SCIENCE CHINA
Technological Sciences

Special Topic: Natural Language Processing Technology
• Review •

https://doi.org/10.1007/s11431-020-1647-3

Pre-trained models for natural language processing: A survey

QIU XiPeng[1,2*], SUN TianXiang[1,2], XU YiGe[1,2], SHAO YunFan[1,2], DAI Ning[1,2] & HUANG XuanJing[1,2]

[1] *School of Computer Science, Fudan University, Shanghai 200433, China;*
[2] *Shanghai Key Laboratory of Intelligent Information Processing, Shanghai 200433, China*

Received March 9, 2020; accepted May 21, 2020; published online September 15, 2020

Recently, the emergence of pre-trained models (PTMs) has brought natural language processing (NLP) to a new era. In this survey, we provide a comprehensive review of PTMs for NLP. We first briefly introduce language representation learning and its research progress. Then we systematically categorize existing PTMs based on a taxonomy from four different perspectives. Next, we describe how to adapt the knowledge of PTMs to downstream tasks. Finally, we outline some potential directions of PTMs for future research. This survey is purposed to be a hands-on guide for understanding, using, and developing PTMs for various NLP tasks.

deep learning, neural network, natural language processing, pre-trained model, distributed representation, word embedding, self-supervised learning, language modelling

Citation: Qiu X P, Sun T X, Xu Y G, et al. Pre-trained models for natural language processing: A survey. Sci China Tech Sci, 2020, 63, https://doi.org/10.1007/s11431-020-1647-3

1 Introduction

With the development of deep learning, various neural networks have been widely used to solve natural language processing (NLP) tasks, such as convolutional neural networks (CNNs) [1–3], recurrent neural networks (RNNs) [4, 5], graph-based neural networks (GNNs) [6–8] and attention mechanisms [9, 10]. One of the advantages of these neural models is their ability to alleviate thefeature engineering problem. Non-neural NLP methods usually heavily rely on the discrete handcrafted features, while neural methods usually use low-dimensional and dense vectors (aka.distributed representation) to implicitly represent the syntactic or semantic features of the language. These representations are learned in specific NLP tasks. Therefore, neural methods make it easy for people to develop various NLP systems.

Despite the success of neural models for NLP tasks, the performance improvement may be less significant compared with the computer vision (CV) field. The main reason is that current datasets for most supervised NLP tasks are rather small (except machine translation). Deep neural networks usually have a large number of parameters, which make them overfit on these small training data and do not generalize well in practice. Therefore, the early neural models for many NLP tasks were relatively shallow and usually consisted of only 1–3 neural layers.

Recently, substantial work has shown that pre-trained models (PTMs[1)]), on the large corpus can learn universal language representations, which are beneficial for downstream NLP tasks and can avoid training a new model from scratch. With the development of computational power, the emergence of the deep models (i.e., Transformer [10]), and the constant enhancement of training skills, the architecture of PTMs has been advanced from shallow to deep. Thefirst-generation PTMs aim to learn good word embeddings. Since these models themselves are no longer needed by down-

*Corresponding author (email: xpqiu@fudan.edu.cn)

1) PTMs are also known as pre-trained language models (PLMs). In this survey, we use PTMs for NLP instead of PLMs to avoid confusion with the narrow concept of statistical (or probabilistic) language models.

stream tasks, they are usually very shallow for computational efficiencies, such as Skip-Gram [11] and GloVe [12]. Although these pre-trained embeddings can capture semantic meanings of words, they are context-free and fail to capture higher-level concepts in context, such as polysemous disambiguation, syntactic structures, semantic roles, anaphora. Thesecond-generation PTMs focus on learning contextual word embeddings, such as CoVe [13], ELMo [14], OpenAI GPT [15] and BERT [16]. These learned encoders are still needed to represent words in context by downstream tasks. Besides, various pre-training tasks are also proposed to learn PTMs for different purposes.

The contributions of this survey can be summarized as follows.

(1) Comprehensive review. We provide a comprehensive review of PTMs for NLP, including background knowledge, model architecture, pre-training tasks, various extensions, adaption approaches, and applications.

(2) New taxonomy. We propose a taxonomy of PTMs for NLP, which categorizes existing PTMs from four different perspectives: 1) representation type, 2) model architecture; 3) type of pre-training task; 4) extensions for specific types of scenarios.

(3) Abundant resources. We collect abundant resources on PTMs, including open-source implementations of PTMs, visualization tools, corpora, and paper lists.

(4) Future directions. We discuss and analyze the limitations of existing PTMs. Also, we suggest possible future research directions.

The rest of the survey is organized as follows. Sect. 2 outlines the background concepts and commonly used notations of PTMs. Sect. 3 gives a brief overview of PTMs and clarifies the categorization of PTMs. Sect. 4 provides extensions of PTMs. Sect. 5 discusses how to transfer the knowledge of PTMs to downstream tasks. Sect. 6 gives the related resources on PTMs. Sect. 7 presents a collection of applications across various NLP tasks. Sect. 8 discusses the current challenges and suggests future directions. Sect. 9 summarizes the paper.

2 Background

2.1 Language representation learning

As suggested by ref. [17], a good representation should express general-purpose priors that are not task-specific but would be likely to be useful for a learning machine to solve AI-tasks. When it comes to language, a good representation should capture the implicit linguistic rules and common sense knowledge hiding in text data, such as lexical meanings, syntactic structures, semantic roles, and even pragmatics.

The core idea of distributed representation is to describe the meaning of a piece of text by low-dimensional real-valued vectors. And each dimension of the vector has no corresponding sense, while the whole represents a concrete concept. Figure 1 illustrates the generic neural architecture for NLP. There are two kinds of word embeddings: non-contextual and contextual embeddings. The difference between them is whether the embedding for a word dynamically changes according to the context it appears in.

Non-contextual embeddings The first step of representing language is to map discrete language symbols into a distributed embedding space. Formally, for each word (or subword) x in a vocabulary $\mathcal{V}$, we map it to a vector $\mathbf{e}_x \in \mathbb{R}^{D_e}$ with a lookup table $\mathbf{E} \in \mathbb{R}^{D_e \times |\mathcal{V}|}$, where D_e is a hyper-parameter indicating the dimension of token embeddings. These embeddings are trained on task data along with other model parameters.

There are two main limitations to this kind of embeddings. The first issue is that the embeddings are static. The embedding for a word does is always the same regardless of its context. Therefore, thesenon-contextual embeddings fail to model polysemous words. The second issue is the out-of-vocabulary problem. To tackle this problem, character-level word representations or sub-word representations are widely used in many NLP tasks, such as CharCNN [18], FastText [19] and Byte-Pair Encoding (BPE) [20].

Contextual embeddings To address the issue of polysemous and the context-dependent nature of words, we need distinguish the semantics of words in different contexts. Given a text $x_1, x_2, \cdots, x_T$ where each token $x_t \in \mathcal{V}$ is a word or sub-word, the contextual representation of x_t depends on the whole text.

$$[\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_T] = f_{\text{enc}}(x_1, x_2, \cdots, x_T), \tag{1}$$

where $f_{\text{enc}}(\cdot)$ is neural encoder, which is described in Sect. 2.2, and $\mathbf{h}_t$ is called contextual embedding or dynamical embedding of token x_t because of the contextual information included in.

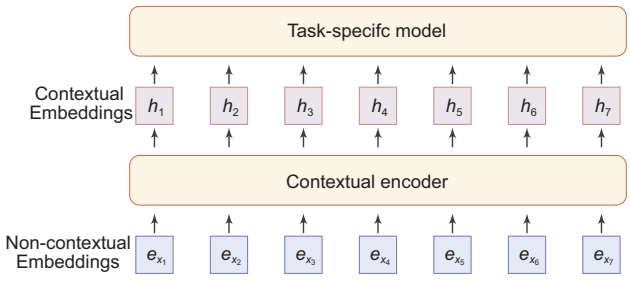

Figure 1 (Color online) Generic neural architecture for NLP.

2.2 Neural contextual encoders

Most of the neural contextual encoders can be classified into two categories: sequence models and non-sequence models. Figure 2 illustrates three representative architectures.

2.2.1 *Sequence models*

Sequence models usually capture local context of a word in sequential order.

Convolutional models Convolutional models take the embeddings of words in the input sentence and capture the meaning of a word by aggregating the local information from its neighbors by convolution operations [2].

Recurrent models Recurrent models capture the contextual representations of words with short memory, such as LSTMs [21] and GRUs [22]. In practice, bi-directional LSTMs or GRUs are used to collect information from both sides of a word, but its performance is often affected by the long-term dependency problem.

2.2.2 *Non-sequence models*

Non-sequence models learn the contextual representation with a pre-defined tree or graph structure between words, such as the syntactic structure or semantic relation. Some popular non-sequence models include Recursive NN [6], TreeLSTM [7,23], and GCN [24].

Although the linguistic-aware graph structure can provide useful inductive bias, how to build a good graph structure is also a challenging problem. Besides, the structure depends heavily on expert knowledge or external NLP tools, such as the dependency parser.

Fully-connected self-attention model In practice, a more straightforward way is to use a fully-connected graph to model the relation of every two words and let the model learn the structure by itself. Usually, the connection weights are dynamically computed by the self-attention mechanism, which implicitly indicates the connection between words. A successful instance of fully-connected self-attention model is the Transformer [10], which also needs other supplement modules, such as positional embeddings, layer normalization, residual connections and position-wise feed-forward network (FFN) layers.

2.2.3 *Analysis*

Sequence models learn the contextual representation of the word with locality bias and are hard to capture the long-range interactions between words. Nevertheless, sequence models are usually easy to train and get good results for various NLP tasks.

In contrast, as an instantiated fully-connected self-attention model, the Transformer can directly model the dependency between every two words in a sequence, which is more powerful and suitable to model long range dependency of language. However, due to its heavy structure and less model bias, the Transformer usually requires a large training corpus and is easy to overfit on small or modestly-sized datasets [15,25].

Currently, the Transformer has become the mainstream architecture of PTMs due to its powerful capacity.

2.3 Why pre-training?

With the development of deep learning, the number of model parameters has increased rapidly. The much larger dataset is needed to fully train model parameters and prevent overfitting. However, building large-scale labeled datasets is a great challenge for most NLP tasks due to the extremely expensive annotation costs, especially for syntax and semantically related tasks.

In contrast, large-scale unlabeled corpora are relatively easy to construct. To leverage the huge unlabeled text data, we can first learn a good representation from them and then use these representations for other tasks. Recent studies have demonstrated significant performance gains on many NLP tasks with the help of the representation extracted from the PTMs on the large unannotated corpora.

The advantages of pre-training can be summarized as follows.

(1) Pre-training on the huge text corpus can learn universal language representations and help with the downstream tasks.

(2) Pre-training provides a better model initialization, which usually leads to a better generalization performance

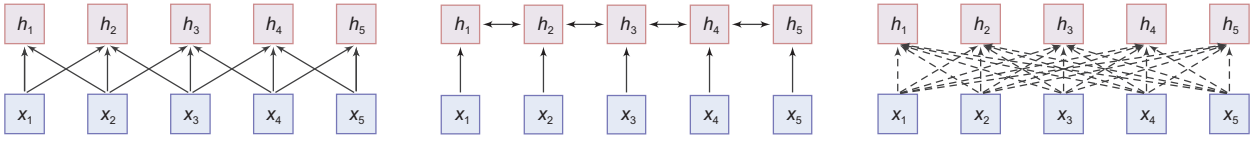

Figure 2 (Color online) Neural contextual encoders. (a) Convolutional model; (b) recurrent model; (c) fully-connected self-attention model.

and speeds up convergence on the target task.

(3) Pre-training can be regarded as a kind of regularization to avoid overfitting on small data [26].

2.4 A brief history of PTMs for NLP

Pre-training has always been an effective strategy to learn the parameters of deep neural networks, which are then fine-tuned on downstream tasks. As early as 2006, the breakthrough of deep learning came with greedy layer-wise unsupervised pre-training followed by supervised fine-tuning [27]. In CV, it has been in practice to pre-train models on the huge ImageNet corpus, and then fine-tune further on smaller data for different tasks. This is much better than a random initialization because the model learns general image features, which can then be used in various vision tasks.

In NLP, PTMs on large corpus have also been proven to be beneficial for the downstream NLP tasks, from the shallow word embedding to deep neural models.

2.4.1 First-generation PTMs: Pre-trained word embeddings

Representing words as dense vectors has a long history [28]. The "modern" word embedding is introduced in pioneer work of neural network language model (NNLM) [29]. Ref. [30] showed that the pre-trained word embedding on the unlabelled data could significantly improve many NLP tasks. To address the computational complexity, they learned word embeddings with pairwise ranking task instead of language modeling. Their work is the first attempt to obtain generic word embeddings useful for other tasks from unlabeled data. Ref. [11] showed that there is no need for deep neural networks to build good word embeddings. They propose two shallow architectures: Continuous Bag-of-Words (CBOW) and Skip-Gram (SG) models. Despite their simplicity, they can still learn high-quality word embeddings to capture the latent syntactic and semantic similarities among words. Word2vec is one of the most popular implementations of these models and makes the pre-trained word embeddings accessible for different tasks in NLP. Besides, GloVe [12] is also a widely-used model for obtaining pre-trained word embeddings, which are computed by global word-word co-occurrence statistics from a large corpus.

Although pre-trained word embeddings have been shown effective in NLP tasks, they are context-independent and mostly trained by shallow models. When used on a downstream task, the rest of the whole model still needs to be learned from scratch.

During the same time period, many researchers also try to learn embeddings of paragraph, sentence or document, such as paragraph vector [31], Skip-thought vectors [32], Context2Vec [33]. Different from their modern successors, these sentence embedding models try to encode input sentences into a fixed-dimensional vector representation, rather than the contextual representation for each token.

2.4.2 Second-generation PTMs: Pre-trained contextual encoders

Since most NLP tasks are beyond word-level, it is natural to pre-train the neural encoders on sentence-level or higher. The output vectors of neural encoders are also called contextual word embeddings since they represent the word semantics depending on its context.

Ref. [34] proposed the first successful instance of PTM for NLP. They initialized LSTMs with a language model (LM) or a sequence autoencoder, and found the pre-training can improve the training and generalization of LSTMs in many text classification tasks. Ref. [5] pre-trained a shared LSTM encoder with LM and fine-tuned it under the multi-task learning (MTL) framework. They found the pre-training and fine-tuning can further improve the performance of MTL for several text classification tasks. Ref. [35] found the Seq2Seq models can be significantly improved by unsupervised pre-training. The weights of both encoder and decoder are initialized with pre-trained weights of two language models and then fine-tuned with labeled data. Besides pre-training the contextual encoder with LM, ref. [13] pre-trained a deep LSTM encoder from an attentional sequence-to-sequence model with machine translation (MT). The context vectors (CoVe) output by the pre-trained encoder can improve the performance of a wide variety of common NLP tasks.

Since these precursor PTMs, the modern PTMs are usually trained with larger scale corpora, more powerful or deeper architectures (e.g., Transformer), and new pre-training tasks.

Ref. [14] pre-trained 2-layer LSTM encoder with a bidirectional language model (BiLM), consisting of a forward LM and a backward LM. The contextual representations output by the pre-trained BiLM, ELMo (Embeddings from Language Models), are shown to bring large improvements on a broad range of NLP tasks. Ref. [36] captured word meaning with contextual string embeddings pre-trained with character-level LM. However, these two PTMs are usually used as a feature extractor to produce the contextual word embeddings, which are fed into the main model for downstream tasks. Their parameters are fixed, and the rest parameters of the main model are still trained from scratch. ULMFiT (Universal Language Model Fine-tuning) [37] attempted to fine-tune pre-trained LM for text classification (TC) and achieved state-of-the-art results on six widely-used TC datasets. ULMFiT consists of 3 phases: (1) pre-training LM on general-domain data; (2) fine-tuning LM on target data; (3) fine-

tuning on the target task. ULMFiT also investigates some effective fine-tuning strategies, including discriminative fine-tuning, slanted triangular learning rates, and gradual unfreezing.

More recently, the very deep PTMs have shown their powerful ability in learning universal language representations: e.g., OpenAI GPT (Generative Pre-training) [15] and BERT (Bidirectional Encoder Representation from Transformer) [16]. Besides LM, an increasing number of self-supervised tasks (see Sect. 3.1) is proposed to make the PTMs capturing more knowledge form large scale text corpora.

Since ULMFiT and BERT, fine-tuning has become the mainstream approach to adapt PTMs for the downstream tasks.

3 Overview of PTMs

The major differences between PTMs are the usages of contextual encoders, pre-training tasks, and purposes. We have briefly introduced the architectures of contextual encoders in Sect. 2.2. In this section, we focus on the description of pre-training tasks and give a taxonomy of PTMs.

3.1 Pre-training tasks

The pre-training tasks are crucial for learning the universal representation of language. Usually, these pre-training tasks should be challenging and have substantial training data. In this section, we summarize the pre-training tasks into three categories: supervised learning, unsupervised learning, and self-supervised learning.

(1) Supervised learning (SL) is to learn a function that maps an input to an output based on training data consisting of input-output pairs.

(2) Unsupervised learning (UL) is to find some intrinsic knowledge from unlabeled data, such as clusters, densities, latent representations.

(3) Self-Supervised learning (SSL) is a blend of supervised learning and unsupervised learning[2)]. The learning paradigm of SSL is entirely the same as supervised learning, but the labels of training data are generated automatically. The key idea of SSL is to predict any part of the input from other parts in some form. For example, the masked language model (MLM) is a self-supervised task that attempts to predict the masked words in a sentence given the rest words.

In CV, many PTMs are trained on large supervised training sets like ImageNet. However, in NLP, the datasets of most supervised tasks are not large enough to train a good PTM. The only exception is machine translation (MT). A large-scale MT dataset, WMT 2017, consists of more than 7 million sentence pairs. Besides, MT is one of the most challenging tasks in NLP, and an encoder pre-trained on MT can benefit a variety of downstream NLP tasks. As a successful PTM, CoVe [13] is an encoder pre-trained on MT task and improves a wide variety of common NLP tasks: sentiment analysis (SST, IMDb), question classification (TREC), entailment (SNLI), and question answering (SQuAD).

In this section, we introduce some widely-used pre-training tasks in existing PTMs. We can regard these tasks as self-supervised learning. Table 1 also summarizes their loss functions.

3.1.1 Language modeling (LM)

The most common unsupervised task in NLP is probabilistic language modeling (LM), which is a classic probabilistic density estimation problem. Although LM is a general concept, in practice, LM often refers in particular to auto-regressive LM or unidirectional LM.

Given a text sequence $\mathbf{x}_{1:T} = [x_1, x_2, \cdots, x_T]$, its joint probability $p(x_{1:T})$ can be decomposed as

$$p(\mathbf{x}_{1:T}) = \prod_{t=1}^{T} p(x_t|\mathbf{x}_{0:t-1}), \tag{2}$$

where x_0 is special token indicating the begin of sequence.

The conditional probability $p(x_t|\mathbf{x}_{0:t-1})$ can be modeled by a probability distribution over the vocabulary given linguistic context $\mathbf{x}_{0:t-1}$. The context $\mathbf{x}_{0:t-1}$ is modeled by neural encoder $f_{\text{enc}}(\cdot)$, and the conditional probability is

$$p(x_t|\mathbf{x}_{0:t-1}) = g_{\text{LM}}\Big(f_{\text{enc}}(\mathbf{x}_{0:t-1})\Big), \tag{3}$$

where $g_{\text{LM}}(\cdot)$ is prediction layer.

Given a huge corpus, we can train the entire network with maximum likelihood estimation (MLE).

A drawback of unidirectional LM is that the representation of each token encodes only the leftward context tokens and itself. However, better contextual representations of text should encode contextual information from both directions. An improved solution is bidirectional LM (BiLM), which consists of two unidirectional LMs: a forward left-to-right LM and a backward right-to-left LM. For BiLM, ref. [38] proposed a two-tower model that the forward tower operates the left-to-right LM and the backward tower operates the right-to-left LM.

2) Indeed, it is hard to clearly distinguish the unsupervised learning and self-supervised learning. For clarification, we refer "unsupervised learning" to the learning without human-annotated supervised labels. The purpose of "self-supervised learning" is to learn the general knowledge from data rather than standard unsupervised objectives, such as density estimation.

Table 1 Loss functions of pre-training tasks[a)]

Task	Loss function	Description
LM	$\mathcal{L}_{\text{LM}} = -\sum_{t=1}^{T} \log p(x_t \mid \mathbf{x}_{<t})$	$\mathbf{x}_{<t} = x_1, x_2, \cdots, x_{t-1}$
MLM	$\mathcal{L}_{\text{MLM}} = -\sum_{\hat{x} \in m(\mathbf{x})} \log p\left(\hat{x} \mid \mathbf{x}_{\backslash m(\mathbf{x})}\right)$	$m(\mathbf{x})$ and $\mathbf{x}_{\backslash m(\mathbf{x})}$ denote the masked words from $\mathbf{x}$ and the rest words respectively
Seq2Seq MLM	$\mathcal{L}_{\text{S2SMLM}} = -\sum_{t=i}^{j} \log p\left(x_t \mid \mathbf{x}_{\backslash \mathbf{x}_{i:j}}, \mathbf{x}_{i:t-1}\right)$	$\mathbf{x}_{i:j}$ denotes an masked n-gram span from i to j in $\mathbf{x}$
PLM	$\mathcal{L}_{\text{PLM}} = -\sum_{t=1}^{T} \log p(z_t \mid \mathbf{z}_{<t})$	$\mathbf{z} = \text{perm}(\mathbf{x})$ is a permutation of $\mathbf{x}$ with random order
DAE	$\mathcal{L}_{\text{DAE}} = -\sum_{t=1}^{T} \log p(x_t \mid \hat{\mathbf{x}}, \mathbf{x}_{<t})$	$\hat{\mathbf{x}}$ is randomly perturbed text from $\mathbf{x}$
DIM	$\mathcal{L}_{\text{DIM}} = s(\hat{\mathbf{x}}_{i:j}, \mathbf{x}_{i:j}) - \log \sum_{\tilde{\mathbf{x}}_{i:j} \in \mathcal{N}} s(\hat{\mathbf{x}}_{i:j}, \tilde{\mathbf{x}}_{i:j})$	$\mathbf{x}_{i:j}$ denotes an n-gram span from i to j in $\mathbf{x}$, $\hat{\mathbf{x}}_{i:j}$ denotes a sentence masked at position i to j, and $\tilde{\mathbf{x}}_{i:j}$ denotes a randomly-sampled negative n-gram from corpus
NSP/SOP	$\mathcal{L}_{\text{NSP/SOP}} = -\log p(t \mid \mathbf{x}, \mathbf{y})$	$t = 1$ if $\mathbf{x}$ and $\mathbf{y}$ are continuous segments from corpus
RTD	$\mathcal{L}_{\text{RTD}} = -\sum_{t=1}^{T} \log p(y_t \mid \hat{\mathbf{x}})$	$y_t = \mathbf{1}(\hat{x}_t = x_t)$, $\hat{\mathbf{x}}$ is corrupted from $\mathbf{x}$

a) $\mathbf{x} = [x_1, x_2, \cdots, x_T]$ denotes a sequence.

3.1.2 *Masked language modeling (MLM)*

Masked language modeling (MLM) is first proposed by ref. [39], who referred to this as a Cloze task. Ref. [16] adapted this task as a novel pre-training task to overcome the drawback of the standard unidirectional LM. Loosely speaking, MLM first masks out some tokens from the input sentences and then trains the model to predict the masked tokens by the rest of the tokens. However, this pre-training method will create a mismatch between the pre-training phase and the fine-tuning phase because the mask token does not appear during the fine-tuning phase. Empirically, to deal with this issue, ref. [16] used a special `[MASK]` token 80% of the time, a random token 10% of the time and the original token 10% of the time to perform masking.

Sequence-to-sequence MLM (Seq2Seq MLM) MLM is usually solved as classification problem. We feed the masked sequences to a neural encoder whose output vectors are further fed into a softmax classifier to predict the masked token. Alternatively, we can use encoder-decoder (aka. sequence-to-sequence) architecture for MLM, in which the encoder is fed a masked sequence, and the decoder sequentially produces the masked tokens in auto-regression fashion. We refer to this kind of MLM as sequence-to-sequence MLM (Seq2Seq MLM), which is used in MASS [40] and T5 [41]. Seq2Seq MLM can benefit the Seq2Seq-style downstream tasks, such as question answering, summarization, and machine translation.

Enhanced masked language modeling (E-MLM) Concurrently, there are multiple research proposing different enhanced versions of MLM to further improve on BERT. Instead of static masking, RoBERTa [42] improves BERT by dynamic masking.

UniLM [43, 44] extends the task of mask prediction on three types of language modeling tasks: unidirectional, bidirectional, and sequence-to-sequence prediction. XLM [45] performs MLM on a concatenation of parallel bilingual sentence pairs, called Translation Language Modeling (TLM). SpanBERT [46] replaces MLM with Random Contiguous Words Masking and Span Boundary Objective (SBO) to integrate structure information into pre-training, which requires the system to predict masked spans based on span boundaries. Besides, StructBERT [47] introduces the Span Order Recovery task to further incorporate language structures.

Another way to enrich MLM is to incorporate external knowledge (see Sect. 4.1).

3.1.3 *Permuted language modeling (PLM)*

Despite the wide use of the MLM task in pre-training, ref. [48] claimed that some special tokens used in the pre-training of MLM, like `[MASK]`, are absent when the model is applied on downstream tasks, leading to a gap between pre-training and fine-tuning. To overcome this issue, Permuted Language Modeling (PLM) [48] is a pre-training objective to replace MLM. In short, PLM is a language modeling task on a random permutation of input sequences. A permutation is ran-

domly 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. Note that this permutation does not affect the natural positions of sequences and only defines the order of token predictions. In practice, only the last few tokens in the permuted sequences are predicted, due to the slow convergence. And a special two-stream self-attention is introduced for target-aware representations.

3.1.4 *Denoising autoencoder (DAE)*

Denoising autoencoder (DAE) takes a partially corrupted input and aims to recover the original undistorted input. Specific to language, a sequence-to-sequence model, such as the standard Transformer, is used to reconstruct the original text. There are several ways to corrupt text [49].

(1) Token masking. Randomly sampling tokens from the input and replacing them with `[MASK]` elements.

(2) Token deletion. Randomly deleting tokens from the input. Different from token masking, the model needs to decide the positions of missing inputs.

(3) Text infilling. Like SpanBERT, a number of text spans are sampled and replaced with a single `[MASK]` token. Each span length is drawn from a Poisson distribution ($\lambda = 3$). The model needs to predict how many tokens are missing from a span.

(4) Sentence permutation. Dividing a document into sentences based on full stops and shuffling these sentences in random order.

(5) Document rotation. Selecting a token uniformly at random and rotating the document so that it begins with that token. The model needs to identify the real start position of the document.

3.1.5 *Contrastive learning (CTL)*

Contrastive learning [50] assumes some observed pairs of text that are more semantically similar than randomly sampled text. A score function $s(x, y)$ for text pair (x, y) is learned to minimize the objective function:

$$\mathcal{L}_{\text{CTL}} = \mathbb{E}_{x,y^+,y^-}\left[-\log \frac{\exp\left(s(x,y^+)\right)}{\exp\left(s(x,y^+)\right)+\exp\left(s(x,y^-)\right)}\right], \quad (4)$$

where (x, y^+) are a similar pair and y^- is presumably dissimilar to x. y^+ and y^- are typically called positive and negative sample. The score function $s(x, y)$ is often computed by a learnable neural encoder in two ways: $s(x, y) = f_{\text{enc}(x)}^{\text{T}} f_{\text{enc}(y)}$ or $s(x, y) = f_{\text{enc}}(x \oplus y)$.

The idea behind CTL is "learning by comparison". Compared with LM, CTL usually has less computational complexity and therefore is desirable alternative training criteria for PTMs.

Ref. [30] proposed pairwise ranking task to distinguish real and fake phrases. The model needs to predict a higher score for a legal phrase than an incorrect phrase obtained by replacing its central word with a random word. Ref. [51] trained word embeddings efficiently with Noise-Contrastive Estimation (NCE) [52], which trains a binary classifier to distinguish real and fake samples. The idea of NCE is also used in the well-known word2vec embedding [11].

We briefly describe some recently proposed CTL tasks in the following paragraphs.

Deep InfoMax (DIM) Deep InfoMax (DIM) [53] is originally proposed for images, which improves the quality of the representation by maximizing the mutual information between an image representation and local regions of the image.

Ref. [54] applied DIM to language representation learning. The global representation of a sequence x is defined to be the hidden state of the first token (assumed to be a special start of sentence symbol) output by contextual encoder $f_{\text{enc}}(\mathbf{x})$. The objective of DIM is to assign a higher score for $f_{\text{enc}}(\mathbf{x}_{i:j})^{\text{T}} f_{\text{enc}}(\hat{\mathbf{x}}_{i:j})$ than $f_{\text{enc}}(\tilde{\mathbf{x}}_{i:j})^{\text{T}} f_{\text{enc}}(\hat{\mathbf{x}}_{i:j})$, where $\mathbf{x}_{i:j}$ denotes an n-gram[3)] span from i to j in $\mathbf{x}$, $\hat{\mathbf{x}}_{i:j}$ denotes a sentence masked at position i to j, and $\tilde{\mathbf{x}}_{i:j}$ denotes a randomly-sampled negative n-gram from corpus.

Replaced token detection (RTD) Replaced token detection (RTD) is the same as NCE but predicts whether a token is replaced given its surrounding context.

CBOW with negative sampling (CBOW-NS) [11] can be viewed as a simple version of RTD, in which the negative samples are randomly sampled from vocabulary with simple proposal distribution.

ELECTRA [55] improves RTD by utilizing a generator to replacing some tokens of a sequence. A generator G and a discriminator D are trained following a two-stage procedure: (1) train only the generator with MLM task for n_1 steps; (2) initialize the weights of the discriminator with the weights of the generator. Then train the discriminator with a discriminative task for n_2 steps, keeping G frozen. Here the discriminative task indicates justifying whether the input token has been replaced by G or not. The generator is thrown after pre-training, and only the discriminator will be fine-tuned on downstream tasks.

RTD is also an alternative solution for the mismatch problem. The `[MASK]` token is used during pre-training but is absent at fine-tuning time.

3) n is drawn from a Gaussian distribution $\mathcal{N}(5, 1)$ clipped at 1 (minimum length) and 10 (maximum length).

Similarly, WKLM [56] replaces words on the entity-level instead of token-level. Concretely, WKLM replaces entity mentions with names of other entities of the same type and train the models to distinguish whether the entity has been replaced.

Next sentence prediction (NSP) Punctuations are the natural separators of text data. So, it is reasonable to construct pre-training methods by utilizing them. Next sentence prediction (NSP) [16] is just a great example of this. As its name suggests, NSP trains the model to distinguish whether two input sentences are continuous segments from the training corpus. Specifically, when choosing the sentences pair for each pre-training example, 50% of the time, the second sentence is the actual next sentence of the first one, and 50% of the time, it is a random sentence from the corpus. By doing so, it is capable to teach the model to understand the relationship between two input sentences and thus benefit downstream tasks that are sensitive to this information, such as question answering and natural language inference.

However, the necessity of the NSP task has been questioned by subsequent work [42, 46, 48, 57]. Ref. [48] found the impact of the NSP task unreliable, while ref. [46] found that single-sentence training without the NSP loss is superior to sentence-pair training with the NSP loss. Moreover, ref. [42] conducted a further analysis for the NSP task, which shows that when training with blocks of text from a single document, removing the NSP loss matches or slightly improves performance on downstream tasks.

Sentence order prediction (SOP) To better model inter-sentence coherence, ALBERT [57] replaces the NSP loss with a sentence order prediction (SOP) loss. As conjectured in ref. [57], NSP conflates topic prediction and coherence prediction in a single task. Thus, the model is allowed to make predictions merely rely on the easier task, topic prediction. Different from NSP, SOP uses two consecutive segments from the same document as positive examples, and the same two consecutive segments but with their order swapped as negative examples. As a result, ALBERT consistently outperforms BERT on various downstream tasks.

StructBERT [47] and BERTje [58] also take SOP as their self-supervised learning task.

3.1.6 Others

Apart from the above tasks, there are many other auxiliary pre-training tasks designated to incorporate factual knowledge (see Sect. 4.1), improve cross-lingual tasks (see Sect. 4.2), multi-modal applications (see Sect. 4.3), or other specific tasks (see Sect. 4.4).

3.2 Taxonomy of PTMs

To clarify the relations of existing PTMs for NLP, we build the taxonomy of PTMs, which categorizes existing PTMs from four different perspectives.

(1) Representation type. According to the representation used for downstream tasks, we can divide PTMs into non-contextual and contextual models.

(2) Architectures. The backbone network used by PTMs, including LSTM, Transformer encoder, Transformer decoder, and the full Transformer architecture. "Transformer" means the standard encoder-decoder architecture. "Transformer encoder" and "Transformer decoder" mean the encoder and decoder part of the standard Transformer architecture, respectively. Their difference is that the decoder part uses masked self-attention with a triangular matrix to prevent tokens from attending their future (right) positions.

(3) Pre-training task types. The type of pre-training tasks used by PTMs. We have discussed them in Sect. 3.1.

(4) Extensions. PTMs designed for various scenarios, including knowledge-enriched PTMs, multilingual or language-specific PTMs, multi-model PTMs, domain-specific PTMs and compressed PTMs. We will particularly introduce these extensions in Sect. 4.

Figure 3 [5, 11–16, 34, 40–43, 45, 47–49, 55–94] shows the taxonomy as well as some corresponding representative PTMs. Besides, Table 2 distinguishes some representative PTMs in more detail.

3.3 Model analysis

Due to the great success of PTMs, it is important to understand what kinds of knowledge are captured by them, and how to induce knowledge from them. There is a wide range of literature analyzing linguistic knowledge and world knowledge stored in pre-trained non-contextual and contextual embeddings.

3.3.1 Non-contextual embeddings

Static word embeddings are first probed for kinds of knowledge. Ref. [95] found that word representations learned by neural network language models are able to capture linguistic regularities in language, and the relationship between words can be characterized by a relation-specific vector offset. Further analogy experiments [11] demonstrated that word vectors produced by skip-gram model can capture both syntactic and semantic word relationships, such as vec("China") − vec("Beijing") ≈ vec("Japan") − vec("Tokyo"). Besides, they find compositionality property of word vectors, for example, vec("Germany") + vec("capital") is close to vec("Berlin"). Inspired by these work, ref. [96] found that

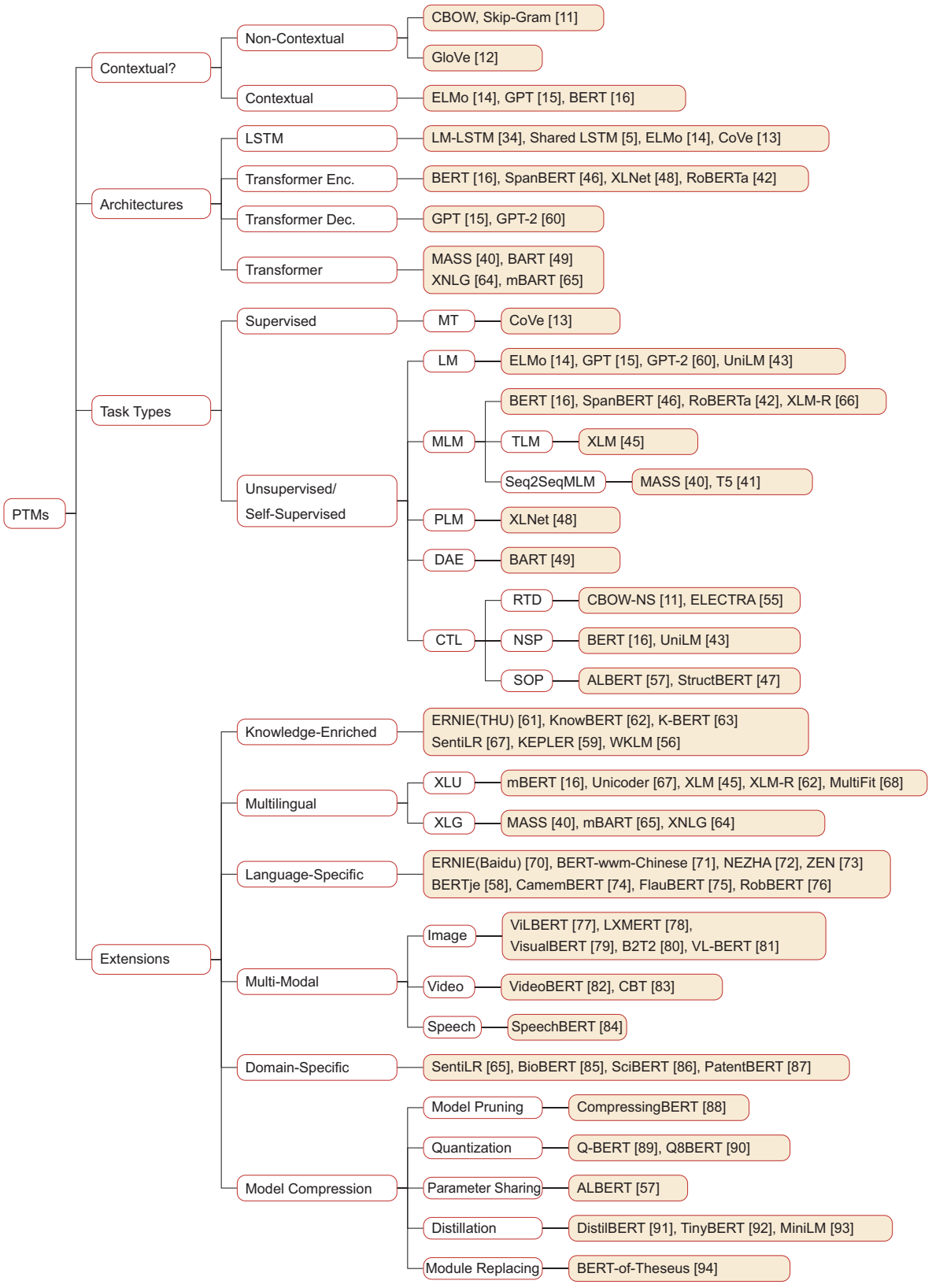

Figure 3 Taxonomy of PTMs with representative examples.

distributional word representations are good at predicting taxonomic properties (e.g., dog is an animal) but fail to learn attributive properties (e.g., swan is white). Similarly, ref. [97] showed that word2vec embeddings implicitly encode referen-

Table 2 List of representative PTMs

PTMs	Architecture[a)]	Input	Pre-training task	Corpus	Params	GLUE[b)]	FT?[e)]
ELMo [14]	LSTM	Text	BiLM	WikiText-103			No
GPT [15]	Transformer Dec.	Text	LM	BookCorpus	117M	72.8	Yes
GPT-2 [60]	Transformer Dec.	Text	LM	WebText	117M–1542M		No
BERT [16]	Transformer Enc.	Text	MLM & NSP	WikiEn+BookCorpus	110M–340M	81.9[c)]	Yes
InfoWord [54]	Transformer Enc.	Text	DIM+MLM	WikiEn+BookCorpus	=BERT	81.1[c)]	Yes
RoBERTa [42]	Transformer Enc.	Text	MLM	BookCorpus+CC-News+OpenWebText+ STORIES	355M	88.5	Yes
XLNet [48]	Two-Stream Transformer Enc.	Text	PLM	WikiEn+ BookCorpus+Giga5 +ClueWeb+Common Crawl	≈BERT	90.5[d)]	Yes
ELECTRA [55]	Transformer Enc.	Text	RTD+MLM	same to XLNet	335M	88.6	Yes
UniLM [43]	Transformer Enc.	Text	MLM[f)]+ NSP	WikiEn+BookCorpus	340M	80.8	Yes
MASS [40]	Transformer	Text	Seq2Seq MLM	*Task-dependent			Yes
BART [49]	Transformer	Text	DAE	same to RoBERTa	110% of BERT	88.4[c)]	Yes
T5 [41]	Transformer	Text	Seq2Seq MLM	Colossal Clean Crawled Corpus (C4)	220M–11B	89.7[c)]	Yes
ERNIE(THU) [61]	Transformer Enc.	Text+Entities	MLM+NSP+dEA	WikiEn + Wikidata	114M	79.6	Yes
KnowBERT [62]	Transformer Enc.	Text	MLM+NSP+EL	WikiEn + WordNet/Wiki	253M–523M		Yes
K-BERT [63]	Transformer Enc.	Text+Triples	MLM+NSP	WikiZh + WebtextZh + CN-DBpedia + HowNet + MedicalKG	=BERT		Yes
KEPLER [59]	Transformer Enc.	Text	MLM+KE	WikiEn + Wikidata/WordNet			Yes
WKLM [56]	Transformer Enc.	Text	MLM+ERD	WikiEn + Wikidata	=BERT		Yes

a) "Transformer Enc." and "Transformer Dec." mean the encoder and decoder part of the standard Transformer architecture respectively. Their difference is that the decoder part uses masked self-attention with triangular matrix to prevent tokens from attending their future (right) positions. "Transformer" means the standard encoder-decoder architecture.

b) The averaged score on 9 tasks of GLUE benchmark (see Sect. 7.1).

c) Without WNLI task.

d) Indicates ensemble result.

e) Means whether is model usually used in fine-tuning fashion.

f) The MLM of UniLM is built on three versions of LMs: Unidirectional LM, Bidirectional LM, and Sequence-to-Sequence LM.

tial attributes of entities. The distributed word vectors, along with a simple supervised model, can learn to predict numeric and binary attributes of entities with a reasonable degree of accuracy.

3.3.2 Contextual embeddings

A large number of studies have probed and induced different types of knowledge in contextual embeddings. In general, there are two types of knowledge: linguistic knowledge and world knowledge.

Linguistic knowledge A wide range of probing tasks are designed to investigate the linguistic knowledge in PTMs. Refs. [98, 99] found that BERT performs well on many syntactic tasks such as part-of-speech tagging and constituent labeling. However, BERT is not good enough at semantic and fine-grained syntactic tasks, compared with simple syntactic tasks.

Besides, ref. [100] analyzed the roles of BERT's layers in different tasks and found that BERT solves tasks in a similar order to that in NLP pipelines. Furthermore, knowledge of subject-verb agreement [101] and semantic roles [102] are also confirmed to exist in BERT. Besides, refs. [103–105] proposed several methods to extract dependency trees and constituency trees from BERT, which proved the BERT's ability to encode syntax structure. Ref. [106] explored the geometry of internal representations in BERT and find some evidence: (1) linguistic features seem to be represented in separate semantic and syntactic subspaces; (2) attention matrices contain grammatical representations; (3) BERT distinguishes word senses at a very fine level.

World knowledge Besides linguistic knowledge, PTMs may also store world knowledge presented in the training data. A straightforward method of probing world knowledge is to query BERT with "fill-in-the-blank" cloze statements, for example, "Dante was born in `[MASK]`". Ref. [107] constructed LAMA (Language Model Analysis) task by manually creating single-token cloze statements (queries) from several knowledge sources. Their experiments show that BERT contains world knowledge competitive with traditional information extraction methods. Since the simplicity of query generation procedure in LAMA, ref. [108] argued that LAMA just measures a lower bound for what lan-

guage models know and propose more advanced methods to generate more efficient queries. Despite the surprising findings of LAMA, it has also been questioned by subsequent work [109, 110]. Similarly, several studies induce relational knowledge [111] and commonsense knowledge [112] from BERT for downstream tasks.

4 Extensions of PTMs

4.1 Knowledge-enriched PTMs

PTMs usually learn universal language representation from general-purpose large-scale text corpora but lack domain-specific knowledge. Incorporating domain knowledge from external knowledge bases into PTM has been shown to be effective. The external knowledge ranges from linguistic [62, 67, 113, 114], semantic [115], commonsense [116], factual [56, 59, 61–63], to domain-specific knowledge [63, 117].

On the one hand, external knowledge can be injected during pre-training. Early studies [118–121] focused on learning knowledge graph embeddings and word embedding jointly. Since BERT, some auxiliary pre-training tasks are designed to incorporate external knowledge into deep PTMs. LIBERT [113] (linguistically-informed BERT) incorporates linguistic knowledge via an additional linguistic constraint task. Ref. [67] integrated sentiment polarity of each word to extend the MLM to Label-Aware MLM (LA-MLM). As a result, their proposed model, SentiLR, achieves state-of-the-art performance on several sentence- and aspect-level sentiment classification tasks. Ref. [115] proposed SenseBERT, which is pre-trained to predict not only the masked tokens but also their supersenses in WordNet. ERNIE(THU) [61] integrates entity embeddings pre-trained on a knowledge graph with corresponding entity mentions in the text to enhance the text representation. Similarly, KnowBERT [62] trains BERT jointly with an entity linking model to incorporate entity representation in an end-to-end fashion. Ref. [59] proposed KEPLER, which jointly optimizes knowledge embedding and language modeling objectives. These work inject structure information of knowledge graph via entity embedding. In contrast, K-BERT [63] explicitly injects related triples extracted from KG into the sentence to obtain an extended tree-form input for BERT. Moreover, ref. [56] adopted entity replacement identification to encourage the model to be more aware of factual knowledge. However, most of these methods update the parameters of PTMs when injecting knowledge, which may suffer from catastrophic forgetting when injecting multiple kinds of knowledge. To address this, K-Adapter [114] injects multiple kinds of knowledge by training different adapters independently for different pre-training tasks, which allows continual knowledge infusion.

On the other hand, one can incorporate external knowledge into pre-trained models without retraining them from scratch. As an example, K-BERT [63] allows injecting factual knowledge during fine-tuning on downstream tasks. Ref. [116] employed commonsense knowledge bases, ConceptNet and ATOMIC, to enhance GPT-2 for story generation. Ref. [122] proposed a knowledge-text fusion model to acquire related linguistic and factual knowledge for machine reading comprehension.

Besides, refs. [123, 124] extended language model to-knowledge graph language model (KGLM) andlatent relation language model (LRLM) respectively, both of which allow prediction conditioned on knowledge graph. These novel KG-conditioned language models show potential for pre-training.

4.2 Multilingual and language-specific PTMs

4.2.1 Multilingual PTMs

Learning multilingual text representations shared across languages plays an important role in many cross-lingual NLP tasks.

Cross-lingual language understanding (XLU) Most of the early work focus on learning multilingual word embedding [125–127], which represents text from multiple languages in a single semantic space. However, these methods usually need (weak) alignment between languages.

Multilingual BERT[4)] (mBERT) is pre-trained by MLM with the shared vocabulary and weights on Wikipedia text from the top 104 languages. Each training sample is a monolingual document, and there are no cross-lingual objectives specifically designed nor any cross-lingual data. Even so, mBERT performs cross-lingual generalization surprisingly well [128]. Ref. [129] showed that the lexical overlap between languages plays a negligible role in cross-lingual success.

XLM [45] improves mBERT by incorporating a cross-lingual task, translation language modeling (TLM), which performs MLM on a concatenation of parallel bilingual sentence pairs. Unicoder [68] further propose three new cross-lingual pre-training tasks, including cross-lingual word recovery, cross-lingual paraphrase classification and cross-lingual masked language model (XMLM).

XLM-RoBERTa (XLM-R) [66] is a scaled multilingual encoder pre-trained on a significantly increased amount of training data, 2.5TB clean CommonCrawl data in 100 different languages. The pre-training task of XLM-RoBERTa

4) https://github.com/google-research/bert/blob/master/multilingual.md

is monolingual MLM only. XLM-R achieves state-of-the-arts results on multiple cross-lingual benchmarks, including XNLI, MLQA, and NER.

Cross-lingual language generation (XLG) Multilingual generation is a kind of tasks to generate text with different languages from the input language, such as machine translation and cross-lingual abstractive summarization.

Different from the PTMs for multilingual classification, the PTMs for multilingual generation usually needs to pre-train both the encoder and decoder jointly, rather than only focusing on the encoder.

MASS [40] pre-trains a Seq2Seq model with monolingual Seq2Seq MLM on multiple languages and achieves significant improvement for unsupervised NMT. XNLG [64] performs two-stage pre-training for cross-lingual natural language generation. The first stage pre-trains the encoder with monolingual MLM and Cross-Lingual MLM (XMLM) tasks. The second stage pre-trains the decoder by using monolingual DAE and Cross-Lingual Auto-Encoding (XAE) tasks while keeping the encoder fixed. Experiments show the benefit of XNLG on cross-lingual question generation and cross-lingual abstractive summarization. mBART [65], a multilingual extension of BART [49], pre-trains the encoder and decoder jointly with Seq2Seq denoising auto-encoder (DAE) task on large-scale monolingual corpora across 25 languages. Experiments demonstrate that mBART produces significant performance gains across a wide variety of machine translation (MT) tasks.

4.2.2 *Language-specific PTMs*

Although multilingual PTMs perform well on many languages, recent work showed that PTMs trained on a single language significantly outperform the multilingual results [74, 75, 130].

For Chinese, which does not have explicit word boundaries, modeling larger granularity [71–73] and multi-granularity [70, 131] word representations have shown great success. Ref. [132] used transfer learning techniques to adapt a multilingual PTM to a monolingual PTM for Russian language. In addition, some monolingual PTMs have been released for different languages, such as CamemBERT [74] and FlauBERT [75] for French, FinBERT [130] for Finnish, BERTje [58] and RobBERT [76] for Dutch, AraBERT [133] for Arabic language.

4.3 Multi-modal PTMs

Observing the success of PTMs across many NLP tasks, some research has focused on obtaining a cross-modal version of PTMs. A great majority of these models are designed for a general visual and linguistic feature encoding. And these models are pre-trained on some huge corpus of cross-modal data, such as videos with spoken words or images with captions, incorporating extended pre-training tasks to fully utilize the multi-modal feature. Typically, tasks like visual-based MLM, masked visual-feature modeling and visual-linguistic matching, are widely used in multi-modal pre-training, such as VideoBERT [82], VisualBERT [79], ViLBERT [77].

4.3.1 *Video-text PTMs*

VideoBERT [82] and CBT [83] are joint video and text models. To obtain sequences of visual and linguistic tokens used for pre-training, the videos are pre-processed by CNN-based encoders and off-the-shelf speech recognition techniques, respectively. And a single Transformer encoder is trained on the processed data to learn the vision-language representations for downstream tasks like video caption. Furthermore, UniViLM [134] proposes to bring in generation tasks to further pre-train the decoder using in downstream tasks.

4.3.2 *Image-text PTMs*

Besides methods for video-language pre-training, several work introduce PTMs on image-text pairs, aiming to fit downstream tasks like visual question answering(VQA) and visual commonsense reasoning(VCR). Several proposed models adopt two separate encoders for image and text representation independently, such as ViLBERT [77] and LXMERT [78]. While other methods like VisualBERT [79], B2T2 [80], VL-BERT [81], Unicoder-VL [135] and UNITER [136] propose single-stream unified Transformer. Though these model architectures are different, similar pre-training tasks, such as MLM and image-text matching, are introduced in these approaches. And to better exploit visual elements, images are converted into sequences of regions by applying RoI or bounding box retrieval techniques before encoded by pre-trained Transformers.

4.3.3 *Audio-text PTMs*

Moreover, several methods have explored the chance of PTMs on audio-text pairs, such as SpeechBERT [84]. This work tries to build an end-to-end Speech Question Answering (SQA) model by encoding audio and text with a single Transformer encoder, which is pre-trained with MLM on speech and text corpus and fine-tuned on question answering.

4.4 Domain-specific and task-specific PTMs

Most publicly available PTMs are trained on general domain corpora such as Wikipedia, which limits their applications to specific domains or tasks. Recently, some studies have proposed PTMs trained on specialty corpora, such as BioBERT [85] for biomedical text, SciBERT [86] for scientific text, ClinicalBERT [137, 138] for clinical text.

In addition to pre-training a domain-specific PTM, some work attempts to adapt available pre-trained models to target applications, such as biomedical entity normalization [139], patent classification [87], progress notes classification and keyword extraction [140].

Some task-oriented pre-training tasks were also proposed, such as sentimentLabel-Aware MLM in SentiLR [67] for sentiment analysis, Gap Sentence Generation (GSG) [141] for text summarization, and Noisy Words Detection for disfluency detection [142].

4.5 Model compression

Since PTMs usually consist of at least hundreds of millions of parameters, they are difficult to be deployed on the on-line service in real-life applications and on resource-restricted devices. Model compression [143] is a potential approach to reduce the model size and increase computation efficiency.

There are five ways to compress PTMs [144]: (1) model pruning, which removes less important parameters; (2) weight quantization [145], which uses fewer bits to represent the parameters; (3) parameter sharing across similar model units; (4) knowledge distillation [146], which trains a smaller student model that learns from intermediate outputs from the original model; (5) module replacing, which replaces the modules of original PTMs with more compact substitutes.

Table 3 gives a comparison of some representative compressed PTMs.

4.5.1 *Model pruning*

Model pruning refers to removing part of neural network (e.g., weights, neurons, layers, channels, attention heads), thereby achieving the effects of reducing the model size and speeding up inference time.

Ref. [88] explored the timing of pruning (e.g., pruning during pre-training, after downstream fine-tuning) and the pruning regimes. Refs. [152, 153] tried to prune the entire self-attention heads in the transformer block.

4.5.2 *Quantization*

Quantization refers to the compression of higher precision parameters to lower precision. Work from refs. [89,90] solely focus on this area. Note that quantization often requires compatible hardware.

4.5.3 *Parameter sharing*

Another well-known approach to reduce the number of parameters is parameter sharing, which is widely used in CNNs, RNNs, and Transformer [154]. ALBERT [57] usescross-layer parameter sharing andfactorized embedding parameterization to reduce the parameters of PTMs. Although the number of parameters is greatly reduced, the training and inference time of ALBERT are even longer than the standard BERT.

Generally, parameter sharing does not improve the computational efficiency at inference phase.

4.5.4 *Knowledge distillation*

Knowledge distillation (KD) [146] is a compression technique in which a small model calledstudent model is trained to reproduce the behaviors of a large model calledteacher model. Here the teacher model can be an ensemble of many models and usually well pre-trained. Different to model compression, distillation techniques learn a small student model from a fixed teacher model through some optimization objectives, while compression techniques aiming at searching a sparser architecture.

Generally, distillation mechanisms can be divided into three types: (1) distillation from soft target probabilities, (2) distillation from other knowledge, and (3) distillation to other structures.

(1) Distillation from soft target probabilities. Ref. [143] showed that making the student approximate the teacher model can transfer knowledge from teacher to student. A common method is approximating the logits of the teacher model. DistilBERT [91] trained the student model with a distillation loss over the soft target probabilities of the teacher as

$$\mathcal{L}_{\text{KD-CE}} = \sum_i t_i * \log(s_i), \tag{5}$$

where t_i and s_i are the probabilities estimated by the teacher model and the student, respectively.

Distillation from soft target probabilities can also be used in task-specific models, such as information retrieval [155], and sequence labeling [156].

(2) Distillation from other knowledge. Distillation from soft target probabilities regards the teacher model as a black box and only focus on its outputs. Moreover, decomposing the teacher model and distilling more knowledge can bring improvement to the student model.

Table 3 Comparison of compressed PTMs[a)]

Method	Type	#Layer	Loss function[c)]	Speed Up	Params	Source PTM	GLUE[b)]
$\text{BERT}_{\text{BASE}}$ [16]	Baseline	12	$\mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{NSP}}$		110M		79.6
$\text{BERT}_{\text{LARGE}}$ [16]		24	$\mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{NSP}}$		340M		81.9
Q-BERT [89]	Quantization	12	HAWQ + GWQ	-		$\text{BERT}_{\text{BASE}}$	≈ 99% BERT[g)]
Q8BERT [90]		12	DQ + QAT	-		$\text{BERT}_{\text{BASE}}$	≈ 99% BERT
ALBERT [57]	Param. Sharing	12	$\mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{SOP}}$	×5.6 ~ 0.3	12 ~ 235M		89.4 (ensemble)
DistilBERT [91]	Distillation	6	$\mathcal{L}_{\text{KD-CE}}$+$\text{Cos}_{\text{KD}}$+ $\mathcal{L}_{\text{MLM}}$	×1.63	66M	$\text{BERT}_{\text{BASE}}$	77.0 (dev)
TinyBERT[d),e)] [92]		4	$\text{MSE}_{\text{embed}}$+$\text{MSE}_{\text{attn}}$+ MSE_{hidn}+$\mathcal{L}_{\text{KD-CE}}$	×9.4	14.5M	$\text{BERT}_{\text{BASE}}$	76.5
BERT-PKD [147]		3–6	$\mathcal{L}_{\text{KD-CE}}$+$\text{PT}_{\text{KD}}$+ $\mathcal{L}_{\text{Task}}$	×3.73 ~ 1.64	45.7 ~ 67 M	$\text{BERT}_{\text{BASE}}$	76.0 ~ 80.6[h)]
PD [148]		6	$\mathcal{L}_{\text{KD-CE}}$+$\mathcal{L}_{\text{Task}}$+ $\mathcal{L}_{\text{MLM}}$	×2.0	67.5M	$\text{BERT}_{\text{BASE}}$	81.2[h)]
MobileBERT[d)] [149]		24	FMT+AT+PKT+ $\mathcal{L}_{\text{KD-CE}}$+$\mathcal{L}_{\text{MLM}}$	×4.0	25.3M	$\text{BERT}_{\text{LARGE}}$	79.7
MiniLM [93]		6	AT+AR	×1.99	66M	$\text{BERT}_{\text{BASE}}$	81.0[f)]
DualTrain[d),e)] [150]		12	Dual Projection+$\mathcal{L}_{\text{MLM}}$	-	1.8 ~ 19.2M	$\text{BERT}_{\text{BASE}}$	75.8 ~ 81.9[i)]
BERT-of-Theseus [94]	Module Replacing	6	$\mathcal{L}_{\text{Task}}$	×1.94	66M	$\text{BERT}_{\text{BASE}}$	78.6

a) The desing of this table is borrowed from [94, 151].

b) The averaged score on 8 tasks (without WNLI) of GLUE benchmark (see Sect. 7.1). Here MNLI-m and MNLI-mm are regarded as two different tasks. "dev" indicates the result is on dev set. "ensemble" indicates the result is from the ensemble model.

c) "$\mathcal{L}_{\text{MLM}}$ ", "$\mathcal{L}_{\text{NSP}}$", and "$\mathcal{L}_{\text{SOP}}$" indicate pre-training objective (see Sect. 3.1 and Table 1)."$\mathcal{L}_{\text{Task}}$" means task-specific loss.

"HAWQ", "GWQ", "DQ", and "QAT" indicate Hessian AWare Quantization, Group-wise Quantization, Quantization-Aware Training, and Dynamically Quantized, respectively. "KD" means knowledge distillation. "FMT", "AT", and 'PKT" mean Feature Map Transfer, Attention Transfer, and Progressive Knowledge Transfer, respectively. "AR" means Self-Attention value relation.

d) The dimensionality of the hidden or embedding layers is reduced.

e) Use a smaller vocabulary.

f) Generally, the F1 score is usually used as the main metric of the QQP task. But MiniLM reports the accuracy, which is incomparable to other work.

g) Result on MNLI and SST-2 only.

h) Result on the other tasks except for STS-B and CoLA.

i) Result on MRPC, MNLI, and SST-2 only.

TinyBERT [92] performs layer-to-layer distillation with embedding outputs, hidden states, and self-attention distributions. MobileBERT [149] also perform layer-to-layer distillation with soft target probabilities, hidden states, and self-attention distributions. MiniLM [93] distill self-attention distributions and self-attention value relation from teacher model.

Besides, other models distill knowledge through many approaches. Ref. [147] introduced a "patient" teacher-student mechanism, ref. [157] exploited KD to improve a pre-trained multi-task deep neural network.

(3) Distillation to other structures. Generally, the structure of the student model is the same as the teacher model, except for a smaller layer size and a smaller hidden size. However, not only decreasing parameters but also simplifying model structures from Transformer to RNN [158] or CNN [159] can reduce the computational complexity.

4.5.5 Module replacing

Module replacing is an interesting and simple way to reduce the model size, which replaces the large modules of original PTMs with more compact substitutes. Ref. [94] proposed Theseus Compression motivated by a famous thought experiment called "Ship of Theseus", which progressively substitutes modules from the source model with modules of fewer parameters. Different from KD, Theseus Compression only requires one task-specific loss function. The compressed model, BERT-of-Theseus, is 1.94× faster while retaining more than 98% performance of the source model.

4.5.6 Others

In addition to reducing model sizes, there are other ways to improve the computational efficiency of PTMs in practical scenarios with limited resources. Ref. [160] proposed a practical speed-tunable BERT, namely FastBERT, which can dynamically reduce computational steps with sample-wise adaptive mechanism.

5 Adapting PTMs to downstream tasks

Although PTMs capture the general language knowledge from a large corpus, how effectively adapting their knowledge to the downstream task is still a key problem.

5.1 Transfer learning

Transfer learning [161] is to adapt the knowledge from a source task (or domain) to a target task (or domain). Figure 4 gives an illustration of transfer learning.

There are many types of transfer learning in NLP, such as domain adaptation, cross-lingual learning, multi-task learning. Adapting PTMs to downstream tasks issequential transfer learning task, in which tasks are learned sequentially and the target task has labeled data.

5.2 How to transfer?

To transfer the knowledge of a PTM to the downstream NLP tasks, we need to consider the following issues.

5.2.1 Choosing appropriate pre-training task, model architecture and corpus

Different PTMs usually have different effects on the same downstream task, since these PTMs are trained with various pre-training tasks, model architecture, and corpora.

(1) Currently, the language model is the most popular pre-training task and can more efficiently solve a wide range of NLP problems [60]. However, different pre-training tasks have their own bias and give different effects for different tasks. For example, the NSP task [16] makes PTM understand the relationship between two sentences. Thus, the PTM can benefit downstream tasks such as question answering (QA) and natural language inference (NLI).

(2) The architecture of PTM is also important for the downstream task. For example, although BERT helps with most natural language understanding tasks, it is hard to generate language.

(3) The data distribution of the downstream task should be approximate to PTMs. Currently, there are a large number of off-the-shelf PTMs, which can just as conveniently be used for various domain-specific or language-specific downstream tasks.

Therefore, given a target task, it is always a good solution to choose the PTMs trained with appropriate pre-training task, architecture, and corpus.

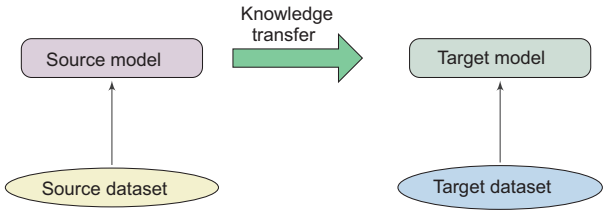

Figure 4 (Color online) Transfer learning.

5.2.2 Choosing appropriate layers

Given a pre-trained deep model, different layers should capture different kinds of information, such as POS tagging, parsing, long-term dependencies, semantic roles, coreference. For RNN-based models, refs. [33, 162] showed that representations learned from different layers in a multi-layer LSTM encoder benefit different tasks (e.g., predicting POS tags and understanding word sense). For transformer-based PTMs, ref. [100] found BERT represents the steps of the traditional NLP pipeline: basic syntactic information appears earlier in the network, while high-level semantic information appears at higher layers.

Let $\mathbf{H}^{(l)}(1 \le l \le L)$ denotes the l-th layer representation of the pre-trained model with L layers, and $g(\cdot)$ denote the task-specific model for the target task.

There are three ways to select the representation.

(1) Embedding only. One approach is to choose only the pre-trained static embeddings, while the rest of the model still needs to be trained from scratch for a new target task.

They fail to capture higher-level information that might be even more useful. Word embeddings are only useful in capturing semantic meanings of words, but we also need to understand higher-level concepts like word sense.

(2) Top layer. The most simple and effective way is to feed the representation at the top layer into the task-specific model $g(\mathbf{H}^{(L)})$.

(3) All layers. A more flexible way is to automatic choose the best layer in a soft version, like ELMo [14]:

$$\mathbf{r}_t = \gamma \sum_{l=1}^{L} \alpha_l \mathbf{h}_t^{(l)}, \tag{6}$$

where α_l is the softmax-normalized weight for layer l and γ is a scalar to scale the vectors output by pre-trained model. The mixup representation is fed into the task-specific model $g(\mathbf{r}_t)$.

5.2.3 To tune or not to tune?

Currently, there are two common ways of model transfer: feature extraction (where the pre-trained parameters are frozen), and fine-tuning (where the pre-trained parameters are unfrozen and fine-tuned).

In feature extraction way, the pre-trained models are regarded as off-the-shelf feature extractors. Moreover, it is important to expose the internal layers as they typically encode the most transferable representations [163].

Although both these two ways can significantly benefit most of NLP tasks, feature extraction way requires more complex task-specific architecture. Therefore, the fine-tuning

way is usually more general and convenient for many different downstream tasks than feature extraction way.

Table 4 gives some common combinations of adapting PTMs.

5.3 Fine-tuning strategies

With the increase of the depth of PTMs, the representation captured by them makes the downstream task easier. Therefore, the task-specific layer of the whole model is simple. Since ULMFit and BERT, fine-tuning has become the main adaption method of PTMs. However, the process of fine-tuning is often brittle: even with the same hyper-parameter values, distinct random seeds can lead to substantially different results [166].

Besides standard fine-tuning, there are also some useful fine-tuning strategies.

Two-stage fine-tuning An alternative solution is two-stage transfer, which introduces an intermediate stage between pre-training and fine-tuning. In the first stage, the PTM is transferred into a model fine-tuned by an intermediate task or corpus. In the second stage, the transferred model is fine-tuned to the target task. Ref. [167] showed that the "further pre-training" on the related-domain corpus can further improve the ability of BERT and achieved state-of-the-art performance on eight widely-studied text classification datasets. Refs. [168, 169] introduced the intermediate supervised task related to the target task, which brings a large improvement for BERT, GPT, and ELMo. Ref. [170] also used a two-stage transfer for the story ending prediction. The proposed TransBERT (transferable BERT) can transfer not only general language knowledge from large-scale unlabeled data but also specific kinds of knowledge from various semantically related supervised tasks.

Multi-task fine-tuning Ref. [171] fine-tuned BERT under the multi-task learning framework, which demonstrates that multi-task learning and pre-training are complementary technologies.

Fine-tuning with extra adaptation modules The main drawback of fine-tuning is its parameter inefficiency: every downstream task has its own fine-tuned parameters. Therefore, a better solution is to inject some fine-tunable adaptation modules into PTMs while the original parameters are fixed.

Ref. [172] equipped a single share BERT model with small additional task-specific adaptation modules, projected attention layers (PALs). The shared BERT with the PALs matches separately fine-tuned models on the GLUE benchmark with roughly 7 times fewer parameters. Similarly, ref. [173] modified the architecture of pre-trained BERT by adding adapter modules. Adapter modules yield a compact and extensible model; they add only a few trainable parameters per task, and new tasks can be added without revisiting previous ones. The parameters of the original network remain fixed, yielding a high degree of parameter sharing.

Others Motivated by the success of widely-used ensemble models, ref. [174] improved the fine-tuning of BERT with two effective mechanisms: self-ensemble and self-distillation, which can improve the performance of BERT on downstream tasks without leveraging external resource or significantly decreasing the training efficiency. They integrated ensemble and distillation within a single training process. The teacher model is an ensemble model by parameter-averaging several student models in previous time steps.

Instead of fine-tuning all the layers simultaneously, gradual unfreezing [37] is also an effective method that gradually unfreezes layers of PTMs starting from the top layer. Ref. [175] proposed a simpler unfreezing method, sequential unfreezing, which first fine-tunes only the randomly-initialized task-specific layers, and then unfreezes the hidden layers of PTM, and finally unfreezes the embedding layer.

Ref. [176] compressed ELMo embeddings using variational information bottleneck while keeping only the information that helps the target task.

Generally, the above work shows that the utility of PTMs can be further stimulated by better fine-tuning strategies.

Table 4 Some common combinations of adapting PTMs

Where	FT/FE?[a)]	PTMs
Embedding Only	FT/FE	Word2vec [11],GloVe [12]
Top Layer	FT	BERT [16],RoBERTa [42]
Top Layer	FE	BERT[b)] [164, 165]
All Layers	FE	ELMo [14]

a) FT and FE mean Fine-tuning and Feature Extraction respectively.
b) BERT used as feature extractor.

6 Resources of PTMs

There are many related resources for PTMs available online. Table 5 provides some popular repositories, including third-party implementations, paper lists, visualization tools, and other related resources of PTMs.

Besides, there are some other good survey papers on PTMs for NLP [151, 182, 183].

7 Applications

In this section, we summarize some applications of PTMs in several classic NLP tasks.

Table 5 Resources of PTMs

Resource	Description	URL
	Open-Source Implementations[a)]	
word2vec	CBOW,Skip-Gram	https://github.com/tmikolov/word2vec
GloVe	Pre-trained word vectors	https://nlp.stanford.edu/projects/glove
FastText	Pre-trained word vectors	https://github.com/facebookresearch/fastText
Transformers	Framework: PyTorch&TF, PTMs: BERT, GPT-2, RoBERTa, XLNet, etc.	https://github.com/huggingface/transformers
Fairseq	Framework: PyTorch, PTMs:English LM, German LM, RoBERTa, etc.	https://github.com/pytorch/fairseq
Flair	Framework: PyTorch, PTMs:BERT, ELMo, GPT, RoBERTa, XLNet, etc.	https://github.com/flairNLP/flair
AllenNLP [177]	Framework: PyTorch, PTMs: ELMo, BERT, GPT-2, etc.	https://github.com/allenai/allennlp
fastNLP	Framework: PyTorch, PTMs: RoBERTa, GPT, etc.	https://github.com/fastnlp/fastNLP
UniLMs	Framework: PyTorch, PTMs: UniLM v1&v2, MiniLM, LayoutLM, etc.	https://github.com/microsoft/unilm
Chinese-BERT [71]	Framework: PyTorch&TF, PTMs: BERT, RoBERTa, etc. (for Chinese)	https://github.com/ymcui/Chinese-BERT-wwm
BERT [16]	Framework: TF, PTMs: BERT, BERT-wwm	https://github.com/google-research/bert
RoBERTa [42]	Framework: PyTorch	https://github.com/pytorch/fairseq/tree/master/examples/roberta
XLNet [48]	Framework: TF	https://github.com/zihangdai/xlnet/
ALBERT [57]	Framework: TF	https://github.com/google-research/ALBERT
T5 [41]	Framework: TF	https://github.com/google-research/text-to-text-transfer-transformer
ERNIE(Baidu) [70, 131]	Framework: PaddlePaddle	https://github.com/PaddlePaddle/ERNIE
CTRL [178]	Conditional transformer language model for controllable generation.	https://github.com/salesforce/ctrl
BertViz [179]	Visualization Tool	https://github.com/jessevig/bertviz
exBERT [180]	Visualization Tool	https://github.com/bhoov/exbert
TextBrewer [181]	PyTorch-based toolkit for distillation of NLP models.	https://github.com/airaria/TextBrewer
DeepPavlov	Conversational AI Library. PTMs for the Russian, Polish, Bulgarian, Czech, and informal English.	https://github.com/deepmipt/DeepPavlov
	Corpora	
OpenWebText	Open clone of OpenAI's unreleased WebText dataset.	https://github.com/jcpeterson/openwebtext
Common Crawl	A very large collection of text.	http://commoncrawl.org/
WikiEn	English Wikipedia dumps.	https://dumps.wikimedia.org/enwiki/
	Other Resources	
Paper List		https://github.com/thunlp/PLMpapers
Paper List		https://github.com/tomohideshibata/BERT-related-papers
Paper List		https://github.com/cedrickchee/awesome-bert-nlp
Bert Lang Street	A collection of BERT models with reported performances on different datasets, tasks and languages.	https://bertlang.unibocconi.it/

a) Most papers for PTMs release their links of official version. Here we list some popular third-party and official implementations.

7.1 General evaluation benchmark

There is an essential issue for the NLP community that how can we evaluate PTMs in a comparable metric. Thus, large-scale-benchmark is necessary.

The general language understanding evaluation (GLUE) benchmark [184] is a collection of nine natural language understanding tasks, including single-sentence classification tasks (CoLA and SST-2), pairwise text classification tasks (MNLI, RTE, WNLI, QQP, and MRPC), text similarity task (STS-B), and relevant ranking task (QNLI). GLUE benchmark is well-designed for evaluating the robustness as well as generalization of models. GLUE does not provide the labels for the test set but set up an evaluation server.

However, motivated by the fact that the progress in recent years has eroded headroom on the GLUE benchmark dramatically, a new benchmark called SuperGLUE [185] was presented. Compared with GLUE, SuperGLUE has more chal-

lenging tasks and more diverse task formats (e.g., coreference resolution and question answering).

State-of-the-art PTMs are listed in the corresponding leaderboard[5),6)].

7.2 Question answering

Question answering (QA), or a narrower concept machine reading comprehension (MRC), is an important application in the NLP community. From easy to hard, there are three types of QA tasks: single-round extractive QA (SQuAD) [186], multi-round generative QA (CoQA) [187], and multi-hop QA (HotpotQA) [188].

BERT creatively transforms the extractive QA task to the spans prediction task that predicts the starting span as well as the ending span of the answer [16]. After that, PTM as an encoder for predicting spans has become a competitive baseline. For extractive QA, ref. [189] proposed a retrospective reader architecture and initialize the encoder with PTM (e.g., ALBERT). For multi-round generative QA, ref. [190] proposed a "PTM+Adversarial Training+Rationale Tagging+Knowledge Distillation" model. For multi-hop QA, ref. [191] proposed an interpretable "Select, Answer, and Explain" (SAE) system that PTM acts as the encoder in the selection module.

Generally, encoder parameters in the proposed QA model are initialized through a PTM, and other parameters are randomly initialized. State-of-the-art models are listed in the corresponding leaderboard[7),8),9)].

7.3 Sentiment analysis

BERT outperforms previous state-of-the-art models by simply fine-tuning on SST-2, which is a widely used dataset for sentiment analysis (SA) [16]. Ref. [192] utilized BERT with transfer learning techniques and achieve new state-of-the-art in Japanese SA.

Despite their success in simple sentiment classification, directly applying BERT to aspect-based sentiment analysis (ABSA), which is a fine-grained SA task, shows less significant improvement [193]. To better leverage the powerful representation of BERT, ref. [193] constructed an auxiliary sentence by transforming ABSA from a single sentence classification task to a sentence pair classification task. Ref. [194] proposed post-training to adapt BERT from its source domain and tasks to the ABSA domain and tasks. Furthermore, ref. [195] extended the work of ref. [194] by analyzing the behavior of cross-domain post-training with ABSA performance. Ref. [196] showed that the performance of post-trained BERT could be further improved via adversarial training. Ref. [197] added an additional pooling module, which can be implemented as either LSTM or attention mechanism, to leverage BERT intermediate layers for ABSA. In addition, ref. [198] jointly learned aspect detection and sentiment classification towards end-to-end ABSA. SentiLR [67] acquires part-of-speech tag and prior sentiment polarity from SentiWordNet and adoptsLabel-Aware MLM to utilize the introduced linguistic knowledge to capture the relationship between sentence-level sentiment labels and word-level sentiment shifts. SentiLR achieves state-of-the-art performance on several sentence- and aspect-level sentiment classification tasks.

For sentiment transfer, ref. [199] proposed "Mask and Infill" based on BERT. In the mask step, the model disentangles sentiment from content by masking sentiment tokens. In the infill step, it uses BERT along with a target sentiment embedding to infill the masked positions.

7.4 Named entity recognition

Named entity recognition (NER) in information extraction and plays an important role in many NLP downstream tasks. In deep learning, most of NER methods are in the sequence-labeling framework. The entity information in a sentence will be transformed into the sequence of labels, and one label corresponds to one word. The model is used to predict the label of each word. Since ELMo and BERT have shown their power in NLP, there is much work about pre-trained models for NER.

Ref. [36] used a pre-trained character-level language model to produce word-level embedding for NER. TagLM [200] and ELMo [14] use a pre-trained language model's last layer output and weighted-sum of each layer output as a part of word embedding. Ref. [201] used layer-wise pruning and dense connection to speed up ELMo's inference on NER. Ref. [16] used the first BPE's BERT representation to predict each word's label without CRF. Ref. [128] realized zero-shot NER through multilingual BERT. Ref. [156] leveraged knowledge distillation to run a small BERT for NER on a single CPU. Besides, BERT is also used on domain-specific NER, such as biomedicine [85, 202], etc.

5) https://gluebenchmark.com/
6) https://super.gluebenchmark.com/
7) https://rajpurkar.github.io/SQuAD-explorer/
8) https://stanfordnlp.github.io/coqa/
9) https://hotpotqa.github.io/

7.5 Machine translation

Machine translation (MT) is an important task in the NLP community, which has attracted many researchers. Almost all of neural machine translation (NMT) models share the encoder-decoder framework, which first encodes input tokens to hidden representations by the encoder and then decodes output tokens in the target language from the decoder. Ref. [35] found the encoder-decoder models can be significantly improved by initializing both encoder and decoder with pre-trained weights of two language models. Ref. [203] used ELMo to set the word embedding layer in the NMT model. This work shows performance improvements on English-Turkish and English-German NMT model by using a pre-trained language model for source word embedding initialization.

Given the superb performance of BERT on other NLP tasks, it is natural to investigate how to incorporate BERT into NMT models. Ref. [45] tried to initialize the entire encoder and decoder by a multilingual pre-trained BERT model and showed a significant improvement could be achieved on unsupervised MT and English-Romanian supervised MT. Similarly, ref. [204] devised a series of different experiments for examining the best strategy to utilize BERT on the encoder part of NMT models. They achieved some improvement by using BERT as an initialization of the encoder. Also, they found that these models can get better performance on the out-of-domain dataset. Ref. [205] proposed a two stages BERT fine-tuning method for NMT. At the first stage, the encoder is initialized by a pre-trained BERT model, and they only train the decoder on the training set. At the second stage, the whole NMT model is jointly fine-tuned on the training set. By experiment, they show this approach can surpass the one stage fine-tuning method, which directly fine-tunes the whole model. Apart from that, ref. [165] suggested using pre-trained BERT as an extra memory to facilitate NMT models. Concretely, they first encode the input tokens by a pre-trained BERT and use the output of the last layer as extra memory. Then, the NMT model can access the memory via an extra attention module in each layer of both encoder and decoder. And they show a noticeable improvement in supervised, semi-supervised, and unsupervised MT.

Instead of only pre-training the encoder, MASS (Masked Sequence-to-Sequence Pre-Training) [40] utilizes Seq2Seq MLM to pre-train the encoder and decoder jointly. In the experiment, this approach can surpass the BERT-style pre-training proposed by ref. [45] both on unsupervised MT and English-Romanian supervised MT. Different from MASS, mBART [65], a multilingual extension of BART [49], pre-trains the encoder and decoder jointly with Seq2Seq denoising auto-encoder (DAE) task on large-scale monolingual corpora across 25 languages. Experiments demonstrated that mBART could significantly improve both supervised and unsupervised machine translation at both the sentence level and document level.

7.6 Summarization

Summarization, aiming at producing a shorter text which preserves the most meaning of a longer text, has attracted the attention of the NLP community in recent years. The task has been improved significantly since the widespread use of PTM. Ref. [164] introduced transferable knowledge (e.g., BERT) for summarization and surpassed previous models. Ref. [206] tries to pre-trained a document-level model that predicts sentences instead of words, and then apply it on downstream tasks such as summarization. More elaborately, ref. [141] designed a gap sentence generation (GSG) task for pre-training, whose objective involves generating summary-like text from the input. Furthermore, ref. [207] proposed BERTSUM. BERTSUM included a novel document-level encoder, and a general framework for both extractive summarization and abstractive summarization. In the encoder frame, BERTSUM extends BERT by inserting multiple `[CLS]` tokens to learn the sentence representations. For extractive summarization, BERTSUM stacks several inter-sentence Transformer layers. For abstractive summarization, BERTSUM proposes a two-staged fine-tuning approach using a new fine-tuning schedule. Ref. [208] proposed a novel summary-level framework MATCHSUM and conceptualized extractive summarization as a semantic text matching problem. They proposed a Siamese-BERT architecture to compute the similarity between the source document and the candidate summary and achieved a state-of-the-art result on CNN/DailyMail (44.41 in ROUGE-1) by only using the base version of BERT.

7.7 Adversarial attacks and defenses

The deep neural models are vulnerable to adversarial examples that can mislead a model to produce a specific wrong prediction with imperceptible perturbations from the original input. In CV, adversarial attacks and defenses have been widely studied. However, it is still challenging for text due to the discrete nature of languages. Generating of adversarial samples for text needs to possess such qualities: (1) imperceptible to human judges yet misleading to neural models; (2) fluent in grammar and semantically consistent with original inputs. Ref. [209] successfully attacked the fine-tuned BERT on text classification and textual entailment with adversarial examples. Ref. [210] defined universal adversarial triggers that can induce a model to produce a specific-purpose predic-

tion when concatenated to any input. Some triggers can even cause the GPT-2 model to generate racist text. Ref. [211] showed BERT is not robust on misspellings.

PTMs also have great potential to generate adversarial samples. Ref. [212] proposed **BERT-Attack**, a BERT-based high-quality and effective attacker. They turned BERT against another fine-tuned BERT on downstream tasks and successfully misguided the target model to predict incorrectly, outperforming state-of-the-art attack strategies in both success rate and perturb percentage, while the generated adversarial samples are fluent and semantically preserved.

Besides, adversarial defenses for PTMs are also promising, which improve the robustness of PTMs and make them immune against adversarial attack.

Adversarial training aims to improve the generalization by minimizes the maximal risk for label-preserving perturbations in embedding space. Recent work [213, 214] showed that adversarial pre-training or fine-tuning can improve both generalization and robustness of PTMs for NLP.

8 Future directions

Though PTMs have proven their power for various NLP tasks, challenges still exist due to the complexity of language. In this section, we suggest five future directions of PTMs.

(1) Upper bound of PTMs Currently, PTMs have not yet reached its upper bound. Most of the current PTMs can be further improved by more training steps and larger corpora.

The state of the art in NLP can be further advanced by increasing the depth of models, such as Megatron-LM [215] (8.3 billion parameters, 72 Transformer layers with a hidden size of 3072 and 32 attention heads) and Turing-NLG[10] (17 billion parameters, 78 Transformer layers with a hidden size of 4256 and 28 attention heads).

The general-purpose PTMs are always our pursuits for learning the intrinsic universal knowledge of languages (even world knowledge). However, such PTMs usually need deeper architecture, larger corpus, and challenging pre-training tasks, which further result in higher training costs. However, training huge models is also a challenging problem, which needs more sophisticated and efficient training techniques such as distributed training, mixed precision, gradient accumulation, etc. Therefore, a more practical direction is to design more efficient model architecture, self-supervised pre-training tasks, optimizers, and training skills using existing hardware and software. ELECTRA [55] is a good solution towards this direction.

(2) Architecture of PTMs The transformer has been proven to be an effective architecture for pre-training. However, the main limitation of the Transformer is its computation complexity, which is quadratic to the input length. Limited by the memory of GPUs, most of current PTMs cannot deal with the sequence longer than 512 tokens. Breaking this limit needs to improve the architecture of the Transformer, such as Transformer-XL [216]. Therefore, searching for more efficient model architecture for PTMs is important to capture longer-range contextual information.

The design of deep architecture is challenging, and we may seek help from some automatic methods, such as neural architecture search (NAS) [217].

(3) Task-oriented pre-training and model compression In practice, different downstream tasks require the different abilities of PTMs. The discrepancy between PTMs and downstream tasks usually lies in two aspects: model architecture and data distribution. A larger discrepancy may result in that the benefit of PTMs may be insignificant. For example, text generation usually needs a specific task to pre-train both the encoder and decoder, while text matching needs pre-training tasks designed for sentence pairs.

Besides, although larger PTMs can usually lead to better performance, a practical problem is how to leverage these huge PTMs on special scenarios, such as low-capacity devices and low-latency applications. Therefore, we can carefully design the specific model architecture and pre-training tasks for downstream tasks or extract partial task-specific knowledge from existing PTMs.

Instead of training task-oriented PTMs from scratch, we can teach them with existing general-purpose PTMs by using techniques such as model compression (see Sect. 4.5). Although model compression is widely studied for CNNs in CV [218], compression for PTMs for NLP is just beginning. The fully-connected structure of the Transformer also makes model compression more challenging.

(4) Knowledge transfer beyond fine-tuning Currently, fine-tuning is the dominant method to transfer PTMs' knowledge to downstream tasks, but one deficiency is its parameter inefficiency: every downstream task has its own fine-tuned parameters. An improved solution is to fix the original parameters of PTMs and by adding small fine-tunable adaption modules for specific task [172, 173]. Thus, we can use a shared PTM to serve multiple downstream tasks. Indeed, mining knowledge from PTMs can be more flexible, such as feature extraction, knowledge distillation [181], data augmentation [219,220], using PTMs as external knowledge [107]. More efficient methods are expected.

(5) Interpretability and reliability of PTMs Although PTMs reach impressive performance, their deep non-linear

10) https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/

architecture makes the procedure of decision-making highly non-transparent.

Recently, explainable artificial intelligence (XAI) [221] has become a hotspot in the general AI community. Unlike CNNs for images, interpreting PTMs is harder due to the complexities of both the Transformer-like architecture and language. Extensive efforts (see Sect. 3.3) have been made to analyze the linguistic and world knowledge included in PTMs, which help us understand these PMTs with some degree of transparency. However, much work on model analysis depends on the attention mechanism, and the effectiveness of attention for interpretability is still controversial [222, 223].

Besides, PTMs are also vulnerable to adversarial attacks (see Sect. 7.7). The reliability of PTMs is also becoming an issue of great concern with the extensive use of PTMs in production systems. The studies of adversarial attacks against PTMs help us understand their capabilities by fully exposing their vulnerabilities. Adversarial defenses for PTMs are also promising, which improve the robustness of PTMs and make them immune against adversarial attack.

Overall, as key components in many NLP applications, the interpretability and reliability of PTMs remain to be explored further in many respects, which helps us understand how PTMs work and provides a guide for better usage and further improvement.

9 Conclusion

In this survey, we conduct a comprehensive overview of PTMs for NLP, including background knowledge, model architecture, pre-training tasks, various extensions, adaption approaches, related resources, and applications. Based on current PTMs, we propose a new taxonomy of PTMs from four different perspectives. We also suggest several possible future research directions for PTMs.

This work was supported by the National Natural Science Foundation of China (Grant Nos. 61751201 and 61672162), the Shanghai Municipal Science and Technology Major Project (Grant No. 2018SHZDZX01) and ZJLab. We thank Zhiyuan Liu, Wanxiang Che, Minlie Huang, Danqing Wang and Luyao Huang for their valuable feedback on this manuscript.

1 Kalchbrenner N, Grefenstette E, Blunsom P. A convolutional neural network for modelling sentences. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Baltimore, 2014. 655–665
2 Kim Y. Convolutional neural networks for sentence classification. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Doha, 2014. 1746–1751
3 Gehring J, Auli M, Grangier D, et al. Convolutional sequence to sequence learning. In: Proceedings of the International Conference on Machine Learning. Sydney, 2017. 1243–1252
4 Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks. In: Proceedings of the Advances in Neural Information Processing Systems. Montreal, 2014. 3104–3112
5 Liu P, Qiu X, Huang X. Recurrent neural network for text classification with multi-task learning. In: Proceedings of the International Joint Conference on Artificial Intelligence. New York, 2016. 2873–2879
6 Socher R, Perelygin A, Wu J Y, et al. Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Seattle, 2013. 1631–1642
7 Tai K S, Socher R, Manning C D. Improved semantic representations from tree-structured long short-term memory networks. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Beijing, 2015. 1556–1566
8 Marcheggiani D, Bastings J, Titov I. Exploiting semantics in neural machine translation with graph convolutional networks. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. New Orleans, 2018. 486–492
9 Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. In: Proceedings of the International Conference on Learning Representations. San Diego, 2015
10 Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. In: Proceedings of the Advances in Neural Information Processing Systems. Long Beach, 2017. 5998–6008
11 Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality. In: Proceedings of the Advances in Neural Information Processing Systems. Lake Tahoe, 2013. 3111–3119
12 Pennington J, Socher R, Manning C D. GloVe: Global vectors for word representation. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Doha, 2014. 1532–1543
13 McCann B, Bradbury J, Xiong C, et al. Learned in translation: Contextualized word vectors. In: Proceedings of the Advances in Neural Information Processing Systems. Long Beach, 2017. 6294–6305
14 Peters M E, Neumann M, Iyyer M, et al. Deep contextualized word representations. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. New Orleans, 2018. 2227–2237
15 Radford A, Narasimhan K, Salimans T, et al. Improving language understanding by generative pre-training. OpenAI Blog. 2018
16 Devlin J, Chang M, Lee K, et al. BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, 2019. 4171–4186
17 Bengio Y, Courville A, Vincent P. Representation learning: A review and new perspectives. IEEE Trans Pattern Anal Mach Intell, 2013, 35: 1798–1828
18 Kim Y, Jernite Y, Sontag D, et al. Character-aware neural language models. In: Proceedings of the AAAI Conference on Artificial Intelligence. Phoenix, 2016. 2741–2749
19 Bojanowski P, Grave E, Joulin A, et al. Enriching word vectors with subword information. Trans Associat Comput Linguist, 2017, 5: 135–146
20 Sennrich R, Haddow B, Birch A. Neural machine translation of rare words with subword units. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Berlin, 2016
21 Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation. 1997. 9: 1735–1780
22 Chung J, Gulcehre C, Cho K, et al. Empirical evaluation of gated recurrent neural networks on sequence modeling. ArXiv: 1412.3555

23 Zhu X, Sobihani P, Guo H. Long short-term memory over recursive structures. In: Proceedings of the International Conference on Machine Learning. Lille, 2015. 1604–1612
24 Kipf T N and Welling M. Semi-supervised classification with graph convolutional networks. In: Proceedings of the International Conference on Learning Representations. Toulon, 2017
25 Guo Q, Qiu X, Liu P, et al. Star-transformer. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, 2019. 1315–1325
26 Erhan D, Bengio Y, Courville A C, et al. Why does unsupervised pre-training help deep learning? J Mach Learn Res, 2010, 11: 625–660
27 Hinton G E. Reducing the dimensionality of data with neural networks. Science, 2006, 313: 504–507
28 Hinton G, McClelland J, Rumelhart D. Distributed representations. The Philosophy of Artificial Intelligence, 1990, 248–280
29 Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model. J Mach Learn Res, 2003, 3: 1137–1155
30 Collobert R, Weston J, Bottou L, et al. Natural language processing (almost) from scratch. J Mach Learn Res, 2011, 12: 2493–2537
31 Le Q, Mikolov T. Distributed representations of sentences and documents. In: Proceedings of the International Conference on Machine Learning. Beijing, 2014. 1188–1196
32 Kiros R, Zhu Y, Salakhutdinov R R, et al. Skip-thought vectors. In: Proceedings of the Advances in Neural Information Processing Systems. Montreal, 2015. 3294–3302
33 Melamud O, Goldberger J, Dagan I. Context2Vec: Learning generic context embedding with bidirectional LSTM. In: Proceedings of the Conference on Computational Natural Language Learning. Berlin, 2016. 51–61
34 Dai A M and Le Q V. Semi-supervised sequence learning. In: Proceedings of the Advances in Neural Information Processing Systems. Montreal, 2015. 3079–3087
35 Ramachandran P, Liu P J, Le Q. Unsupervised pretraining for sequence to sequence learning. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Copenhagen, 2017. 383–391
36 Akbik A, Blythe D, Vollgraf R. Contextual string embeddings for sequence labeling. In: Proceedings of the International Conference on Computational Linguistics. Santa Fe, 2018. 1638–1649
37 Howard J and Ruder S. Universal language model fine-tuning for text classification. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Melbourne, 2018. 328–339
38 Baevski A, Edunov S, Liu Y, et al. Cloze-driven pretraining of self-attention networks. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 5359–5368
39 Taylor W L. "Cloze Procedure": A new tool for measuring readability. Jism Q, 1953, 30: 415–433
40 Song K, Tan X, Qin T, et al. MASS: Masked sequence to sequence pre-training for language generation. In: Proceedings of the International Conference on Machine Learning. Long Beach, 2019. 5926–5936
41 Raffel C, Shazeer N, Roberts A, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. ArXiv: 1910.10683
42 Liu Y, Ott M, Goyal N, et al. RoBERTa: A robustly optimized BERT pretraining approach. ArXiv: 1907.11692
43 Dong L, Yang N, Wang W, et al. Unified language model pre-training for natural language understanding and generation. In: Proceedings of the Advances in Neural Information Processing Systems. Vancouver, 2019. 13042–13054
44 Bao H, Dong L, Wei F, et al. UniLMv2: Pseudo-masked language models for unified language model pre-training. ArXiv: 2002.12804
45 Conneau A, Lample G. Cross-lingual language model pretraining. In: Proceedings of the Advances in Neural Information Processing Systems. Vancouver, 2019. 7057–7067
46 Joshi M, Chen D, Liu Y, et al. SpanBERT: Improving pre-training by representing and predicting spans. Transactions of the Association for Computational Linguistics, 2019, 8: 64–77
47 Wang W, Bi B, Yan M, et al. StructBERT: Incorporating language structures into pre-training for deep language understanding. In: Proceedings of the International Conference on Learning Representations. Addis Ababa, 2020.
48 Yang Z, Dai Z, Yang Y, et al. XLNet: Generalized autoregressive pretraining for language understanding. In: Proceedings of the Advances in Neural Information Processing Systems. Vancouver, 2019. 5754–5764
49 Lewis M, Liu Y, Goyal N, et al. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. ArXiv: 1910.13461
50 Saunshi N, Plevrakis O, Arora S, et al. A theoretical analysis of contrastive unsupervised representation learning. In: Proceedings of the International Conference on Machine Learning. Long Beach, 2019. 5628–5637
51 Mnih A, Kavukcuoglu K. Learning word embeddings efficiently with noise-contrastive estimation. In: Proceedings of the Advances in Neural Information Processing Systems. Lake Tahoe, 2013. 2265–2273
52 Gutmann M, Hyvärinen A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: Proceedings of the International Conference on Artificial Intelligence and Statistics. Chia Laguna Resort, 2010. 297–304
53 Hjelm R D, Fedorov A, Lavoie-Marchildon S, et al. Learning deep representations by mutual information estimation and maximization. In: Proceedings of the International Conference on Learning Representations. New Orleans, 2019
54 Kong L, de Masson d'Autume C, Yu L, et al. A mutual information maximization perspective of language representation learning. In: Proceedings of the International Conference on Learning Representations. New Orleans, 2019
55 Clark K, Luong M T, Le Q V, et al. ELECTRA: Pre-training text encoders as discriminators rather than generators. In: Proceedings of the International Conference on Learning Representations. Addis Ababa, 2020
56 Xiong W, Du J, Wang W Y, et al. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model. In: Proceedings of the International Conference on Learning Representations. Addis Ababa, 2020
57 Lan Z, Chen M, Goodman S, et al. ALBERT: A lite BERT for self-supervised learning of language representations. In: Proceedings of the International Conference on Learning Representations. Addis Ababa, 2020
58 de Vries W, van Cranenburgh A, Bisazza A, et al. BERTje: A Dutch BERT model. ArXiv: 1912.09582
59 Wang X, Gao T, Zhu Z, et al. KEPLER: A unified model for knowledge embedding and pre-trained language representation. ArXiv: 1911.06136
60 Radford A, Wu J, Child R, et al. Language models are unsupervised multitask learners. OpenAI Blog. 2019
61 Zhang Z, Han X, Liu Z, et al. ERNIE: enhanced language representation with informative entities. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Florence, 2019. 1441–1451.
62 Peters M E, Neumann M, IV R L L, et al. Knowledge enhanced contextual word representations. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong,

2019. 43–54
63 Liu W, Zhou P, Zhao Z, et al. K-BERT: Enabling language representation with knowledge graph. In: Proceedings of the AAAI Conference on Artificial Intelligence. New York, 2020. 2901–2908
64 Chi Z, Dong L, Wei F, et al. Cross-lingual natural language generation via pre-training. In: Proceedings of the AAAI Conference on Artificial Intelligence. New York, 2020. 7570–7577
65 Liu Y, Gu J, Goyal N, et al. Multilingual denoising pre-training for neural machine translation. ArXiv: 2001.08210
66 Conneau A, Khandelwal K, Goyal N, et al. Unsupervised cross-lingual representation learning at scale. ArXiv: 1911.02116
67 Ke P, Ji H, Liu S, et al. SentiLR: Linguistic knowledge enhanced language representation for sentiment analysis. ArXiv: 1911.02493
68 Huang H, Liang Y, Duan N, et al. Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 2485–2494
69 Eisenschlos J, Ruder S, Czapla P, et al. MultiFiT: Efficient multi-lingual language model fine-tuning. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 5701–5706
70 Sun Y, Wang S, Li Y, et al. ERNIE: enhanced representation through knowledge integration. ArXiv: 1904.09223
71 Cui Y, Che W, Liu T, et al. Pre-training with whole word masking for chinese BERT. ArXiv: 1906.08101
72 Wei J, Ren X, Li X, et al. NEZHA: Neural contextualized representation for chinese language understanding. ArXiv: 1909.00204
73 Diao S, Bai J, Song Y, et al. ZEN: Pre-training chinese text encoder enhanced by n-gram representations. ArXiv: 1911.00720
74 Martin L, Müller B, Suárez P J O, et al. CamemBERT: A tasty French language model. ArXiv: 1911.03894
75 Le H, Vial L, Frej J, et al. FlauBERT: Unsupervised language model pre-training for French. ArXiv: 1912.05372
76 Delobelle P, Winters T, Berendt B. RobBERT: A Dutch RoBERTa-based language model. ArXiv: 2001.06286
77 Lu J, Batra D, Parikh D, et al. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In: Proceedings of the Advances in Neural Information Processing Systems. Vancouver, 2019. 13–23
78 Tan H, Bansal M. LXMERT: Learning cross-modality encoder representations from transformers. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 5099–5110
79 Li L H, Yatskar M, Yin D, et al. VisualBERT: A simple and performant baseline for vision and language. ArXiv: 1908.03557
80 Alberti C, Ling J, Collins M, et al. Fusion of detected objects in text for visual question answering. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 2131–2140
81 Su W, Zhu X, Cao Y, et al. VL-BERT: Pre-training of generic visual-linguistic representations. In: Proceedings of the International Conference on Learning Representations. Addis Ababa, 2020
82 Sun C, Myers A, Vondrick C, et al. VideoBERT: A joint model for video and language representation learning. In: Proceedings of the International Conference on Computer Vision. Seoul, 2019. 7463–7472
83 Sun C, Baradel F, Murphy K, et al. Contrastive bidirectional transformer for temporal representation learning. ArXiv: 1906.05743
84 Chuang Y, Liu C, Lee H. SpeechBERT: Cross-modal pre-trained language model for end-to-end spoken question answering. ArXiv: 1910.11559
85 Lee J, Yoon W, Kim S, et al. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. Bioinformatics, 2020, 36: 1234–1240
86 Beltagy I, Lo K, Cohan A. SciBERT: A pretrained language model for scientific text. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 3613–3618
87 Lee J and Hsiang J. PatentBERT: Patent classification with fine-tuning a pre-trained BERT model. ArXiv: 1906.02124
88 Gordon M A, Duh K, Andrews N. Compressing BERT: Studying the effects of weight pruning on transfer learning. ArXiv: 2002.08307
89 Shen S, Dong Z, Ye J, et al. Q-BERT: Hessian based ultra low precision quantization of BERT. In: Proceedings of the AAAI Conference on Artificial Intelligence. New York, 2020. 8815–8821
90 Zafrir O, Boudoukh G, Izsak P, et al. Q8BERT: Quantized 8bit BERT. ArXiv: 1910.06188
91 Sanh V, Debut L, Chaumond J, et al. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. ArXiv: 1910.01108
92 Jiao X, Yin Y, Shang L, et al. TinyBERT: Distilling BERT for natural language understanding. ArXiv: 1909.10351
93 Wang W, Wei F, Dong L, et al. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. ArXiv: 2002.10957
94 Xu C, Zhou W, Ge T, et al. BERT-of-Theseus: Compressing BERT by progressive module replacing. ArXiv: 2002.02925
95 Mikolov T, Yih W, Zweig G. Linguistic regularities in continuous space word representations. In: Proceedings of the Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics. Atlanta, 2013. 746–751
96 Rubinstein D, Levi E, Schwartz R, et al. How well do distributional models capture different types of semantic knowledge? In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Beijing, 2015. 726–730
97 Gupta A, Boleda G, Baroni M, et al. Distributional vectors encode referential attributes. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Lisbon, 2015. 12–21
98 Tenney I, Xia P, Chen B, et al. What do you learn from context? Probing for sentence structure in contextualized word representations. In: Proceedings of the International Conference on Learning Representations. New Orleans, 2019
99 Liu N F, Gardner M, Belinkov Y, et al. Linguistic knowledge and transferability of contextual representations. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, 2019. 1073–1094
100 Tenney I, Das D, Pavlick E. BERT rediscovers the classical NLP pipeline. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Florence, 2019. 4593–4601
101 Goldberg Y. Assessing BERT's syntactic abilities. ArXiv: 1901.05287
102 Ettinger A. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. Trans Associat Comput Linguist, 2020, 8: 34–48
103 Hewitt J, Manning C D. A structural probe for finding syntax in word representations. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, 2019. 4129–4138
104 Jawahar G, Sagot B, Seddah D. What does BERT learn about the structure of language? In: Proceedings of the Conference of the Association for Computational Linguistics. Florence, 2019. 3651–3657
105 Kim T, Choi J, Edmiston D, et al. Are pre-trained language models aware of phrases? Simple but strong baselines for grammar induction. In: Proceedings of the International Conference on Learning Representations. Addis Ababa, 2020
106 Reif E, Yuan A, Wattenberg M, et al. Visualizing and measuring the

geometry of BERT. In: Proceedings of the Advances in Neural Information Processing Systems. Vancouver, 2019. 8592–8600
107 Petroni F, Rocktäschel T, Riedel S, et al. Language models as knowledge bases? In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 2463–2473
108 Jiang Z, Xu F F, Araki J, et al. How can we know what language models know? ArXiv: 1911.12543
109 Pörner N, Waltinger U, Schütze H. BERT is not a knowledge base (yet): Factual knowledge vs. name-based reasoning in unsupervised QA. ArXiv: 1911.03681
110 Kassner N, Schütze H. Negated LAMA: Birds cannot fly. ArXiv: 1911.03343
111 Bouraoui Z, Camacho-Collados J, Schockaert S. Inducing relational knowledge from BERT. In: Proceedings of the AAAI Conference on Artificial Intelligence. New York, 2020. 7456–7463
112 Davison J, Feldman J, Rush A M. Commonsense knowledge mining from pretrained models. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 1173–1178
113 Lauscher A, Vulic I, Ponti E M, et al. Informing unsupervised pretraining with external linguistic knowledge. ArXiv: 1909.02339
114 Wang R, Tang D, Duan N, et al. K-adapter: Infusing knowledge into pre-trained models with adapters. ArXiv: 2002.01808
115 Levine Y, Lenz B, Dagan O, et al. SenseBERT: Driving some sense into BERT. ArXiv: 1908.05646
116 Guan J, Huang F, Zhao Z, et al. A knowledge-enhanced pretraining model for commonsense story generation. ArXiv: 2001.05139
117 He B, Zhou D, Xiao J, et al. Integrating graph contextualized knowledge into pre-trained language models. ArXiv: 1912.00147
118 Wang Z, Zhang J, Feng J, et al. Knowledge graph and text jointly embedding. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Doha, 2014. 1591–1601
119 Zhong H, Zhang J, Wang Z, et al. Aligning knowledge and text embeddings by entity descriptions. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Lisbon, 2015. 267–272
120 Xie R, Liu Z, Jia J, et al. Representation learning of knowledge graphs with entity descriptions. In: Proceedings of the AAAI Conference on Artificial Intelligence. Phoenix, 2016. 2659–2665
121 Xu J, Qiu X, Chen K, et al. Knowledge graph representation with jointly structural and textual encoding. In: Proceedings of the International Joint Conference on Artificial Intelligence. Melbourne, 2017. 1318–1324
122 Yang A, Wang Q, Liu J, et al. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In: Proceedings of the Conference of the Association for Computational Linguistics. Florence, 2019. 2346–2357
123 Logan IV R L, Liu N F, Peters M E, et al. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. In: Proceedings of the Conference of the Association for Computational Linguistics. Florence, 2019. 5962–5971
124 Hayashi H, Hu Z, Xiong C, et al. Latent relation language models. In: Proceedings of the AAAI Conference on Artificial Intelligence. New York, 2020. 7911–7918
125 Faruqui M, Dyer C. Improving vector space word representations using multilingual correlation. In: Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics. Gothenburg, 2014. 462–471
126 Luong M T, Pham H, Manning C D. Bilingual word representations with monolingual quality in mind. In: Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing. Denver, 2015. 151–159
127 Singla K, Can D, Narayanan S. A multi-task approach to learning multilingual representations. In: Proceedings of Annual Meeting of the Association for Computational Linguistics. Melbourne, 2018. 214–220
128 Pires T, Schlinger E, Garrette D. How multilingual is multilingual BERT? In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Florence, 2019. 4996–5001
129 K K, Wang Z, Mayhew S, et al. Cross-lingual ability of multilingual BERT: An empirical study. In: Proceedings of the International Conference on Learning Representations. Addis Ababa, 2020
130 Virtanen A, Kanerva J, Ilo R, et al. Multilingual is not enough: BERT for Finnish. ArXiv: 1912.07076
131 Sun Y, Wang S, Li Y, et al. ERNIE 2.0: A continual pre-training framework for language understanding. In: Proceedings of the AAAI Conference on Artificial Intelligence. New York, 2020. 8968–8975
132 Kuratov Y, Arkhipov M. Adaptation of deep bidirectional multilingual transformers for russian language. ArXiv: 1905.07213
133 Antoun W, Baly F, Hajj H. AraBERT: Transformer-based model for Arabic language understanding. ArXiv: 2003.00104
134 Luo H, Ji L, Shi B, et al. UniViLM: A unified video and language pre-training model for multimodal understanding and generation. ArXiv: 2002.06353
135 Li G, Duan N, Fang Y, et al. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In: Proceedings of the AAAI Conference on Artificial Intelligence. New York, 2020. 11336–11344
136 Chen Y, Li L, Yu L, et al. UNITER: Learning universal image-text representations. ArXiv: 1909.11740
137 Huang K, Altosaar J, Ranganath R. ClinicalBERT: Modeling clinical notes and predicting hospital readmission. ArXiv: 1904.05342
138 Alsentzer E, Murphy J R, Boag W, et al. Publicly available clinical BERT embeddings. ArXiv: 1904.03323
139 Ji Z, Wei Q, Xu H. BERT-based ranking for biomedical entity normalization. ArXiv: 1908.03548
140 Tang M, Gandhi P, Kabir M A, et al. Progress notes classification and keyword extraction using attention-based deep learning models with BERT. ArXiv: 1910.05786
141 Zhang J, Zhao Y, Saleh M, et al. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. ArXiv: 1912.08777
142 Wang S, Che W, Liu Q, et al. Multi-task self-supervised learning for disfluency detection. In: Proceedings of the AAAI Conference on Artificial Intelligence. New York, 2020. 9193–9200
143 Bucilua C, Caruana R, Niculescu-Mizil A. Model compression. In: Proceedings of the International Conference on Knowledge Discovery and Data Mining. Philadelphia, 2006. 535–541
144 Ganesh P, Chen Y, Lou X, et al. Compressing large-scale transformer-based models: A case study on BERT. ArXiv: 2002.11985
145 Dong Z, Yao Z, Gholami A, et al. Hawq: Hessian aware quantization of neural networks with mixed-precision. In: Proceedings of the International Conference on Computer Vision. Seoul, 2019. 293–302
146 Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. ArXiv: 1503.02531
147 Sun S, Cheng Y, Gan Z, et al. Patient knowledge distillation for BERT model compression. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 4323–4332
148 Turc I, Chang M W, Lee K, et al. Well-read students learn better: The impact of student initialization on knowledge distillation. ArXiv: 1908.08962
149 Sun Z, Yu H, Song X, et al. MobileBERT: A compact task-agnostic BERT for resource-limited devices. ArXiv: 2004.02984
150 Zhao S, Gupta R, Song Y, et al. Extreme language model compression with optimal subwords and shared projections. ArXiv: 1909.11687

151 Rogers A, Kovaleva O, Rumshisky A. A primer in BERTology: What we know about how BERT works. ArXiv: 2002.12327

152 Michel P, Levy O, Neubig G. Are sixteen heads really better than one? In: Proceedings of the Advances in Neural Information Processing Systems. Vancouver, 2019. 14014–14024

153 Voita E, Talbot D, Moiseev F, et al. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In: Proceedings of the Conference of the Association for Computational Linguistics. Florence, 2019. 5797–5808

154 Dehghani M, Gouws S, Vinyals O, et al. Universal transformers. In: Proceedings of the International Conference on Learning Representations. New Orleans, 2019

155 Lu W, Jiao J, Zhang R. TwinBERT: Distilling knowledge to twin-structured BERT models for efficient retrieval. ArXiv: 2002.06275

156 Tsai H, Riesa J, Johnson M, et al. Small and practical BERT models for sequence labeling. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 3632–3636

157 Liu X, He P, Chen W, et al. Improving multi-task deep neural networks via knowledge distillation for natural language understanding. ArXiv: 1904.09482

158 Tang R, Lu Y, Liu L, et al. Distilling task-specific knowledge from BERT into simple neural networks. ArXiv: 1903.12136

159 Chia Y K, Witteveen S, Andrews M. Transformer to CNN: Label-scarce distillation for efficient text classification. ArXiv: 1909.03508

160 Liu W, Zhou P, Zhao Z, et al. FastBERT: A self-distilling BERT with adaptive inference time. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Online, 2020. 6035–6044

161 Pan S J, Yang Q. A survey on transfer learning. IEEE Trans Knowledge Data Eng, 2009, 22, 1345–1359

162 Belinkov Y, Durrani N, Dalvi F, et al. What do neural machine translation models learn about morphology? In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Vancouver, 2017. 861–872

163 Peters M E, Ruder S, Smith N A. To tune or not to tune? Adapting pretrained representations to diverse tasks. In: Proceedings of the 4th Workshop on Representation Learning for NLP, RepL4NLP@ACL 2019. Florence, 2019. 7–14

164 Zhong M, Liu P, Wang D, et al. Searching for effective neural extractive summarization: What works and what's next. In: Proceedings of the Conference of the Association for Computational Linguistics. Florence, 2019. 1049–1058

165 Zhu J, Xia Y, Wu L, et al. Incorporating BERT into neural machine translation. In: Proceedings of the International Conference on Learning Representations. Addis Ababa, 2020

166 Dodge J, Ilharco G, Schwartz R, et al. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. ArXiv: 2002.06305

167 Sun C, Qiu X, Xu Y, et al. How to fine-tune BERT for text classification? In: Proceedings of the China National Conference on Chinese Computational Linguistics. Kunming, 2019. 194–206

168 Phang J, Févry T, Bowman S R. Sentence encoders on STILTs: Supplementary training on intermediate labeled-data tasks. ArXiv: 1811.01088

169 Garg S, Vu T, Moschitti A. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. In: Proceedings of the AAAI Conference on Artificial Intelligence. New York, 2020. 7780–7788

170 Li Z, Ding X, Liu T. Story ending prediction by transferable bert. In: Proceedings of the International Joint Conference on Artificial Intelligence. Macao, 2019. 1800–1806

171 Liu X, He P, Chen W, et al. Multi-task deep neural networks for natural language understanding. In: Proceedings of the Conference of the Association for Computational Linguistics. Florence, 2019. 4487–4496

172 Stickland A C, Murray I. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. In: Proceedings of the International Conference on Machine Learning. Long Beach, 2019. 5986–5995

173 Houlsby N, Giurgiu A, Jastrzebski S, et al. Parameter-efficient transfer learning for NLP. In: Proceedings of the International Conference on Machine Learning. Long Beach, 2019. 2790–2799

174 Xu Y, Qiu X, Zhou L, et al. Improving BERT fine-tuning via self-ensemble and self-distillation. ArXiv: 2002.10345

175 Chronopoulou A, Baziotis C, Potamianos A. An embarrassingly simple approach for transfer learning from pretrained language models. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, 2019. 2089–2095

176 Li X L, Eisner J. Specializing word embeddings (for parsing) by information bottleneck. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 2744–2754

177 Gardner M, Grus J, Neumann M, et al. Allennlp: A deep semantic natural language processing platform. ArXiv: 1803.07640

178 Keskar N S, McCann B, Varshney L R, et al. CTRL: A conditional transformer language model for controllable generation. ArXiv: 1909.05858

179 Vig J. A multiscale visualization of attention in the transformer model. In: Proceedings of the Conference of the Association for Computational Linguistics. Florence, 2019. 37–42

180 Hoover B, Strobelt H, Gehrmann S. Exbert: A visual analysis tool to explore learned representations in transformers models. ArXiv: 1910.05276

181 Yang Z, Cui Y, Chen Z, et al. Textbrewer: An open-source knowledge distillation toolkit for natural language processing. ArXiv: 2002.12620

182 Wang Y, Hou Y, Che W, et al. From static to dynamic word representations: A survey. Int J Mach Learn Cyber, 2020, 11: 1611–1630

183 Liu Q, Kusner M J, Blunsom P. A survey on contextual embeddings. ArXiv: 2003.07278

184 Wang A, Singh A, Michael J, et al. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In: Proceedings of the International Conference on Learning Representations. New Orleans, 2019

185 Wang A, Pruksachatkun Y, Nangia N, et al. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In: Proceedings of the Advances in Neural Information Processing Systems. Vancouver, 2019. 3261–3275

186 Rajpurkar P, Zhang J, Lopyrev K, et al. Squad: 100, 000+ questions for machine comprehension of text. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Austin, 2016. 2383–2392

187 Reddy S, Chen D, Manning C D. CoQA: A conversational question answering challenge. Trans Associat Comput Linguist, 2019, 7: 249–266

188 Yang Z, Qi P, Zhang S, et al. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Brussels, 2018. 2369–2380

189 Zhang Z, Yang J, Zhao H. Retrospective reader for machine reading comprehension. ArXiv: 2001.09694

190 Ju Y, Zhao F, Chen S, et al. Technical report on conversational question answering. ArXiv: 1909.10772

191 Tu M, Huang K, Wang G, et al. Select, answer and explain: Inter-

pretable multi-hop reading comprehension over multiple documents. In: Proceedings of the AAAI Conference on Artificial Intelligence. New York, 2020. 9073–9080
192 Bataa E, Wu J. An investigation of transfer learning-based sentiment analysis in japanese. In: Proceedings of the Conference of the Association for Computational Linguistics. Florence, 2019. 4652–4657
193 Sun C, Huang L, Qiu X. Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, 2019. 380–385
194 Xu H, Liu B, Shu L, et al. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, 2019. 2324–2335
195 Rietzler A, Stabinger S, Opitz P, et al. Adapt or get left behind: Domain adaptation through BERT language model finetuning for aspect-target sentiment classification. ArXiv: 1908.11860
196 Karimi A, Rossi L, Prati A, et al. Adversarial training for aspect-based sentiment analysis with BERT. ArXiv: 2001.11316
197 Song Y, Wang J, Liang Z, et al. Utilizing BERT intermediate layers for aspect based sentiment analysis and natural language inference. ArXiv: 2002.04815
198 Li X, Bing L, Zhang W, et al. Exploiting BERT for end-to-end aspect-based sentiment analysis. In: Proceedings of the W-NUT@Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 34–41
199 Wu X, Zhang T, Zang L, et al. "Mask and infill": Applying masked language model to sentiment transfer. ArXiv: 1908.08039
200 Peters M E, Ammar W, Bhagavatula C, et al. Semi-supervised sequence tagging with bidirectional language models. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Vancouver, 2017. 1756–1765
201 Liu L, Ren X, Shang J, et al. Efficient contextualized representation: Language model pruning for sequence labeling. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Brussels, 2018. 1215–1225
202 Hakala K, Pyysalo S. Biomedical named entity recognition with multilingual BERT. In: Proceedings of the BioNLP Open Shared Tasks@Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 56–61
203 Edunov S, Baevski A, Auli M. Pre-trained language model representations for language generation. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, 2019. 4052–4059
204 Clinchant S, Jung K W, Nikoulina V. On the use of BERT for neural machine translation. In: Proceedings of the Proceedings of the 3rd Workshop on Neural Generation and Translation. Hong Kong, 2019. 108–117
205 Imamura K, Sumita E. Recycling a pre-trained BERT encoder for neural machine translation. In: Proceedings of the 3rd Workshop on Neural Generation and Translation. Hong Kong, 2019. 23–31
206 Zhang X, Wei F, Zhou M. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In: Proceedings of the Conference of the Association for Computational Linguistics. Florence, 2019. 5059–5069
207 Liu Y, Lapata M. Text summarization with pretrained encoders. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 3728–3738
208 Zhong M, Liu P, Chen Y, et al. Extractive summarization as text matching. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Online, 2020. 6197–6208
209 Jin D, Jin Z, Zhou J T, et al. Is BERT really robust? Natural language attack on text classification and entailment. In: Proceedings of the AAAI Conference on Artificial Intelligence. New York, 2020. 8018–8025
210 Wallace E, Feng S, Kandpal N, et al. Universal adversarial triggers for attacking and analyzing NLP. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. 2153–2162
211 Sun L, Hashimoto K, Yin W, et al. Adv-BERT: BERT is not robust on misspellings! Generating nature adversarial samples on BERT. ArXiv: 2003.04985
212 Li L, Ma R, Guo Q, et al. BERT-ATTACK: Adversarial attack against BERT using BERT. ArXiv: 2004.09984
213 Zhu C, Cheng Y, Gan Z, et al. FreeLB: Enhanced adversarial training for natural language understanding. In: Proceedings of the International Conference on Learning Representations. Addis Ababa, 2020
214 Liu X, Cheng H, He P C, et al. Adversarial training for large neural language models. ArXiv: 2004.08994
215 Shoeybi M, Patwary M, Puri R, et al. Megatron-LM: Training multi-billion parameter language models using gpu model parallelism. ArXiv: 1909.08053
216 Dai Z, Yang Z, Yang Y, et al. Transformer-XL: Attentive language models beyond a fixed-length context. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. Florence, 2019. 2978–2988
217 Zoph B, Le Q V. Neural architecture search with reinforcement learning. In: Proceedings of the International Conference on Learning Representations. Toulon, 2017
218 Cheng Y, Wang D, Zhou P, et al. A survey of model compression and acceleration for deep neural networks. ArXiv: 1710.09282
219 Wu X, Lv S, Zang L, et al. Conditional BERT contextual augmentation. In: Proceedings of the International Conference on Computational Science. Faro, 2019. 84–95
220 Kumar V, Choudhary A, Cho E. Data augmentation using pre-trained transformer models. ArXiv: 2003.02245
221 Barredo Arrieta A, Díaz-Rodríguez N, Del Ser J, et al. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Inf Fusion, 2020, 58: 82–115
222 Jain S, Wallace B C. Attention is not explanation. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, 2019. 3543–3556
223 Serrano S, Smith N A. Is attention interpretable? In: Proceedings of the Conference of the Association for Computational Linguistics. Florence, 2019. 2931–2951