier deep neural models in the NLP field, this project focuses more on transformer-based pre-trained models.

Self-supervised learning is one of the most important approaches to train PTMs. Specifically, self-supervised learning mainly uses auxiliary tasks to mine its own supervision information from large-scale unsupervised data and trains the neural models through this constructed supervision information, so that it can learn valuable representations for downstream tasks. Particularly, this project is more interested in masked language model framework (MLM), permuted language modeling (PLM), denoising autoencoder

[1]Department of Mathematics, University of Waterloo, Waterloo, CA. Correspondence to: Eric Wang<e246wang@uwaterloo.ca>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

(DAE) and next sentence prediction (NSP) widely adopted for self-supervised language pre-training, which basically covers the mainstream pre-training tasks in this field. More detailed description about these pre-training tasks are introduced in the subsequent sections.

In this project, we mainly selected four typical PTMs within the topic, T5 (Raffel et al., 2020), XLNet (Yang et al., 2019), BART (Lewis et al., 2019) and DistilBERT (Sanh et al., 2019), respectively corresponding to four different pre-training tasks, MLM, PLM, DAE and NSP. We also intensively discussed their structural advantages and differences from the point of view of architectures. Then we evaluated their performances in terms of three important downstream tasks including language modeling, text summarization and question answering. We used different evaluation metrics, including but not limited to EM and F1, to examine the performance of PTMs on different downstream tasks. The involved pre-train models are achieved from Hugging Face repository (Wolf et al., 2020). The used dataset for downstream language modelling, text summarization and question answering tasks are WikiText 2 (Merity et al., 2016), CNN / DailyMail (See et al., 2017) and (Hermann et al., 2015) and SQuAD 2 (Rajpurkar et al., 2018). Besides, we also analyzed the limitations of the existing PTMs and propose the possible directions for the future research.

The contribution of this project can be summarized as it follows.

(1) Comprehensive review: we provided comprehensive introduction and review for the typical transformer-based PTMs using mainstream pre-training tasks, including background knowledge, related techniques, architectures and structural advantages and differences, all of which can serve as in-depth theoretical analysis for future learners.

(2) Benchmark results: we evaluated the different PTMs fine-tuned for different downstream tasks using general dataset. The qualitative and quantitative results can support the community as the useful benchmark test results of evaluation and comparisons for future research.

2. Background

2.1. Contextual embeddings

For NLP tasks, it is vital for a language representation to capture the implicit linguistic rules and common sense knowledge hiding in text data, such as lexical meanings, syntactic structures, semantic roles, and even pragmatics (Qiu et al., 2020). Deep learning based language representation learning methods, such as Word2Vec (Mikolov et al., 2013) and Doc2Vec (Le & Mikolov, 2014), generally use low dimensional real-value vectors to represent the meaning of a piece of text including words, sentences or documents. Though each dimension of the vector has no corresponding sense, the whole represents a concrete concept. Recent transformer-based PTMs train contextual embeddings. In other words, the embedding for a piece of text varies dynamically according to the context it appears in. For example, BERT (Devlin et al., 2018) has an advantage over models like Word2Vec because while each word has a fixed representation under Word2Vec regardless of the context within which the word appears, BERT produces outputs contextualized embedding that are dynamically informed by the words around them.

One of the greatest advantages of contextual embeddings is that they can help alleviate the problems of polysemous and context dependent nature of words where it is necessary to distinguish the semantics of words in different contexts (Qiu et al., 2020). Figure 1(b) shows a generic neural network architecture of transformer-based PTMs for NLP tasks where $x_1, x_2, ..., x_T$ denote the tokens $x_t \in \mathcal{V}$ that correspond to words or sub-words in a text. The contextual embeddings $\mathbf{h}_t$ can be represented by

$$[\mathbf{h}_1, \mathbf{h}_1, ..., \mathbf{h}_T] = f_{enc}(x1, x2, ..., x_T) \quad (1)$$

where $f_{enc}(\cdot)$ denotes a neural encoder.

2.2. Transformer

The neural contextual encoders can generally be classified as sequence models and non-sequence models. Particularly, sequence models, including RNNs and LSTMs, generally hire recurrent models to capture local context of a word in sequential order, which is considerably inefficient and subject to the long-term dependency problem. However, the transformer model revolutionized the implementation of attention by dispensing with recurrence and convolutions and, alternatively, relying solely on a self-attention mechanism, which uses a fully-connected graph to model the relation of every two words and let the model learn the structure by itself (Qiu et al., 2020). Specifically, the connection weights are dynamically computed by the self-attention mechanism, which implicitly indicates the connection between words. The PTMs involved in this project use different architectures of transformer, which would be further discussed in the following sections. Figure 1(a) clarifies the relationships between tokens and the contextual representations through a generic self-attention model.

Because of the complex structure and less model bias, transformer usually requires a large training corpus and is easier to overfit on small dataset. But transformer is able to directly model the dependency between every two words in an input sequence, which is more powerful and suitable to model long range dependency of language (Qiu et al., 2020). Meanwhile, the property that the attention mechanism pro-

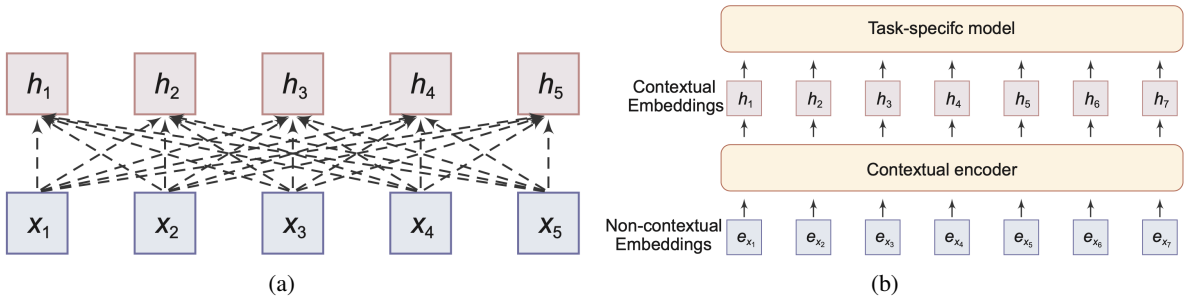


Figure 1. (a) self-attention model (Qiu et al., 2020). (b) neural network architecture for NLP (Qiu et al., 2020).

vides context for any position in the input sequence allows for more parallelization than sequence models. These factors combined to contribute to the rise of PTMs for NLP tasks.

2.3. Transfer learning

The training cost of a deep neural network increases with the amount of data and the number of network parameters. In particular, when we try to deal with real-life problems such as image recognition, language modeling, etc., it is very common to include some hidden layers in the model. At the same time, adding one more hidden layer will consume huge computing resources. Transfer learning allows us to make small changes on the basis of models trained by others and put them into use.

Specifically, neural networks need to use a large amount of data to train and extract features and information from the data, and then convert them into corresponding weights. These weights can be extracted and transferred to other neural networks. We "transfer" these learned features, so there is no need to train a neural network from scratch. We can directly use the corresponding structure and weights, and apply them to the problem we are facing. In other words, we transfer the pre-trained model to the downstream task we are dealing with. Since the pre-training model has been well trained, we often do not want to modify too many weights in a short period of time. When it is used in transfer learning, it is often only fine tuned.

2.4. Self-supervised learning

Before fine tuning the model for any downstream task, model pre-training is necessary. Model pre-training always uses self-supervised learning, which is a blend of supervised learning and unsupervised learning. The learning paradigm of self-supervised learning is entirely the same as supervised learning, but the labels of training data are generated automatically. The key idea of self-supervised learning is to predict any part of the input from other parts in some form. Because unlabeled data is ubiquitous, then we can pre-train models on these large unlabeled corpora to learn the universal representation of language effortlessly. After pre-training, the model can be fine tuned to a particular task with a labeled dataset. For example, a common way to fine-tune a BERT model is to add a task-specific network layer on the pre-trained BERT. Then the new model can be obtained after training on the specific data set for the downstream task.

One of the major differences between PTMs is the usage of pre-training tasks, which is crucial for learning the universal representation of language. Some widely-used mainstream pre-training tasks used in self-supervised learning would be introduced in the following sections.

3. Methods

In this section, we comprehensively introduce four famous PTMs that we used to fine-tune, DistilBERT, XLNet, T5 and BART, for the downstream tasks. These PTMs basically cover the mainstream pre-training tasks, including masked language model framework (MLM), per-muted language modeling (PLM), denoising autoencoder (DAE) and next sentence prediction (NSP). Besides, these models are mainly selected by faster and lighter standard given our limited computational power.

3.1. DistilBERT

The structure of BERT implements a contextual information encoding by using the encoder part of the transformer. The self-attention mechanism in the encoder makes use of the contextual tokens when encoding a token, instead of inputting sentences in reverse order like Bi-LSTM, so BERT is an actual bidirectional contextual encoding model, which is very important to many downstream tasks such as question answering. Additionally, BERT provides a unified structure for solving various downstream tasks. When we want to fine-tune specific tasks, we only need to add some network layers to the original structure. In this way, the difference between the pre-trained network structure and

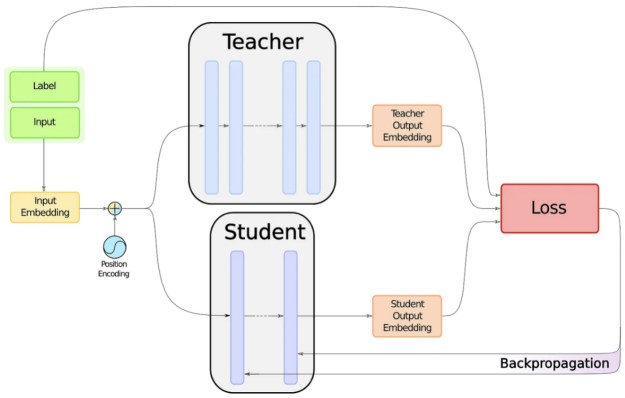


Figure 2. knowledge distillation in DistilBERT (Reboul, 2021).

the network structure of specific downstream tasks is very small, which helps to preserve the features learned during BERT pre-training as much as possible.

3.1.1. KNOWLEDGE DISTILLATION

Since BERT was proposed, large-scale pre-trained language models have become a basic tool for NLP tasks, but the fact that they usually have hundreds of millions of parameters leads to high training costs even for fine-tuning. DistilBERT (Sanh et al., 2019) uses a straightforward way, knowledge distillation, to compress model with little impact on the performance of the base BERT model. The basic idea of knowledge distillation is to use a large model to train a small model, just like a teacher with a student. Unlike direct end-to-end training, the teacher model can tell the output vector of the middle layer of the student model, so the model is easier to converge. Specifically, DistilBERT first trains a base BERT model as a complex complete teacher model, and then trains a simple student model, and makes the distribution of the student model as close as possible to the distribution of the teacher model by minimizing the distillation loss over the soft target probabilities of the teacher as

$$\mathcal{L} = \sum_i t_i * log(s_i), \tag{2}$$

where t_i and s_i are the probabilities estimated by the teacher model and the student model respectively. Figure 2 shows the complete training process of DistilBERT.

3.1.2. MASKED LANGUAGE MODEL

Masked language model (MLM) is a pre-training task used in BERT that attempts to predict the masked words in a sentence given the rest of the words. Specifically, the token, or a span of token, is replaced with a mask token. The goal then becomes reconstructing the original sequence, i.e. to reveal what is hidden under the mask as seen in Figure 3, where no text needs to be labeled manually in order to predict the missing values. Some of the words are masked, but the underlying word is available during the improvement step of the back propagation. MLM is generally solved as a classification problem, where the masked sequences are fed into the encoder whose output are further fed into a softmax classifier to predict the masked token. The loss function of MLM can be represented as

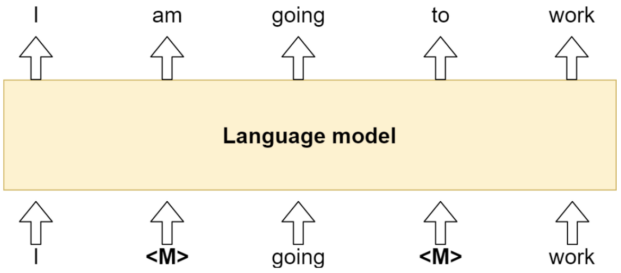


Figure 3. masked language model (Verma, 2020).

$$\mathcal{L}_{MLM} = -\sum_{\hat{x} \in m(\mathbf{x})} \log p(\hat{x} | \mathbf{x}_{\backslash m(\mathbf{x})}), \tag{3}$$

where $\mathbf{x} = [x_1, x_2, ..., x_T]$, $m(\mathbf{x})$ and $\mathbf{x}_{\backslash m(\mathbf{x})}$ denote a sequence, the masked words from $\mathbf{x}$ and the remaining words respectively. All below notations follow these definitions.

3.1.3. NEXT SENTENCE PREDICTION

Another pre-training task used in BERT is next sentence prediction (NSP). Many important downstream tasks, such as question answering and natural language inference, are based on understanding the relationship between two text sentences, which cannot be directly captured by language modeling. In order to train a model that understands the relationship between sentences, we can pre-train a NSP binary classification task. This task requires the input of the model to be two sentences. With 50% probability the second sentence is the next of first sentence, that is, a sentence is randomly selected from the corpus as the second sentence (Qiu et al., 2020). In this way the model can learn sentence-level relationships in corpus. This pre-training task is very helpful to text generation tasks. The loss function of NSP can be represented as

$$\mathcal{L}_{NSP} = -\log p(t | \mathbf{x}, \mathbf{y}), \tag{4}$$

where $t = 1$ and $\mathbf{x}$ and $\mathbf{y}$ are continuous segments from the corpus.

3.2. XLNet

In addition to autoencoding language modeling (AE), such as BERT, another autoregressive language modeling (AR) mainly predicts the token at the current moment based on the tokens that appear before or behind, such as ELMO (Peters et al., 2018) and GPT (Radford et al., 2018). The main advantage of AR models is that their reasoning naturally fits the generative process of generative tasks. However, the disadvantage is that they can only exploit uni-directional semantics and cannot exploit contextual information at the same time. For example, ELMO, which stitches bidirectional AR models, does not perform very well on certain downstream tasks that require accuracy.

The AE models do not have the autoregressive nature brought by AR models and learn the dependence between predicted tokens, and AR models do not have the representation learning of deep bidirectional information brought by AE model. XLNet proposed a permutation language model (PLM) that unifies the advantages of AR and AE models. It introduced contextual information by optimizing the expected likelihood of all decomposable permutations and combinations of the joint probability distribution function.

3.2.1. Permutation Language Model

PLM is a language modeling task on a random permutation of input sequences. A permutation is randomly sampled from all possible permutations. Then some of the tokens in the permuted sequence are chosen as the target, and the model is trained to predict these targets, depending on the rest of the tokens and the natural positions of targets (Qiu et al., 2020). Specifically, an arrangement of a sentence is selected, then a certain amount of words at the end are covered, and finally AR modeling is used to predict the covered words in sequence step by step according to this arrangement. By randomly picking one of the permutations, PLM learns bidirectional information through the uni-directional essence of AR. The specific implementation of permutation is to operate using the attention mask in the encoder part of the transformer. Figure 4 shows how PLM is used to learn bidirectional information. The loss function of PLM can be represented as

$$\mathcal{L}_{PLM} = -\sum_{t=1}^{T} \log p(z_t|\mathbf{z}_{<t}), \tag{5}$$

where $\mathbf{z} = perm(\mathbf{x})$ is a permutation of $\mathbf{x}$ with random order.

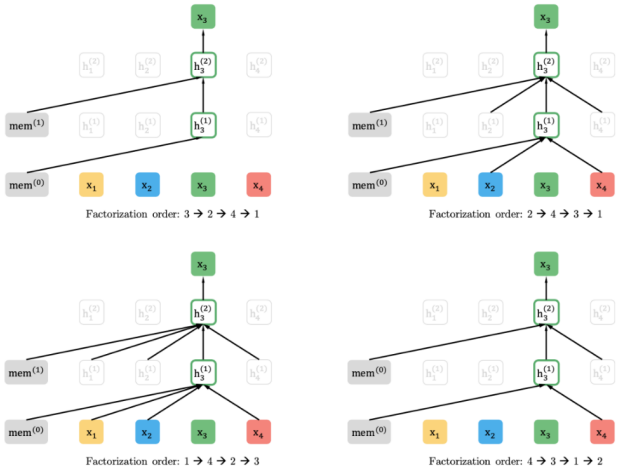


Figure 4. Permutation language model (Yang et al., 2019). The words in the sequence are randomly shuffled, so that, for a word x_i, its original context words may appear before itself. When we want to predict the word x_3, if the order of 1, 2, 3, and 4 is changed, x_4 may appear in the preceding context of x_3. So, although a model only considers the preceding context to make the prediction, it is still able to touch information about the original whole context.

3.3. T5

DistilBERT and XLNet only hire the encoder part of transformer architecture, which is more suitable for tasks that only need to understand the input semantics, such as sentence classification and named entity recognition. Decoder-based models, such as GPT, are more suitable for generative tasks, such as text generation. The transformer-based encoder-decoder models, or Seq2Seq models, use both parts of the transformer architecture. At each stage, the encoder's attention layer has access to all the words in the initial input sentence, while the decoder's attention layer can only access words preceding a given word in the input, or already decoded generated words. These models can be pre-trained using either encoder or decoder model objectives, but often include more complex tasks such as automatic summarization, translation, or generative question answering.

The T5 model can be considered as a standard transformer, with some modifications, for text-to-text transfer, so that all NLP tasks can be described as text-to-text problems to solve. Specifically, T5 unifies task-specific formats by adding prefixes to input sequences. A uniform input format lets a transformer model produce a sequence of results, no matter what the problem is. This unified process allows the use of the same models, hyperparameters and optimizers for a wide range of tasks.

3.3.1. Sequence-to-sequence Masked Language Modeling

The pre-training task T5 uses is sequence-to-sequence masked language modeling (Seq2Seq MLM), a special case of MLM. Specifically, the use of complete transformer

mechanism allows us to leverage encoder-decoder architecture for MLM, where masked sequences are still fed into the encoder, but the decoder sequentially produces the masked tokens in an auto-regression fashion (Qiu et al., 2020), unlike the classification problem in regular MLM. The loss function of Seq2Seq MLM can be represented as

$$\mathcal{L}_{S2SMLM} = -\sum_{t=i}^{j} \log p(x_t|\mathbf{x}_{\backslash \mathbf{x}_{i:j}}, \mathbf{x}_{i:t-1}), \quad (6)$$

where $\mathbf{x}_{i:j}$ denotes an masked n-gram span from i to j in $\mathbf{x}$.

3.4. BART

In natural language understanding (NLU) and natural language generation (NLG) tasks, the results of simply applying pre-trained language models such as BERT is not desirable, which is mainly due to the difference between the pre-training stage and the downstream task stage. Another important transformer-based encoder-decoder model to solve this problem is BART, which can be considered as a transformer with both contextual information and autoregressive properties and uses a pre-training method, denoising autoencoder (DAE), that meets the generation task. T5 is also a encoder-decoder model that is pre-trained with a corrupted input, but it is pre-trained by performing token masking and uses the training data to solve a Seq2Seq problem, different from the BART which uses the sequence for causal language model training.

3.4.1. Denoising autoencoder

BART leverages the characteristics of BERT's bidirectional encoder and GPT's left-to-right decoder, and is based on the standard Seq2Seq transformer model, which makes it more suitable for text generation scenarios than BERT. Compared to GPT, more bidirectional context information is provided. Figure 5 simply shows the process of BART pre-training. Specifically, BART combines the pre-training process of BERT and GPT in the encoder-decoder architecture. First, the input sequence can be destroyed by token masking, token deletion, text infilling, sentence permutation and document rotation. Then the sequence is encoded by the encoder and passed to the decoder required to reconstruct the original sequence. This architecture makes the model more flexible for wide NLU or NLG tasks and achieves desirable performance on both. The loss function of DAE can be represented as

$$\mathcal{L}_{DAE} = -\sum_{t=1}^{T} \log p(x_t|\hat{\mathbf{x}}, \mathbf{x}_{<t}), \quad (7)$$

where $\hat{\mathbf{x}}$ is randomly perturbed text from $\mathbf{x}$.

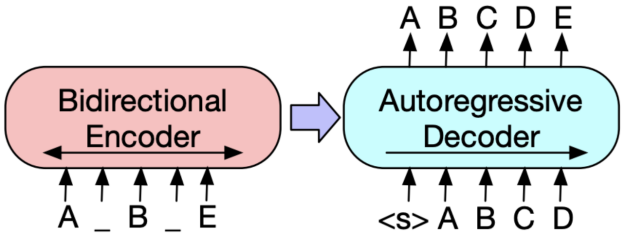


Figure 5. BART pre-training (bar).

4. Downstream Tasks

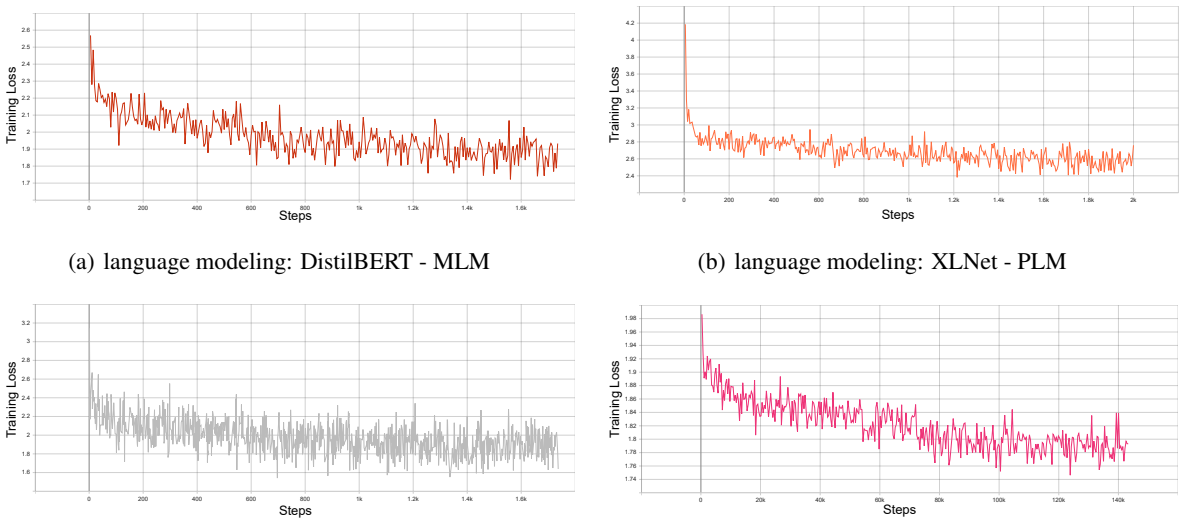
In this section, we evaluate the performances of four models in terms of three important downstream tasks – language modeling, text summarization and question answering.

4.1. Learning

We used Google Colab Pro GPU to fine-tune all models in this section. The hyper-parameters, including epoch number, batch size and learning rate, are primarily referenced from the original papers with some modification according to the performance in different trials. AdamW with weight decay (Loshchilov & Hutter, 2019) and initial learning rate of 3e-5 is hired as the optimizer. The involved pre-trained models, training scripts and evaluation metrics implementation are achieved and modified from Hugging Face repository. Figures 6 and 7 show the training curves and hyper-parameters with which we obtained the best performance for every experiment reported in Table 1. We trained the model with adequate epoch number to ensure that the learning process has reached the plateau as shown in Figures 6 and 7.

4.2. Language modeling

Language modeling, which predicts words in a sequence, is one of the most basic NLP tasks and widely used to evaluate large language models. Generally speaking, the encoder-based models are designed to work on the tasks that require understanding the semantics of entire sentences, such as sentence classification, named entity recognition and extractive question answering, thanks to the MLM and PLM pre-training methods. Therefore, it is more fair to perform the comparative experiments about language modeling within the encoder-based models. We performed the experiments of fine-tuning DistilBERT and XLNet on WikiText 2 dataset. Since MLM and PLM adopt the different loss functions, we only display the training cost in this part and the performance for question answering task would be discussed in the following part. The experiment results reported in Table 1 show that the training efficiency of DistilBERT is one-third higher than that of XLNet, which is reasonable because DistilBERT leverages knowledge distil-



(a) language modeling: DistilBERT - MLM

(b) language modeling: XLNet - PLM

(c) text summarization: T5 - Seq2Seq MLM

(d) text summarization: BART - DAE

Figure 6. (a-b) epoch = 3, batch size = 8, Adam with βs = (0.9, 0.999) and ϵ = 1e-08. (c-d) epoch = 2, batch size = 4, Adam with βs = (0.9, 0.999) and ϵ = 1e-08.

Tasks	LM (WikiText 2)	TS (CNN / DailyMail)			QA (SQuAD 2)	
Metrics	Efficiency	ROUGE-1	ROUGE-2	ROUGE-L	F1-Score	Exact
DistilBERT	**26 min**	—	—	—	67.09	65.28
XLNet	34 min	—	—	—	77.19	73.86
T5	—	23.39	9.01	16.99	81.92	76.84
BART	—	**28.24**	**13.88**	**17.39**	**83.73**	**82.45**

some evaluation metrics: efficiency (training time per epoch), ROUGE (recall-oriented understudy for gisting evaluation) and exact (exact match).

Table 1. experiment results.

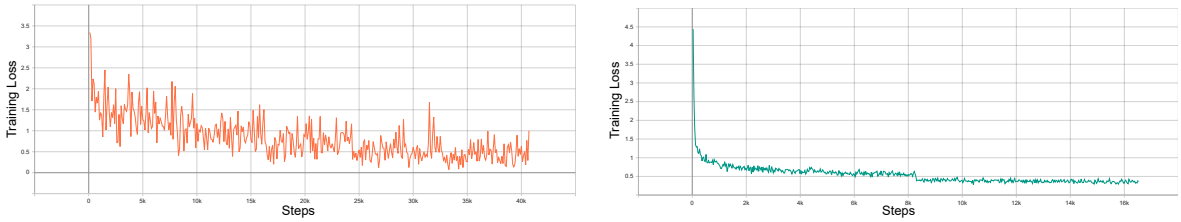
lation to compress the model.

4.3. Text summarization

Text summarization, one of the most important text generation tasks, produces a shorter version of a document but preserving the key information. Due to the encoder decoder-based seq2seq and attention paradigm, the generative models T5 and BART are more suitable than DistilBERT and XLNet for the tasks that require generating new sentences given sequences, so we only compare the performance of text summarization between T5 and BART. We performed the experiments fine-tuning T5 and BART on CNN / DailyMail dataset. The ROUGE indicator (Lin, 2004), imported from Hugging Face repository, is a common evaluation indicator in the fields of machine translation and automatic summarization. It calculates the corresponding score by comparing the summary from the model and the reference answer. The experiment results reported in Table 1 show that BART has stronger similarity between candidates and references than T5, which is reasonable considering that DAE is a more flexible and robust pre-training task than Seq2Seq MLM.

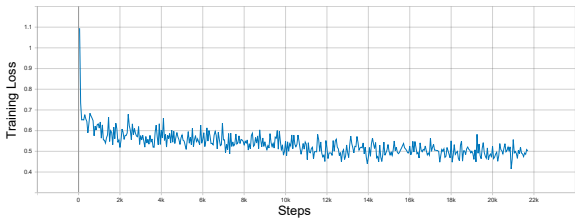
4.4. Question answering

Question answering, which retrieves the answer to a question from a given context, is widely used in many application scenarios including search engine and information retrieval tools. Though DistilBERT, XLNet, T5 and BART have different model architecture (only transformer encoder or transformer encoder-decoder), they all can work on question answering task based on their own inferencing mechanism, so we compare the performance of question answering among four models. We performed the experiments fine-tuning on SQuAD 2 dataset. The F1 and exact match metrics are imported from Hugging Face repository. The experiment results reported in Table 1 show that BART has better question answering accuracy than T5, though T5 has the largest number of parameters. Encoder-based models DistilBERT and XLNet performed not very well compared to complete transformer-based models. Figure 8 shows the

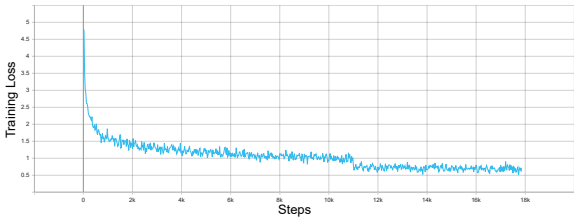


(a) question answering: DistilBERT - MLM

(b) question answering: XLNet - PLM



(c) question answering: T5 - Seq2Seq MLM



(d) question answering: BART - DAE

Figure 7. (a) epoch = 5, batch size = 16, Adam with ϵ = 5e-05. (b) epoch = 4, batch size = 2, Adam with βs = (0.9, 0.999) and ϵ = 1e-08. (c) epoch = 2, batch size = 12, Adam with βs = (0.9, 0.999) and ϵ = 1e-08. (d) epoch = 2, batch size = 12, Adam with βs = (0.9, 0.999) and ϵ = 1e-08.

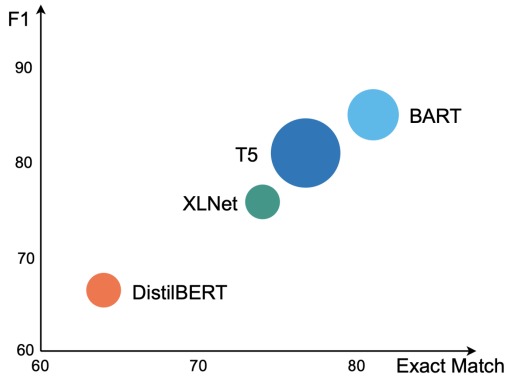


Figure 8. question answering (SQuAD 2) results.

bubble chart for the better visualization of the comparison between the different models, where the size of bubbles represents the number of parameters of models and the colors correspond to the model training curves shown in Figure 7. T5 transforms all NLP tasks to text-to-text problems using Seq2Seq MLM for pre-training, which may lead to the model not having the best fitting for only question answering task. However, under the methods of token masking, token deletion, text infilling, sentence permutation and document rotation, DAE used by BART restores the original lossless sequence from a partially disrupted sequence, which is naturally suitable for question answering task.

5. Conclusion

In this project, we provided a comprehensive review and evaluation of transformer-based PTMs, DistilBERT, XLNet, T5 and BART, given their own respective pre-training tasks. Specifically, we discussed the structural differences and advantages of these models and fine-tuned them for three downstream tasks, including language modeling, text summarization and question answering. We conclude that encoder-based model DistilBERT, with the knowledge distillation technique, has the greatest training efficiency and complete transformer-based model BART, with the DAE pre-training task, achieves the best performance on most downstream tasks. So the combination of enocder-decoder architecture and DAE pre-training task might be a good option when designing and applying large language model for complicated NLP problems, not limited to text generation tasks.

The community demands a comprehensive investigation and bench-marking for most of the recently proposed transformer-based LLMs, but, given the inadequate time and computational power, training four LLMs for three downstream tasks has already reached our limit. We would try experimenting more knowledge-enriched, multilingual and multi-modal models with different optimization techniques on more complicated NLP tasks in the future. We believe that these in-depth analysis and discussion would be an excellent outset for the future research.

References

Papers with code - bart explained. URL https://paperswithcode.com/method/bart.

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

Hermann, K. M., Kociský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. Teaching machines to read and comprehend. In *NIPS*, pp. 1693–1701, 2015. URL http://papers.nips.cc/paper/5945-teaching-machines-to-read-and-comprehend.

Kalchbrenner, N., Grefenstette, E., and Blunsom, P. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

Le, Q. and Mikolov, T. Distributed representations of sentences and documents. In *International conference on machine learning*, pp. 1188–1196. PMLR, 2014.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

Lin, C.-Y. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W04-1013.

Liu, P., Qiu, X., and Huang, X. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*, 2016.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization, 2019.

Marcheggiani, D., Bastings, J., and Titov, I. Exploiting semantics in neural machine translation with graph convolutional networks. *arXiv preprint arXiv:1804.08313*, 2018.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models, 2016.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://aclanthology.org/N18-1202.

Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., and Huang, X. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10): 1872–1897, 2020.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. Improving language understanding by generative pre-training. 2018.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Rajpurkar, P., Jia, R., and Liang, P. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

Reboul, R. O. Distillation of bert-like models: The theory, Dec 2021. URL https://towardsdatascience.com/distillation-of-bert-like-models-the-theory-32e19

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

See, A., Liu, P. J., and Manning, C. D. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1099. URL https://www.aclweb.org/anthology/P17-1099.

Tai, K. S., Socher, R., and Manning, C. D. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*, 2015.

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

Verma, A. Implementation of bert, Jul 2020. URL https://iq.opengenus.org/implementation-of-bert/.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

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