APSIPA Transactions on Signal and Information Processing, 2024, 13, e101
This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (http://creativecommons.org/licenses/by-nc/4.0/), which permits unrestricted re-use, distribution, and reproduction in any medium, for non-commercial use, provided the original work is properly cited.

Overview Paper

An Overview of Language Models: Recent Developments and Outlook

Chengwei Wei[1*], Yun-Cheng Wang[1], Bin Wang[2] and C.-C. Jay Kuo[1]

[1] *University of Southern California, USA*
[2] *National University of Singapore, Singapore*

ABSTRACT

Language modeling studies the probability distributions over strings of texts. It is one of the most fundamental tasks in natural language processing (NLP). It has been widely used in text generation, speech recognition, machine translation, etc. Conventional language models (CLMs) aim to predict the probability of linguistic sequences in a causal manner, while pre-trained language models (PLMs) cover broader concepts and can be used in both causal sequential modeling and fine-tuning for downstream applications. PLMs have their own training paradigms (usually self-supervised) and serve as foundation models in modern NLP systems. This overview paper provides an introduction to both CLMs and PLMs from five aspects, i.e., linguistic units, architectures, training methods, evaluation methods, and applications. Furthermore, we discuss the relationship between CLMs and PLMs and shed light on the future directions of language modeling in the pre-trained era.

Keywords: Language model, Natural language processing, Pre-trained language model, Conventional language model.

*Corresponding author: Chengwei Wei, chengwei@usc.edu.

Received 14 March 2023; Revised 30 July 2023
ISSN 2048-7703; DOI 10.1561/116.00000010
© 2024 C. Wei, Y.-C. Wang, B. Wang and C.-C. Jay Kuo

1 Introduction

Language modeling studies the probability distributions over a sequence of linguistic units, such as words. It is one of the most fundamental tasks and long-standing research topics in natural language processing (NLP). The developed language models (LMs) find applications in many computational linguistic problems such as text generation, machine translation, speech recognition, natural language generation, question-and-answer systems, etc.

There are two major approaches to language modeling: 1) the statistical approach based on a relatively small corpus set, and 2) the data-driven approach based on a significantly larger corpus set. Conventional language models (CLMs) predict the probability of linguistic sequences in a causal manner. They can be learned by both language modeling approaches. The data-driven approach has become mainstream nowadays. It exploits a large number of corpora to train neural-network models, leading to pre-trained language models (PLMs). PLMs are then fine-tuned with task-specific datasets and objectives for downstream applications. In this overview paper, we define CLMs as language models that predict the probability of linguistic sequences in a causal manner. In contrast, PLMs refer to language models pre-trained on a broad range of linguistic tasks and objectives. It is important to note that the two concepts are not exclusive. One LM can fall into both categories. For example, GPT models [133] can predict the probability of linguistic sequences in a causal manner. They are also pre-trained with various downstream tasks. We provide an overview of CLMs and PLMs and study them from five perspectives: 1) linguistic units, 2) architectures, 3) training methods, 4) evaluation methods, and 5) applications. In the end, we point out several future research directions.

The goal of CLMs is to model the probability distributions over sequences of linguistic units:

$$P(u_1, u_2, \cdots, u_t), \tag{1}$$

where u_i can be either a character, a word, a phrase, or other linguistic units. CLMs attempt to predict the next linguistic unit in a text sequence given its preceding contexts:

$$P(u_t|u_{<t}) \tag{2}$$

CLMs are also called auto-regressive language models since the units are predicted in a causal way. Estimating the probability of a text sequence as shown in Equation (1) directly encounters the data sparsity problem. CLMs often estimate the joint probability of the text sequence by decomposing a text sequence into smaller units. For example, CLMs leverage the chain rule and the conditional probability to estimate the joint probability in the form of

$$P(u_1, u_2, \ldots, u_t) = P(u_1)P(u_2|u_1)P(u_3|u_1, u_2)\cdots P(u_t|u_1, \ldots, u_{t-1}). \tag{3}$$

Before the pre-training era, CLMs are often trained from scratch with a training corpus and, then, predict the probability of text sequences with respective applications. Representative models include N-grams LMs [16, 40, 123], exponential LMs [12, 34, 142] and earlier neural LMs [11, 114]. CLMs give a high probability to natural text sequences occurring frequently in the real world. As a result, they play a fundamental role in text generation, speech recognition [9, 72, 74], and machine translation [15, 124, 208] until the emergence of PLMs. Nowadays, high-performance PLMs serve as the backbone of many NLP systems. They are not limited to the causal predictive functionality of CLMs and provide more different types of LMs.

The differences between CLMs before the pre-training era and PLMs can be summarized below.

- **Training Methodology.** With the development of deep learning, PLMs with neural network structures are pre-trained by collections of massive unlabeled corpora to learn generic knowledge which is then transferred to downstream tasks by task-specific fine-tuning.

- **Causality Constraint.** PLMs do not necessarily follow CLMs in predicting linguistic units as shown in Equation (2). For example, bidirectional LMs [36, 107] use both preceding and succeeding contexts to predict the missing linguistic units via probability estimation:

$$P(u_t|u_{<t}, u_{>t}). \tag{4}$$

  Bidirectional LMs do not follow the causality constraint and the chain rule in Equation (3), to access the probability of a text sequence, which makes it inherently different from CLMs.

- **Token Representation.** Apart from the differences in the training paradigm and probability modeling, PLMs adopt a different representation for basic units called tokens. PLMs represent tokens by embedding them in a high-dimensional continuous space such as word embeddings [126, 198] and sentence embeddings [44, 186, 187]. The new representations offer a flexible and powerful tool that enables PLMs to handle a wide range of tasks.

This overview paper serves two objectives. On one hand, instead of only focusing on recently developed PLMs [55, 106, 132], we aim to provide a comprehensive overview of the basic concepts of LMs, the transition from CLMs to PLMs, LM's recent developments and applications to beginners in the field. On the other hand, we would like to shed light on future research directions and offer our outlook to experienced engineers and researchers in the NLP field. For example, we cover large LMs (LLMs) in the survey as there are growing interests in LLMs due to the new services provided by ChatGPT.

Furthermore, we include efficient LMs as an emerging topic since there are increasing concerns about large model sizes and high training costs of LLMs.

The rest of the paper is organized as below. We introduce several types of LMs that go beyond CLMs in Section 2, and provide an overview of common ways to decompose text sequences into smaller linguistic units in Section 3. Section 4 introduces different model architectures. We discuss the training procedures of LMs in Section 5. Common evaluation methods including, both intrinsic and extrinsic ones, are introduced in Section 6. The application of LMs to text generation is discussed in Section 7. We comment on the redundancy problem of LMs and analyze techniques for efficient LMs in Section 8. Promising future research directions are pointed out in Section 9. Concluding remarks are given in Section 10

2 Types of Language Models

CLMs commonly refer to auto-regressive models that predict the next linguistic units given the preceding context as shown in Equation (2). LMs can access the probability of a text sequence using the chain rule. The goal of CLMs is to decode the probability of text sequences in a causal manner. In this section, we introduce more LMs that go beyond CLMs.

2.1 Structural LM

Instead of predicting linguistic units in a sequential or reversed sequential order, structural LMs [19, 20, 52, 117, 199] predict linguistic units based on pre-defined linguistic structures such as dependency or constituent parse trees. Structural LMs utilize the linguistic structure to bring linguistically relevant context closer to the linguistic unit to be predicted. For example, given a parse tree structure, a structural LM can define the ancestor context $A(u_t)$ of u_t as the sequence from the root node to the parent of u_t. For example, the ancestor sequence of word 'strong' is {'binoculars', 'saw', ROOT} in Figure 1. Then, the structural LM uses the ancestor context in the tree to predict the next linguistic unit as

$$P(u_t|A(u_t)), \tag{5}$$

where $A(u_t)$ is the ancestor context of linguistic unit u_t. Similar to CLMs, structural LMs are designed to model the probability of text sequences. Differently, structural LMs decode the sequence probability in the order of their synthetic structures. It has been successfully applied to sentence completion [52, 117] and speech recognition [19, 20].

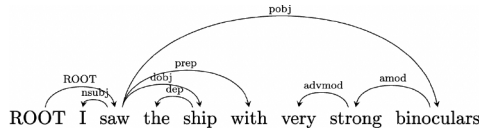

Figure 1: The example of a dependency parse tree example [117].

2.2 Bidirectional LM

Instead of using the causal contexts to make predictions, bidirectional LMs utilize contexts from both directions as shown in Equation (4). The masked LM is one representative bidirectional LM. It masks out linguistic units in a text sequence and, then, encodes their preceding and succeeding contexts to predict the masked linguistic units. Formally, the prediction can be defined as the estimation of the following conditional probability

$$P(u_m|\bar{S}), \tag{6}$$

where u_m is the masked linguistic unit and $\bar{S}$ is the corrupted text sequence by replacing a certain number of linguistic units with `[MASK]` symbols. The goal of bidirectional LMs is to learn the inner dependency between linguistic units in an unsupervised manner. The trained model can inherit semantics meanings from large-scale unlabeled corpora. Different from CLMs that aim to model the generation probability of text sequences, pre-trained bidirectional LMs are used as the backbone that transfers the learned knowledge through further fine-tuning in various downstream applications.

2.3 Permutation LM

CLMs and masked LMs have their own advantages and disadvantages. A masked LM needs to create artificial tokens such as `[mask]`, which never occur in downstream tasks while CLMs only condition on preceding context. The permutation LM [211] is a recently proposed LM that takes advantage of CLMs and masked LMs. Given an input sequence of linguistic units, permutation LMs randomize the order of input linguistic units and construct different permutations of the input sequence. Figure 2 shows an example of different permutations given an input text sequence. Let $\mathbb{Z}$ be the set of all possible permutations. Permutation LMs predict the next linguistic unit, u_t, in one permutation, Z, of the sequence based on

$$P(u_t|u^Z_{<t}), Z \in \mathbb{Z}. \tag{7}$$

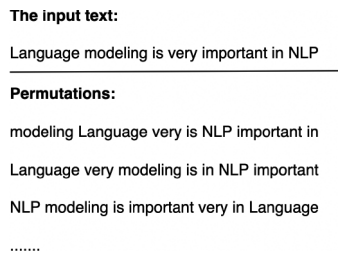

Figure 2: The use of different permutations in a natural sentence.

3 Linguistic Units

To estimate the probability of text sequences, LMs partition text sequences into small linguistic units such as characters, words, phrases, or sentences. This process is called tokenization. The resulting linguistic units are called tokens. Different languages and models may have different appropriate tokenization methods. Here, we focus on English and use it as an example. In this section, we examine typical tokenization methods used in language modeling according to unit sizes.

3.1 Characters

LMs can model text sequences probability based on characters [67, 86, 136, 170, 207]. As compared with other linguistics units, using characters has a much smaller vocabulary size, leading to a smaller discrete space and model size. On the other hand, it is challenging to predict the next character. Usually, it requires a long historical context. This makes the performance of character-level LMs poorer than that of word-level LMs. In addition, the input and output lengths have to be longer to model the character distribution accurately. This results in higher computational costs, especially for auto-regressive decoding. Several LM methods use the combination of words and characters to alleviate the issue [80, 118, 180].

3.2 Words and Subwords

The most natural tokenization for English is to decompose a text sequence into words by white spaces. Many LMs apply word tokenization. However, there are several issues of naive word tokenization. The first one is the Out-Of-Vocabulary (OOV) problem. Because an LM has a pre-defined vocabulary size that cannot be arbitrarily large. Less frequent words and words with character-level errors may not be stored in the pre-defined vocabulary. Thus, they cannot be retrieved from the dictionary. Although one can extend the

vocabulary size to alleviate this problem, it will increase the model size and still cannot handle all possible words.

LMs beyond the word level still have the OOV problem while a single character is not semantically meaningful by themselves. Recently, researchers are in favor of decomposing words into subwords if they do not appear in the dictionary. This offers a flexible and effective solution to the OOV problem [116, 158]. Several subword segmentation algorithms are developed to boost the performance of LMs. They strike a balance between the good performance of word-level models and the flexibility of character-level models. Two subword segmentation approaches, statistics-based and linguistics-based, are presented below.

3.2.1 Statistics-based Subword Tokenizers

The statistics-based subword tokenizers generate subword vocabulary purely based on the corpus. The associated methods are derived from a compression point of view. They work by replacing the commonly appeared character sequences with a new symbol (word) that does not exist in the current vocabulary. Then, fewer bytes are needed for information transmission.

Byte Pair Encoding (BPE). BPE [42] is a simple data compression technique that replaces the most common pair of bytes in a sequence by a single unused byte recursively. It was adopted by [158] to solve the word segmentation problem. That is, frequent characters or character sequences are merged to generate subwords. BPE is also used by several advanced PLMs such as GPT-2 [134] and RoBERTa [107] with the following algorithm, called the BPE merge operation.

1. Prepare a training corpus and define the size of the subword vocabulary.
2. Split all words into characters.
3. Generate a new subword by merging a pair of characters or subwords with the highest frequency.
4. Repeat step 3 until the desired vocabulary size is reached.

An illustration of the BPE merge operation conducted on a small dictionary is given in Figure 3.

WordPiece. [154] WordPiece is another data-driven subword algorithm. The difference between WordPiece and BPE is that WordPiece merges the pair of A and B if they have the highest score $P(AB)/P(A)P(B)$ (rather than the highest frequency $P(AB)$) at each iterative step. For example, WordPiece merges the pair of "u" and "g" in Figure 3 only if they have the highest value, $P('ug')/P('u')P('g')$, as compared with other pairs. WordPiece is used as the tokenization method in BERT [36], DistilBERT [148], and Electra [27].

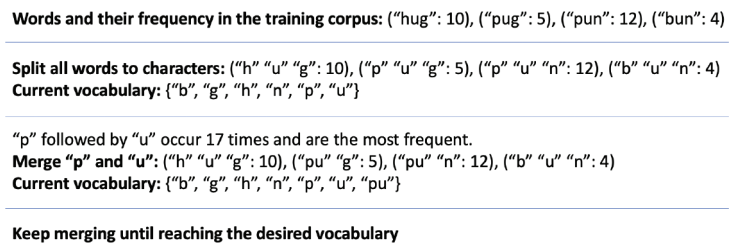

Figure 3: Illustration of the BPE merge operation conducted on the dictionary {"hug", "pug", "pun", "bun"}. The vocabulary is initialized with all characters. Then, a new subword is created by merging the most frequent pair.

There are other statistics-based subword tokenizers such as **Unigram** [90]. SentencePiece,[1] Huggingface tokenizers,[2] and OpenNMT[3] are popular tokenizers. Their implementation contains the statistics-based subword tokenization. Different subword tokenizers and their performance comparison are studied in [14].

3.2.2 Linguistics-based Subword Tokenizers

Linguistics-based subword tokenizers exploit the linguistic knowledge and decompose words into smaller grammatical units, such as morphemes or syllables. Such subword tokenizers are widely used in machine translation and speech recognition among different languages [2, 28, 29, 84, 144, 146, 150]. For example, in machine translation, words formed by compounding, affixation, or inflection, can be conveniently translated by translating the morphemes, respectively. However, linguistics-based subword tokenizers are not as popular as statistics-based ones due to the complexity and the rule-based nature of language decomposition.

3.3 Phrases

The semantic meaning of a single word can be ambiguous because of various contexts and set collocations. Since the linguistic dictionary does not go beyond the word-level, the inter-word dependency is ignored. Phrase-level LMs replace common and cohesive word sequences by phrases [98, 138, 149, 168]. Phrase-level LMs are suitable for some applications. For example, it is observed in [149] that short words with fewer syllables in automatic speech recognition (ASR) are more frequently misrecognized than longer ones. Since

[1] https://github.com/google/sentencepiece.
[2] https://github.com/huggingface/tokenizers.
[3] https://github.com/OpenNMT/Tokenizer.

phrases provide longer phone sequences than their constituents, they are more robust to recognition errors for ASR.

3.4 Sentences

Auto-regressive LMs with smaller linguistic units (e.g., characters, words, subwords, and phrases) rely on conditional probabilities to estimate the probability of text sequences as given in Equation (3). Sentence-level LMs [22, 69, 96, 140, 141] avoid the use of the chain rule. They generate sentence features and, then, model the sentence probability directly. This is because modeling the sentence probability directly is more convenient than that in Equation (3) in encoding the sentence-level information. It is also easier to encode the inter-sentence information such as the effects of preceding utterances in a dialog flow.

4 Architecture of Language Models

In this section, we conduct a survey on several common architectures to model the probability distributions of text sequences. They are N-gram, maximum entropy, and neural network models. While there are other LM architectures, like Gaussian mixture LMs [3] and Hidden Markov LMs [91], we focus on the above-mentioned architectures due to their popularity in the research community. Furthermore, LMs can operate at various levels of linguistic units. For generality and consistency with the most recent literature, we use the term 'token' to refer to all linguistic units leveraged by different LMs for the rest of this paper.

4.1 N-gram Models

An N-gram consists of N consecutive tokens from a text sequence. N-gram LMs [16, 40, 123] assume that the probability of a token depends only on its preceding N-1 tokens and it is independent of other contexts. This is known as the Markov assumption. Thus, instead of using all historical contexts, N-gram LMs only use the previous N-1 tokens to predict the current one; namely,

$$P(u_t|u_{<t}) = P(u_t|u_{t-N+1:t-1}). \tag{8}$$

N-gram LMs calculate the conditional probability by counting the occurrence time of N-grams given a training corpus as

$$P(u_t|u_{t-N+1:t-1}) = \frac{C(u_{t-N+1:t})}{C(u_{t-N+1:t-1})}. \tag{9}$$

N-gram LMs simplify the token probability calculation based on previous N-1 tokens, but they encounter two sparsity issues. First, if an N-gram,

$(u_{t-N+1:t})$, never occurs in the training corpus, the probability for the next tokens being u_t is zero. Second, if the (N-1)-gram, $(u_{t-N+1:t-1})$, in the denominator never occurs, we cannot calculate the probability of any tokens. These sparsity issues can be alleviated by smoothing techniques. A simple smoothing method [79, 104], called additive smoothing, is to add a small value to the count for every N-gram so as to avoid zero in the numerator and the denominator in Equation (9). However, this simple smoothing is still deficient because it assigns the same probability for N-grams that never occur in the training corpus.

There are more advanced smoothing techniques such as back-off and interpolation [21, 25, 73, 83, 88] that achieve better probability estimation. In back-off, lower-order N-grams are used for probability estimation if higher-order N-grams do not occur. For example, if $C(u_{t-3:t-1}) = 0$, we back off to compute $P(u_t|u_{t-2:t-1})$. In interpolation, different N-grams are considered for conditional probability computation. Mathematically, the N-gram probability is estimated by

$$\begin{aligned} P(u_t|u_{t-N+1:t-1}) = \lambda_N P(u_t|u_{t-N+1:t-1}) + \lambda_{N-1} P(u_t|u_{t-N:t-1}) \\ + \lambda_{N-2} P(u_t|u_{t-N-1:t-1}) + \cdots + \lambda_1 P(u_t), \end{aligned} \tag{10}$$

where λ_i is the weight for each n-gram and $\sum_{i=1}^{N} \lambda_i = 1$.

4.2 Maximum Entropy Models

Maximum Entropy models (also called the exponential models) [12, 34, 142] estimate the probability of text sequences using feature functions in the form of

$$P(u|h) = \frac{exp(a^T f(u, u_{<t}))}{\sum_{u'} exp(a^T f(u', u'_{<t}))}, \tag{11}$$

where $f(u, u_{<t})$ is the feature function that generates the feature of token u and its historical context $u_{<t}$, $\sum_{w'} exp(a^T f(u', u'_{<t}))$ is a normalization factor, and a is a parameter vector derived by the Generalized Iterative Scaling algorithm [32]. The features are usually generated from the N-grams.

4.3 Feed-forward Neural Network (FNN) Models

The discrete nature of the N-gram model is its performance bottleneck even with advanced smoothing techniques. Neural LMs embrace the continuous embedding space (distributed representation) to overcome the data sparsity problem. Feed-forward Neural Network (FNN) LMs [5, 11, 156, 157] is one of the earlier neural network models.

An FNN LM takes historical contexts as the input, and outputs the probability distribution of tokens. As shown in Figure 4, each token in the

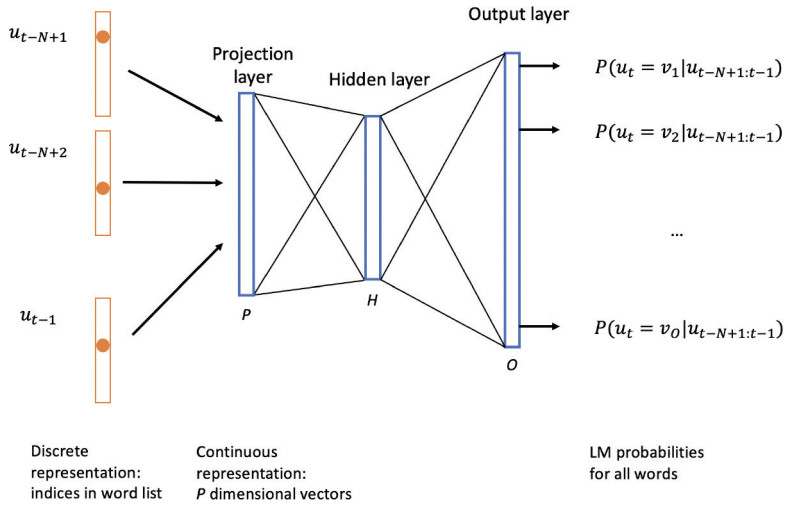

Figure 4: The structure of FFN LMs, where $u_{t-N+1}, \ldots, u_{t-1}$ denotes the preceding contexts of u_t in a fixed-window, and P, H, and O are the dimensions of the projection, the hidden layer, and the output layer, respectively.

preceding context is represented as a vector through a projection layer (i.e., an embedding matrix). These vectors of tokens are sent to the hidden layer with H hidden units followed by non-linear activation. Then, a softmax function is used to obtain the posterior probabilities for token candidates, denoted as $P(u_t = v_i|u_{t-N-1:t-1})$, which represent the probabilities of token u_t being v_i, where v_i represents the i-th token in the vocabulary, given a specific history $u_{t-N-1:t-1}$ predicted by the language model.

An FNN LM uses a fixed window to collect fixed-length contexts. It is essentially a neural version of N-gram LMs. The FNN LM have several advantages over the N-gram LM by projecting tokens into continuous space. First, it can handle unseen N-grams by representing each token as an N-gram with a dense vector space. Second, it is storage-efficient since it does not need to count and store the transition probability of conventional N-gram models.

4.4 Recurrent Neural Network (RNN) Models

It is clearly insufficient to use the historical context in a fixed-length to predict the next token. In contrast to the limited historical context used in the N-gram, maximum entropy and FNN LMs, Recurrent Neural Network (RNN) LMs [89, 114, 115, 169, 210] can exploit arbitrarily long histories to predict the next token.

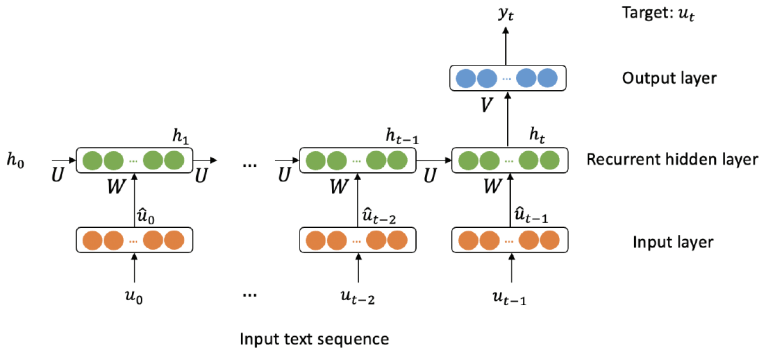

Figure 5: The structure of RNN LMs.

The structure of a vanilla RNN LM is shown in Figure 5. A token u_i in position i is first converted into a one-hot representation $\hat{u}_i$. Then, the recurrent hidden state, h_{i+1}, is computed using the previous hidden state, h_i, and the one-hot representation, $\hat{u}_i$, of token u_i as

$$h_{i+1} = f(W\hat{u}_i + Uh_i), \tag{12}$$

where $f(\cdot)$ is a non-linear activation function, W is the weight matrix of the connections from the input layer to the hidden layer, and U is the connection between the previous and current hidden layers, respectively. By iteratively computing the hidden states, RNN LMs can encode the historical context of varying length. Finally, the output layer gives the conditional probability of tokens $y_t = g(Vh_t)$, where V is the weight matrix connecting the hidden layer and the output layer and $g(\cdot)$ is the softmax activation function.

In theory, RNN LMs do not need the Markov assumption. They can use all preceding history to predict the next token. However, the inherent gradient vanishing problem of RNN hampers the learning of the model [58]. Since the gradient may become very small over a long distance, model weights are actually updated by the nearby context in practice. Generally, RNN LMs cannot learn the dependency between the current token and its far-away historical context. Although an attention mechanism can be introduced to RNNs to alleviate this problem [8, 35]. The inherent sequential nature of RNNs makes them less powerful than transformer-based LMs with a self-attention mechanism.

4.5 Transformers

The transformer architecture [179] can capture long-term dependencies and important sequence components by exploiting a self-attention mechanism.

Unlike the recurrent structure of RNNs, a transformer is easy to parallelize in both training and inference. Its structure is shown in Figure 6. It consists of an encoder and a decoder. Before being sent to the encoder, the input textual sequence is first converted to an embedding through an embedding layer plus positional embedding. Multi-head attention, which is an ensemble of multiple self-attention mechanisms, enables the transformer to capture more robust and diverse attention between tokens. The other parts in the transformer encoder include feed-forward layers, residual connections, and normalization layers. The difference between the transformer encoder and decoder is that the transformer decoder has an additional masked multi-head attention layer. The masking ensures the decoder can only access preceding tokens of the current one, which makes the decoder auto-regressive.

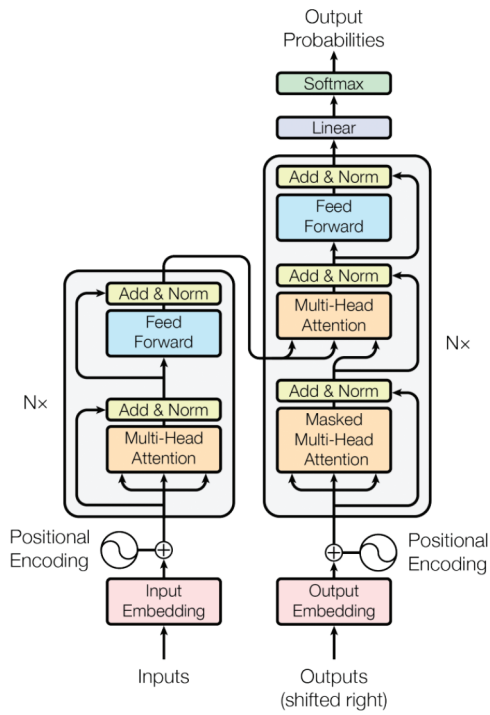

Figure 6: The structure of a transformer [179].

Based on different purposes, transformers have encoder-only, decoder-only, and encoder-decoder three variants as shown in Table 1 and Figure 7. Encoder-only models can access all positions given an input and utilize bi-directional contexts to predict tokens. They are suitable for tasks requiring understanding full sentences, such as text classification. Transformer decoder-only models can

Table 1: Transformer-based PLMs.

Encoder-only models (Bidirectional)	BERT [36] RoBERTa [107] ELECTRA [27]
Decoder-only models (Unidirectional)	PaLM [24] GPT-1, 2 and 3 [17, 133, 134] Transformer XL [31]
Encoder-Decoder models (Sequence to sequence)	BART [99] T5 [135]

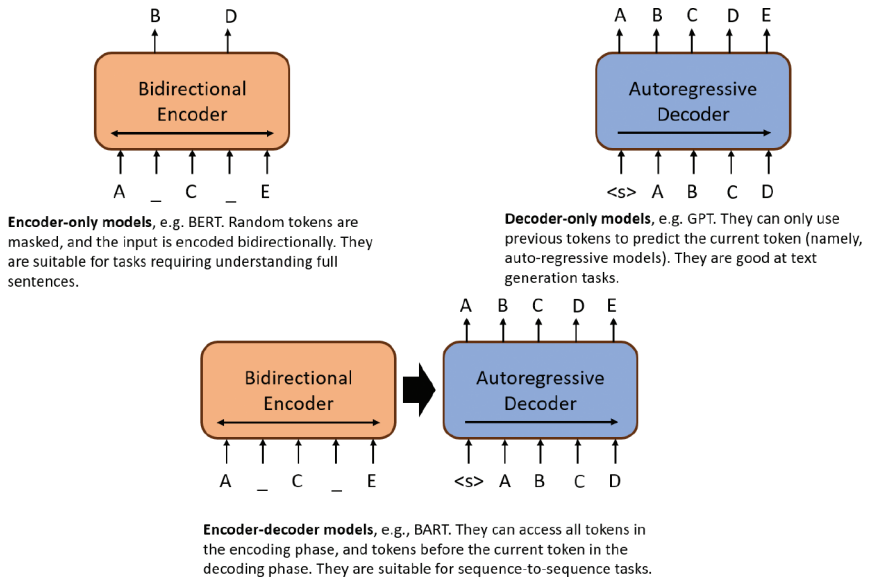

Figure 7: Illustration of different transformer models, where BERT is the encoder-only model, GPT is the decoder-only model, and BART is the encoder-decoder model.

only use previous tokens to predict the current token (namely, auto-regressive models). They are good at text generation tasks such as story generation. Transformer encoder-decoder models can access all tokens in the encoding phase, and tokens before the current token in the decoding phase. They are suitable for sequence-to-sequence tasks such as translation and summarization.

5 Pre-trained Language Models

Pre-trained language models (PLMs) are dominating in the NLP field nowadays. With the development of deep learning, the training and usage of PLMs

have changed a lot as compared with conventional statistical LMs. Before being applied to real-world tasks, PLMs are first pre-trained on massive collections of corpora so that they learn universal representations that carry both syntactic and semantic knowledge. After pre-training, PLMs are fine-tuned for downstream tasks so that the acquired knowledge can be transferred to different tasks. In the following, we first explain the pre-training objectives in Section 5.1 and then talk about how to adapt PLMs to various tasks of interest through fine-tuning in Section 5.2. It is also worthwhile to point out several good survey papers on PLMs, e.g., [55, 106, 132].

5.1 Pre-training

The most commonly used pre-training task is "missing token prediction". There are other pre-training tasks for different purposes, e.g., next-sentence prediction, which allows an LM to learn sentence relationships.

Token Prediction. Auto-regressive language LMs [17, 133, 134] are trained to predict the next token using previous tokens. While bidirectional LMs [36, 94, 107] mask a subset of tokens in a sample and learn to predict such masked tokens using the rest of the context. For the latter, the most popular objective is the masked language model (MLM) objective as proposed in BERT [36]. The MLM objective is the cross-entropy loss in predicting masked tokens. It randomly masks out 15% of the input tokens and then predicts the masked tokens. The number of masked tokens is set to 15% based on experimental verification. If the masking rate is too small, the model only learns from a limited number of masked tokens. On the other hand, if it is too large, there is not enough context to do reasonable predictions and models cannot learn well.

Other Pre-training Tasks. There are other pre-training tasks to make LMs learn better linguistic knowledge such as sentence relationships. For example, next sentence prediction is used as the pre-training task in BERT [36]. Next sentence prediction is formalized as a binary prediction task that decides whether two sentences are two consecutive sentences or not. In this way, a PLM can be used in downstream tasks that require the understanding of the relationship between two sentences, such as Question Answering (QA) and Natural Language Inference (NLI). Other pre-training objectives are adopted by BART [99]. They include token deletion, text infilling, sentence permutation, and document rotation to corrupt the original sequence for reconstruction. Shuffled tokens are used in T5 [135] to increase the robustness of the learned representation.

5.2 Fine-Tuning, Adapter Tuning and Prompt Tuning

PLMs learn non-task-specific language knowledge in the pre-training stage. Fine-tuning performs task-specific adaptations of the model so that they can

be applied to different downstream tasks. The model parameters are updated in the fine-tuning stage. One approach is to design task-specific heads based on different label spaces and losses in different downstream tasks, then update the entire model and task-specific heads. For instance, GPT [133] and BERT [36] added an extra linear output layer as task-specific heads in their original papers, and fine-tuned the entire set of parameters in the PLMs and the heads for various downstream tasks, such as natural language inference, question answering, semantic similarity, and text classification. To make the fine-tuning mechanism more parameter efficient, one can choose to only update certain layers of an LM and the task-specific heads.

Adapter tuning [60, 62, 129] is proposed to make fine-tuning even more parameter efficient compared with updating the last layers of a PLM only. It injects additional compact layers, calls adapters, into the original PLMs. Then, the new adapter layers are updated, while the parameters of the original PLMs are frozen during adapter tuning. In this way, the parameters of the original PLMs can be shared by different downstream tasks.

PLMs are pre-trained by one or several pre-training objectives and, then, applied to different downstream tasks. The gap between pre-training tasks and downstream task-specific fine-tuning can be substantial. Prompt-tuning [106] is used to discover the potential of PLMs by mimicking the pre-training objectives in the fine-tuning or inference stage. As PLMs get more powerful, they can handle various downstream tasks by seeing a few examples without any gradient updates or fine-tuning. This is achieved by prompt-based fine-tuning (or prompt-tuning in short).

The prompt can be divided into discrete prompts (also called hard prompts) and continuous prompts (also called soft prompts). A discrete prompt is a natural text template that could be manually designed by humans [17, 152, 153] or automatic methods [43, 130, 222]. On the contrary, continuous prompts [97, 102, 131, 221] are continuous vectors in the embedding space that do not correspond to real text. It sacrifices interpretability but relaxes the discrete prompt constraint in that prompts should be real texts.

Figure 8 shows an example of the pre-training task, fine-tuning and discrete prompt-tuning of MLMs. In the pre-training, MLMs are trained to predict masked tokens. Assuming that the downstream task is the sentiment analysis of the movie review. In standard fine-tuning, we train a new head on the top of a PLM and predict the sentiment labels. The original input appended with a designed prompt, say, 'It was', is sent to the PLM. The PLM has to assign probabilities to designed answers, which can be 'great' or 'terrible'. If the probability of 'great' is higher, then the label of the input will be positive and vice versa. In this way, prompt-tuning converts a distinct downstream task to the token prediction task to narrow the gap between the pre-training and fine-tuning stages.

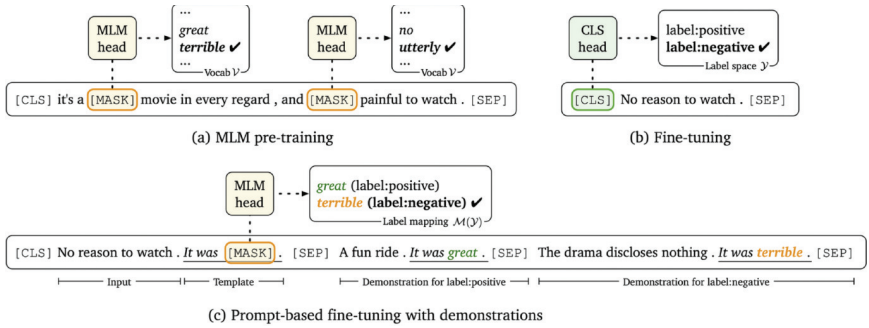

Figure 8: An illustration of (a) LM pre-training, (b) standard fine-tuning, and (c) discrete prompt-based fine-tuning (or prompt-tuning) [43].

6 Model Evaluation

There are two LM evaluation types: intrinsic evaluation and extrinsic evaluation. The intrinsic evaluation examines the internal properties of an LM while the extrinsic evaluation studies its performance in downstream tasks.

6.1 Intrinsic Evaluation

Auto-regressive LM. LMs estimate the probability of text sequences. A good LM assigns higher probabilities to natural text sequences and lower ones to unreal or random text sequences. The perplexity is a common evaluation metric for this purpose. Given a testing text sequence, the perplexity, denoted by PPL, is defined as the inverse probability of the sequence normalized by the number of tokens. Mathematically, we have

$$PPL(S) = \sqrt[N]{\frac{1}{(P(u_1 u_2 \ldots u_N)}}, \tag{13}$$

where $S = u_1 u_2 \ldots u_N$ is a testing text sequence. The perplexity can be rewritten in form of

$$PPL(S) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(u_i | u_1 \ldots u_{i-1})}}. \tag{14}$$

A good LM should maximize the text set probability. It is equivalent to minimizing the perplexity. The lower the perplexity, the better the LM.

Bidirectional Language Model. To calculate the inverse probability in Equation (13), the auto-regressive LMs can use a sequence of conditional

probabilities. However, this approach does not work for bidirectional LMs (or masked LMs). Several intrinsic evaluation metrics have been proposed for bidirectional LMs. The pseudo-log-likelihood score (PLL) [183] is defined as

$$PLL(S) = \sum_{i=1}^{|S|} \log P(u_i|S_{\setminus i}), \tag{15}$$

where $\log P(u_i|S_{\setminus i})$ is the conditional probability of token u_i in sentence S with all remaining tokens. Instead of maximizing the joint probability of the entire text sequence, a good bidirectional LM should maximize the probability of each token in the text sequence given other tokens. Based on PLLs, the pseudo-Perplexity (PPPL) for corpora C is defined as [147]

$$PPPL(C) = exp(-\frac{1}{N} \sum_{S \in C} PLL(S)). \tag{16}$$

Both PLL and PPPL provide effective means to measure the naturalness of sentences for a bidirectional LM. For example, it was shown in [147] that PLL and PPPL correlate well with the performance of an LM on downstream tasks, such as automatic speech recognition and machine translation.

6.2 Extrinsic Evaluation

Any downstream task of LMs can be used for extrinsic evaluation. There are several common downstream tasks selected as extrinsic evaluation benchmarks. Two popular ones are GLUE (General Language Understanding Evaluation) [185] and SuperGLUE [184]. GLU is an evaluation benchmark for natural language understanding. It contains single-sentence tasks, similarity and paraphrase tasks, and inference tasks. SuperGLUE is an enhanced version of GLUE. It includes a new set of more challenging language understanding tasks, more diverse task formats, improved resources, and a public leaderboard.

6.3 Relation between Intrinsic and Extrinsic Evaluations

If an LM achieves a lower perplexity, does that mean it can also perform well on downstream tasks? In other words, is there any correlation between pre-training tasks (based on token prediction) and the downstream tasks? There are many empirical studies on this question but few theoretical studies.

Empirical Studies. Researchers design experiments to understand what kind of knowledge is learned by an LM from the pre-training tasks. Examples include [47, 57, 85, 139, 171, 172]. They use part-of-speech tagging, constituent labeling, and dependency labeling to measure the degree of syntactic knowledge

learning, and named entity labeling, semantic role labeling, and semantic proto-role for testing semantic knowledge. Empirical studies show that pre-training tasks help LMs learn the linguistic knowledge such as the grammar [85] and the semantic role [139]. However, these experimental results can only be used as evidence supporting that the token prediction tasks benefit downstream tasks. They cannot explain the underlying mechanism.

Theoretical Studies. Some researchers attempt to build the connection between LM's perplexities and its performance on downstream tasks mathematically. The text classification tasks were studied in [151]. They first hypothesized and verified that text classification tasks can be reformulated as sentence completion tasks. Since the LM pre-training task is essentially a sentence completion task, it does help the text classification downstream task. Then, they quantified the connection mathematically and showed that the features from LMs that achieve ϵ-optimal in log-perplexity can linearly solve text classification tasks with $O(\sqrt{\epsilon})$ error. An underlying generative model was utilized in [200] to show the relationship between the pre-training tasks and the downstream tasks. Current theoretical studies are limited in the sense that only a specific downstream task (say, the text classification task) is considered and the proof holds under certain conditions.

6.4 Beyond Single Metric for LM Evaluation

Except for the evaluation of LM's performance on standard evaluation test sets, the LM performance on other aspects is also important in real-world applications, such as efficiency [10, 166, 173, 202], bias [1, 13, 112, 119], robustness [48, 77, 122, 125, 145, 192, 214], explainability [223], and logical consistency [137]. In this section, we discuss evaluations on efficiency, bias, and robustness to provide a holistic review of evaluation aspects.

Efficiency of LMs can be evaluated in several aspects, such as inference time, computational complexity, energy consumption, model size, and training data size. Some work [166, 173, 196, 202] calculated the computational complexity, approximate financial, and environmental costs of training PLMs. They also suggested practical steps to reduce expenses in NLP research and applications. Discussion on the model size of recently developed PLMs was given in [10]. In Section 8 of this paper, we also discussed several methods to achieve efficient LMs. Table 2 shows the number of parameters, training data, cost, and time of recently developed LMs.

Bias in NLP refers to systematic prejudices of models resulting from erroneous assumptions, such as racism, sexism, and ableism. Bias is reflected in PLMs since they are trained on a large volume of real word data. Several studies have examined bias in PLMs. The Sentence Encoder Association Test (SEAT) was proposed in [112] to investigate bias in BERT [36]. A dataset was created in [119] to measure bias against gender, profession, race, and religion

across multiple PLMs, including BERT [36], RoBERTa [107], XLNet [211] and GPT-2 [134]. It was demonstrated in [1] that GPT-3 [17] consistently exhibits a significant anti-Muslim bias in various tasks. The work in [13] surveyed 146 papers on bias in NLP and made recommendations for analyzing bias in NLP systems.

Robustness of LMs refers to their capacity to perform effectively and consistently when confronted with input variations (e.g., typos and misspellings) that should not affect the system's output. In other words, a robust LM should not be easily fooled by adversarial text. Recent studies[77, 122, 214] created a set of character or word level perturbations to simulate various types of noise that LMs may encounter in real-world scenarios. They examined robustness of recently developed PLMs, including BERT, RoBERTa and XLNets. The results suggest that input perturbations, even minor alterations, can harm the performance of these LMs. In addition, Robustness Gym [48], WildNLP [145], and TextFlint [192] are tools designed for robustness evaluation.

7 Language Models in Text Generation

One of the most important applications of LMs is text generation, which aims to generate sequences of words based on the input data. There are many text generation tasks because of different purposes and inputs. For example, the automatic speech recognition (ASR) task demands that the input be a speech sequence while the output be the corresponding text sequence. The machine translation task generates the translated text sequence based on the input text sequence and the target language. Story Generation is a topic-to-text generation task. In this section, we introduce common techniques used in text generation and then explain how LMs can be applied in each of the representative tasks.

7.1 Decoding Methods

Decoding decides the next output linguistic unit to generate text. A good decoding method should generate coherent continuation given a context. As LMs get more sophisticated, decoding methods have played an increasingly important role. As shown in Figure 9, deficient decoding methods lead to bad generated texts even with a powerful LM. There are two main decoding methods for text generation.

Maximization-based decoding. This is the most commonly used decoding objective. Assuming that the model assigns a higher probability to a higher quality text which is closer to the ground truth written by humans, the maximization-based decoding strategy searches for tokens with the highest probability as the generated text. Greedy search [206, 220] chooses the token

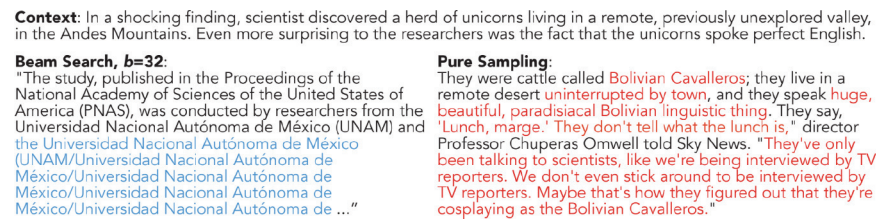

Figure 9: Comparison of texts generated by the powerful GPT-2 large language model (LLM) using Beam search (left) and pure sampling decoding (right). Beam search yields degenerate repetition (in blue) while pure sampling results in incoherent gibberish (in red) [59].

with the highest probability as the next token in a greedy manner. Beam search [92, 100, 181] keeps a certain number of most likely tokens at each time step and selects the generated token sequences with the overall highest probability eventually. It avoids missing reasonable tokens that do not have the highest probability. Trainable decoding algorithms have been proposed recently. Trainable greedy decoding [50] is a neural-based solution that works as part of a neural machine translation decoder. It utilizes reinforcement learning to find a translation that maximizes a decoding objective.

Sampling-based decoding. It chooses the next token from a set of sampled tokens. Because maximization-based decoding depends highly on the underlying model probabilities and suffers from producing degenerate repetition, sampling-based decoding increases the diversity of generated texts by random sampling. However, the simple pure sampling may choose a token with low probability (from an unreliable tail distribution) as the next generated token. As a result, the generated text could be unrelated to the prefix, leading to incoherent gibberish. Top-k sampling [39] and Nucleus sampling [59] have recently been proposed to address this problem. Both Top-k sampling and Nucleus sampling sample from truncated LM distributions (i.e., sampling from the most probable tokens). Diverse Beam search [100] is a trainable sampling-based (stochastic) decoding algorithm based on the Beam search. It uses reinforcement learning to determine the beam diversity parameters for different inputs or tasks.

7.2 Dialogue Systems

A dialogue system aims at simulating human responses when conversing with human users. Recent dialogue systems such as ChatGPT[4] and LaMDA [174] have attracted a lot of attention in the generative AI field because of their superior performance as interactive chatbot systems. Dialogue systems can be

[4] https://openai.com/blog/chatgpt/.

categorized into task-oriented systems and open-domain systems. The former is designed for specific tasks such as customer service for online shopping. The latter is also known as chatbots [121]. Most modern dialogue systems are fine-tuned versions of generative LMs. Taking ChatGPT as an example, ChatGPT is built based on a generative LM, GPT-3 [17] with 175 billion parameters. It is further fine-tuned by supervised learning and reinforcement learning on labeled data.

LMs play an important role in dialogue systems, especially in their natural language understanding (NLU) and natural language generation (NLG) components [189, 190]. NLU is responsible for understanding and recognizing users' intent. Nowadays, for encoder-decoder PLMs', the encoders provide informative representations for NLU, while the associated decoders are responsible for generating an appropriate response. The latter involves constructing the response text, selecting appropriate words, and determining the correct phrasing and tone. The effectiveness of representations of PLMs was examined in [203] for dialogue tasks. The evaluation PLM targets included BERT [36] and GPT-2 [134]. The few-shot capability of PLMs in dialogue tasks such as NLU and NLG was evaluated in [111]. Overall, LMs in dialogue systems play a key role in understanding users' input and generating appropriate and natural responses.

7.3 Automatic Speech Recognition

Automatic speech recognition (ASR) is a speech-to-text generation task that aims to transform raw audio input into the corresponding text sequence. The LM plays an essential role in an ASR system. First, it helps solve acoustically ambiguous utterances. Second, it can lower the computational cost by constraining the search space in a set of words of higher probability. Conventional ASR systems contain two independent models, an acoustic model and a language model, which are related by

$$P(word|sound) \propto P(sound|word)P(word). \tag{17}$$

The acoustic model is conditioned on phones $P(sound|word)$ while the LM gives the word distribution denoted by $P(word)$. LMs help search the word hypotheses during recognition. Different types of LMs have been explored in ASR, such as N-gram [71, 163], FFNN [7], RNN [6, 66] and Transformer [161]

With the development of deep learning techniques, end-to-end (E2E) ASR systems have emerged as the dominant approach in this field nowadays. E2E ASR systems do not train the acoustic model and the language model independently but use a single-network architecture. For example, the Listen, Attend, and Spell (LAS) model [18] contains an encoder, a decoder, and an attention network, which are trained jointly to predict the output text sequence. The LM

component in the E2E ASR system is implicitly learned from the transcribed speech data. To address the challenge of limited transcribed speech data for LM's training, one solution is to integrate external language models trained on extensive text corpora using LM integration [81, 176]. Shallow fusion [23, 53, 113] considers log-linear interpolation between the scores from an E2E ASR model and an external LM at the decoding stage. Deep fusion [53] integrates an external LM and the E2E ASR model by fusing their hidden states. Unlike shallow fusion and deep fusion, where the E2E ASR model and the external LM are separately trained, cold fusion [164] and component fusion [159] train the E2E ASR model and the external LM jointly.

7.4 Machine Translation

Machine translation is a text-to-text generation task where the text in the source language is translated into that of the target language. LMs adopted by machine translation are conditioned on the source sentence and the previous partial translation. The E2E machine translation models become prevailing nowadays. The language model is implicitly learned through E2E training. Recently, transformer-based models achieved great success in machine translation [179, 191]. Similar to ASR advancements, an external LM trained by extensive monolingual corpora can be incorporated into an E2E machine translation model through LM integration techniques [53]. Furthermore, many PLMs have shown their few-shot or zero-shot ability on machine translation [17, 24] although they have never been explicitly trained on translation parallel data between the source and the target languages.

7.5 Detection of Generated texts

As the performance of LMs gets closer to or even outperforms humans, the misuse of LMs, such as fake news and fake product reviews generation, has become a serious problem. The ability to detect machine-generated texts is important. There are two types of detection problems: 1) human written vs. machine generated, and 2) inveracious vs. veracious. Most datasets, e.g., [38, 178, 201], are collected for the first type. Problems of the second type are much harder than those of the first type [175] since one needs to connect generated text to the fact, which requires a high-level knowledge reasoning capability.

Two common approaches to detecting machine-generated text are reviewed below. One is to exploit the probability distribution of LMs [46, 68]. If the probability distribution of a text sequence is closer to that of human-written texts as compared with known machine-generated texts, the text sequence is classified as human-written. The other is to train classifiers with supervised learning [178, 215]. It converts the distribution to a supervised

binary classification task. For more details on the detection of machine-generated texts, readers are referred to two survey papers [70, 165].

8 Efficient Models

As recent PLMs get more powerful, their model size, training cost, and demand for training data increase tremendously. They need high computational resources and energy consumption, limiting their real-world applications. Table 2 shows the model size, training data, cost, and time of recently developed LMs. This issue is a concern to many people and the construction of efficient LMs has received attention.

Table 2: Table of the number of parameters, training data, cost, and time of several large LMs, where blank cells indicate that the data are not available. The sources are cited if the data are not obtained from the original work.

Model	Year	Number of Parameters	Training data	Training cost	Training time
BERT-Large	2018	340M	3.3B words	$7,000[5]	64 TPU chips 4 days
XLNet-Lagre	2019	340M	32.9B tokens	$245,000[5]	512 TPU v3 chips 5.5 days
GPT-2	2019	1.5B	8 million web pages	$12,902–$43,008 [166]	32 TPU v3 chip 168 hours
Megatron-LM	2019	8.3B	174 GB of text data		512 GPUs 2 days per epoch
T5	2019	11B	745GB of text data	Over $1.3 million [160]	
Turing-NLG	2020	17B			
GPT-3	2020	175B	570GB of text data	Over $4.6 million[6]	1024 A100 GPUs 34 days [120]
Megatron-Turing NLG	2022	530B	270B tokens		2K A100 GPUs 3 months[7]

8.1 Data Usage

Pre-training Data Size. A critical question for PLM training is how much data is needed. The effect of the pre-training data size on the RoBERTa model was studied in [218]. The learning curves of four model performance measures as a function of the pre-training dataset size are shown in Figure 10. When the

[5] https://syncedreview.com/2019/06/27/the-staggering-cost-of-training-sota-ai-models/.
[6] https://lambdalabs.com/blog/demystifying-gpt-3.
[7] https://www.deepspeed.ai/.

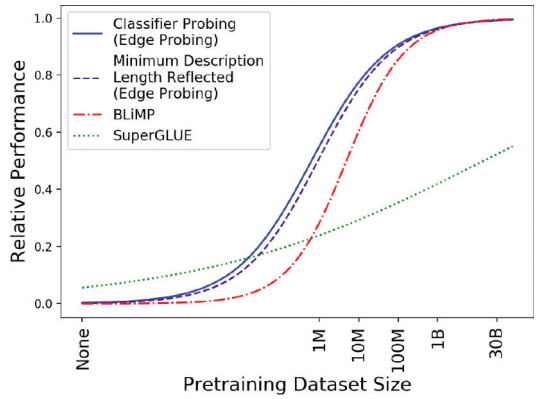

Figure 10: The performance curves as functions of the pre-training dataset size, where the classifier probing measures the quality of the syntactic and semantic features, the minimum description length probing quantifies the accessibility of these features, the BLiMP curve measures the model's knowledge of various syntactic phenomena, and the superGLUE measures the capability of handling NLU tasks [218].

data size ranges between 100M and 1B words, three learning curves gradually level off and it implies that LMs encode most syntactic and semantic features. However, a much larger quantity of data is needed for LMs to acquire enough common-sense knowledge and other skills to achieve better performance on downstream NLU tasks.

Efficient Pre-Training. Several methods have been proposed to use the pre-training data more efficiently. In the pre-training of masked LMs, a certain percentage of tokens are masked and need to be inferred by context. This approach incurs a substantial amount of computational cost because the network only learns from a certain percentage of tokens which are masked. To enhance training efficiency, the work in [27] uses "replaced token detection" (rather than "masked token prediction") as the pre-training task. As shown in Figure 11, a generator is trained to perform the masked LM and predicts the masked tokens. Then, the main model works as a discriminator, called ELECTRA, which learns to decide the original or replaced tokens. In this way, pre-training tasks are conducted on all tokens instead of a small subset of masked tokens. Learning from all input positions causes ELECTRA to train much faster than BERT which adopts masked token prediction. Besides, ELECTRA achieves higher accuracy on downstream tasks when it is fully trained. Later, a new pre-training task using an energy-based model, which is closely related to ELECTRA, is proposed in [26].

Bridging Pre-training and Downstream Tasks. A typical pre-training task is token prediction, which often has a large gap with downstream tasks. To mitigate the gap between pre-training and downstream tasks, prompt tuning

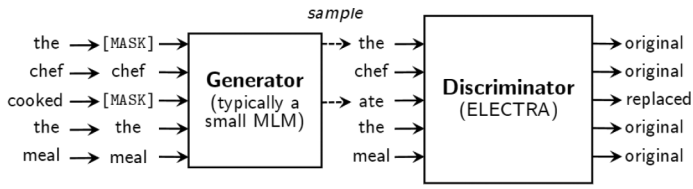

Figure 11: The structure of ELECTRA (Efficiently Learning an Encoder that Classifier Token Replacements Accurately) [27].

has been studied in [43, 134, 152, 162]. As illustrated in Figure 8, the head is trained to predict the masked tokens in masked LMs. For the downstream sentiment analysis task, the head is trained to predict the positive or the negative label in traditional fine-tuning. A template (e.g., 'It was') and its expected text responses (e.g., 'great' and 'terrible') are used in prompt tuning. In this way, pre-training and prompt tuning share the same "token prediction" objective.

8.2 Model Size

Besides improving training efficiency, efficient LMs focus on the design of models of smaller sizes. Many methods are investigated to reduce the model size so that the model can be implemented on mobile or edge devices with limited computing resources. Model compression is a widely studied topic. Compression methods first train a large LM and then compress it into a target size. Examples include model pruning [54, 182, 197], knowledge distillation [76, 148, 177], low rank matrix approximation [61, 110, 210], and parameter sharing [30, 33, 94, 143].

8.3 Inference Latency

Inference efficiency is important to an LM, particularly in real-time applications. A model of a smaller size generally has faster inference speed under the same setting. Knowledge distillation, pruning, and low rank matrix approximation can be employed to achieve faster inference time while reducing the model size. For instance, DistilBERT [148], which is a distilled version of BERT, has demonstrated a 60% improvement in the inference speed compared to the original model. More than 2x speed-up in inference is achieved in [197] by pruning PLMs.

Fast inference speed can also be achieved by fast decoding methods. Non-autoregressive generation (NAG) models [49, 101, 167] predict each token simultaneously. They have a faster inference speed than autoregressive models due to parallel computation. On the other hand, the performance of NAG models is generally worse than autoregressive models since they do not consider the forward or backward dependency between tokens in the output text.

9 Future Research Directions

In this section, we describe several promising future research directions in language modeling.

9.1 Integration of LMs and KGs

Knowledge Graph (KG) provides a structured representation of human knowledge [45, 195]. It has been widely used in many NLP applications, such as question answering [65] and text summarization [64], because of its capability to represent relationships between entities. There is a growing interest in evaluating the knowledge learned in PLMs [127, 128], where the relationship between different semantic units is captured in the embedding space and the self-attention layers. Several ideas are proposed in [4, 56, 193, 204, 212, 217, 219] to leverage KGs for LM training. As a result, the knowledge learned in the models can be greatly improved. Thus, it is worth careful investigation of integrating KGs with LMs and understanding how they interact with each other.

It appears that KG can serve as an information database to be queried by LMs. LMs are powerful in natural language understanding and generation while KGs can organize and store the knowledge information extracted from the training corpus. In other words, we may decompose knowledge sources into semantic and syntactic two components, which can be handled by KGs and LMs, respectively.

Specifically, most reasoning is handled by KGs so that predictions are fact-based and explainable. On the other hand, LM serves as an interface to understand and interpret the language input and improve fluency, comprehensiveness, conciseness, etc., of the language output. Similar concepts were proposed in [63, 213]. In the training phase, a KG is constructed based on the information extracted from the training corpus, and an LM can be trained simultaneously. In the inference phase, an LM can serve as an interface between humans and the knowledge database represented in the form of KGs. There are advantages to assigning semantic and syntactic processing tasks to KGs and LMs, respectively. For example, the decoupling facilitates incremental learning, allows a smaller model size, and improves interpretability. They will be further elaborated on below.

9.2 Incremental Learning

Incremental learning aims to incorporate new information without re-training existing models entirely. The problem of catastrophic forgetting associated with neural network models was pointed out in [41]. That is, the information that has already been learned by a model can be gradually forgotten when training with new information. This problem is particularly critical to large LMs since

new information keeps arriving. A solution to catastrophic forgetting was proposed in [87]. It attempts to remember prior important tasks by slowing down learning on weights that are more relevant to them. However, it is difficult to define important tasks in LMs. In addition, re-training a large LM with both old and new data is too expensive. Lifelong learning of LMs [78, 95, 109] is another solution to accommodate new data to update the knowledge in LMs. It is worth further exploration.

The importance of developing a satisfactory solution to incremental learning for LMs cannot be over-emphasized. Incremental learning is challenging for neural networks. Yet, it is easy for KGs to add new data to (or remove old data from) an existing database by adding or removing factual triples [188]. Clearly, the current information in the KGs will not be overwritten by newly collected data. The information in the database is updated incrementally. To this end, the integration of KGs and LMs provides an excellent solution that meets the need for incremental learning.

9.3 Lightweight Models

As mentioned in Section 8, PLMs get more powerful at the expense of huge computational resources and energy consumption. The cost issue has to be faced seriously in the development of large LMs (LLMs). Besides, LLMs are unfriendly to our environment due to their high carbon footprint. Green Learning (GL) targets learning solutions with low carbon footprint. The design of lightweight models of smaller sizes and lower computational complexity without sacrificing performance has received more attention in recent years [93, 155, 194, 205]. The design of green LMs is an important topic worth serious investigation.

Current PLMs are data-driven models that use neural architectures to learn generic language knowledge from a large amount of data. Efforts have been made in the development of lightweight LMs. Model compression is one of the popular approaches to obtaining a small LM. Examples include knowledge distillation or pruning [103]. However, this methodology appears to be a detour since it trains large models and then shrinks their sizes by compression. Instead, we may incorporate the linguistic information and the domain knowledge to offer a more direct way to reduce the model size and the amount of training data.

9.4 Universal versus Domain-Specific Models

A universal LM is developed to handle tasks in the general domain. For example, ChatGPT is a universal dialogue LM pre-trained on multilingual and general domain corpora. It can converse on open-domain topics in multiple languages. In contrast, domain-specific LMs [51, 105, 108, 216] are designed to deal with domain-specific tasks, e.g., biomedicine, economics, musicology, etc.

A universal LM demands a huge model size, a large number of training examples, and a tremendous amount of computational resources. Based on the scaling law of neural language models [82], the inference performance scales as a power-law with the model size, the dataset size, and the amount of computing used for training. So far, the largest PLM contains 540-billion parameters [24]. Despite the superior performance and the flexibility to adapt to multiple tasks, we may wonder whether a huge universal LM is cost-effective.

For domain-specific LMs, the amount of training data in need is significantly lower. It was believed that the general domain PLMs benefit the training of domain-specific LMs. However, it is reported in [51] that domain-specific LMs, which were pre-trained from scratch on in-domain data, can provide a solid foundation for biomedical NLP. In other words, training a domain-specific LM may not need a huge amount of general corpora and labeled data. Domain-specific LMs to be deployed on task-specific scenarios with less training and inference efforts expect to receive more attention in the future.

9.5 Interpretable Models

Although deep-learning-based LMs are dominating the NLP field, they are inherently black-box methods without mathematical transparency. Its interpretability is of concern. Efforts have been made to explain the black-box LMs. As mentioned in Section 6.3, empirical studies are conducted to understand what PLMs have learned through experimental design. However, the progress in this direction may offer insights but not a satisfactory and clean answer. Providing theoretical explanations or establishing explainable LMs is still a challenging and open issue. A direction to interpretability is to design an interpretable learning model from scratch. For example, we may incorporate KGs with LMs. KG is known to be capable of improving the interpretability and transparency of the system in many reasoning tasks such as information retrieval [37] and recommendation systems [209]. For example, reasoning paths and data sources can be provided with predictions when KGs are incorporated for reasoning. It is challenging for LMs to do so. It is critical to develop an interpretable LM to avoid its hallucination in natural language generation [75].

9.6 Machine Generated Text Detection

The most common application of LMs is text generation. As generative LM's performance gets closer to or even outperforms humans, these LMs can be used for malicious purposes such as academic dishonesty, spamming, targeted bot attacks, and fake news/reviews generation. How to determine whether a text is generated by LMs or written by humans is a big challenge nowadays. A high-performance machine-generated text classifier can only serve as a reference in real-world applications, since it has false positives (i.e., human-written texts

classified as machine-generated) and false negatives (i.e., machine-generated texts classified as human-written). In addition, people may be even more interested in detecting veracious and unveracious texts. They care more about whether the text is true or not. Detecting disinformation could be more difficult than detecting machine/human-generated text without assessing the factuality. Additionally, the factuality may change as time goes by. It is critical to our society in developing effective tools to identify malicious usages of generative LMs.

10 Conclusion

A comprehensive overview of CLMs and their successors, PLMs, was presented in this paper and a wide range of topics was covered. First, different levels of linguistic units were introduced and how linguistic unit prediction is used to train language models was examined. Second, tokenization methods adopted by language models were discussed. Third, language model architectures and the training paradigm of PLMs were reviewed. Fourth, we studied the evaluation and applications of language models. Especially, several applications in the context of text generation were detailed. Finally, several future research directions were pointed out. The need for explainable, reliable, domain-specific, and lightweight language models was emphasized.

References

[1] A. Abid, M. Farooqi, and J. Zou, "Persistent anti-muslim bias in large language models", in *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, 2021, 298–306.

[2] M. Ablimit, G. Neubig, M. Mimura, S. Mori, T. Kawahara, and A. Hamdulla, "Uyghur morpheme-based language models and ASR", in *IEEE 10th INTERNATIONAL CONFERENCE ON SIGNAL PROCESSING PROCEEDINGS*, IEEE, 2010, 581–4.

[3] M. Afify, O. Siohan, and R. Sarikaya, "Gaussian mixture language models for speech recognition", in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, Vol. 4, IEEE, 2007, IV–29.

[4] O. Agarwal, H. Ge, S. Shakeri, and R. Al-Rfou, "Knowledge Graph Based Synthetic Corpus Generation for Knowledge-Enhanced Language Model Pre-training", in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online: Association for Computational Linguistics, June 2021, 3554–65, DOI: 10.18653/v1/2021.naacl-main.278, https://aclanthology.org/2021.naacl-main.278.

[5] E. Arisoy, T. N. Sainath, B. Kingsbury, and B. Ramabhadran, "Deep neural network language models", in *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, 2012, 20–8.

[6] E. Arisoy, A. Sethy, B. Ramabhadran, and S. Chen, "Bidirectional recurrent neural network language models for automatic speech recognition", in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2015, 5421–5.

[7] E. Arısoy, S. F. Chen, B. Ramabhadran, and A. Sethy, "Converting neural network language models into back-off language models for efficient decoding in automatic speech recognition", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(1), 2013, 184–92.

[8] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate", *arXiv preprint arXiv:1409.0473*, 2014.

[9] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition", *IEEE transactions on pattern analysis and machine intelligence*, (2), 1983, 179–90.

[10] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?", in *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, 2021, 610–23.

[11] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model", *Advances in neural information processing systems*, 13, 2000.

[12] A. Berger, S. A. Della Pietra, and V. J. Della Pietra, "A maximum entropy approach to natural language processing", *Computational linguistics*, 22(1), 1996, 39–71.

[13] S. L. Blodgett, S. Barocas, H. Daumé III, and H. Wallach, "Language (Technology) is Power: A Critical Survey of "Bias" in NLP", in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, 5454–76.

[14] K. Bostrom and G. Durrett, "Byte Pair Encoding is Suboptimal for Language Model Pretraining", in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, 4617–24.

[15] P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. Lafferty, R. L. Mercer, and P. S. Roossin, "A statistical approach to machine translation", *Computational linguistics*, 16(2), 1990, 79–85.

[16] P. F. Brown, V. J. Della Pietra, P. V. Desouza, J. C. Lai, and R. L. Mercer, "Class-based n-gram models of natural language", *Computational linguistics*, 18(4), 1992, 467–80.

[17] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners", *Advances in neural information processing systems*, 33, 2020, 1877–901.

[18] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition", in *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, IEEE, 2016, 4960–4.

[19] C. Chelba and F. Jelinek, "Exploiting syntactic structure for language modeling", in *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, 1998, 225–31.

[20] C. Chelba and F. Jelinek, "Structured language modeling", *Computer Speech & Language*, 14(4), 2000, 283–332.

[21] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling", *Computer Speech & Language*, 13(4), 1999, 359–94.

[22] S. F. Chen and R. Rosenfeld, "Efficient sampling and feature selection in whole sentence maximum entropy language models", in *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No. 99CH36258)*, Vol. 1, IEEE, 1999, 549–52.

[23] J. Chorowski and N. Jaitly, "Towards Better Decoding and Language Model Integration in Sequence to Sequence Models", *Proc. Interspeech 2017*, 2017, 523–7.

[24] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, *et al.*, "Palm: Scaling language modeling with pathways", *arXiv preprint arXiv:2204.02311*, 2022.

[25] K. W. Church and W. A. Gale, "A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams", *Computer Speech & Language*, 5(1), 1991, 19–54.

[26] K. Clark, M.-T. Luong, Q. Le, and C. D. Manning, "Pre-Training Transformers as Energy-Based Cloze Models", in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, 285–94.

[27] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "Electra: Pre-training text encoders as discriminators rather than generators", *arXiv preprint arXiv:2003.10555*, 2020.

[28] M. Creutz, T. Hirsimäki, M. Kurimo, A. Puurula, J. Pylkkönen, V. Siivola, M. Varjokallio, E. Arisoy, M. Saraçlar, and A. Stolcke, "Morph-based speech recognition and modeling of out-of-vocabulary words

across languages", *ACM Transactions on Speech and Language Processing (TSLP)*, 5(1), 2007, 1–29.

[29] M. Creutz and K. Lagus, *Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0*, Helsinki University of Technology Helsinki, 2005.

[30] R. Dabre and A. Fujita, "Recurrent stacking of layers for compact neural machine translation models", in *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33, No. 01, 2019, 6292–9.

[31] Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-XL: Attentive Language Models beyond a Fixed-Length Context", in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, 2978–88.

[32] J. N. Darroch and D. Ratcliff, "Generalized iterative scaling for log-linear models", *The annals of mathematical statistics*, 1972, 1470–80.

[33] M. Dehghani, S. Gouws, O. Vinyals, J. Uszkoreit, and Ł. Kaiser, "Universal transformers", *arXiv preprint arXiv:1807.03819*, 2018.

[34] S. A. Della Pietra, V. J. Della Pietra, R. L. Mercer, and S. Roukos, "Adaptive language modeling using minimum discriminant estimation", in *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*, 1992.

[35] H. Deng, L. Zhang, and L. Wang, "Global context-dependent recurrent neural network language model with sparse feature learning", *Neural Computing and Applications*, 31(2), 2019, 999–1011.

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

[37] L. Dietz, C. Xiong, J. Dalton, and E. Meij, "The Second Workshop on Knowledge Graphs and Semantics for Text Retrieval, Analysis, and Understanding (KG4IR)", in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, 1423–6.

[38] T. Fagni, F. Falchi, M. Gambini, A. Martella, and M. Tesconi, "TweepFake: About detecting deepfake tweets", *Plos one*, 16(5), 2021, e0251415.

[39] A. Fan, M. Lewis, and Y. Dauphin, "Hierarchical Neural Story Generation", in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, 889–98.

[40] M. Federico, "Bayesian estimation methods for n-gram language model adaptation", in *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP'96*, Vol. 1, IEEE, 1996, 240–3.

[41] R. M. French, "Catastrophic forgetting in connectionist networks", *Trends in cognitive sciences*, 3(4), 1999, 128–35.

[42] P. Gage, "A new algorithm for data compression", *C Users Journal*, 12(2), 1994, 23–38.

[43] T. Gao, A. Fisch, and D. Chen, "Making Pre-trained Language Models Better Few-shot Learners", in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, 3816–30.

[44] T. Gao, X. Yao, and D. Chen, "SimCSE: Simple Contrastive Learning of Sentence Embeddings", in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, 6894–910.

[45] X. Ge, Y.-C. Wang, B. Wang, and C.-C. J. Kuo, "Compounde: Knowledge graph embedding with translation, rotation and scaling compound operations", *arXiv preprint arXiv:2207.05324*, 2022.

[46] S. Gehrmann, H. Strobelt, and A. M. Rush, "GLTR: Statistical Detection and Visualization of Generated Text", in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2019, 111–6.

[47] M. Giulianelli, J. Harding, F. Mohnert, D. Hupkes, and W. Zuidema, "Under the Hood: Using Diagnostic Classifiers to Investigate and Improve how Language Models Track Agreement Information", in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2018, 240–8.

[48] K. Goel, N. F. Rajani, J. Vig, Z. Taschdjian, M. Bansal, and C. Ré, "Robustness Gym: Unifying the NLP Evaluation Landscape", in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, 2021, 42–55.

[49] J. Gu, J. Bradbury, C. Xiong, V. Li, and R. Socher, "Non-autoregressive neural machine translation", in *International Conference on Learning Representations (ICLR)*, 2018.

[50] J. Gu, K. Cho, and V. O. Li, "Trainable Greedy Decoding for Neural Machine Translation", in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, 1968–78.

[51] Y. Gu, R. Tinn, H. Cheng, M. Lucas, N. Usuyama, X. Liu, T. Naumann, J. Gao, and H. Poon, "Domain-specific language model pretraining for biomedical natural language processing", *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1), 2021, 1–23.

[52] J. Gubbins and A. Vlachos, "Dependency language models for sentence completion", in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, 1405–10.

[53] C. Gulcehre, O. Firat, K. Xu, K. Cho, L. Barrault, H.-C. Lin, F. Bougares, H. Schwenk, and Y. Bengio, "On using monolingual corpora in neural machine translation", *arXiv preprint arXiv:1503.03535*, 2015.

[54] D. Guo, A. M. Rush, and Y. Kim, "Parameter-Efficient Transfer Learning with Diff Pruning", in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, 4884–96.

[55] X. Han, Z. Zhang, N. Ding, Y. Gu, X. Liu, Y. Huo, J. Qiu, Y. Yao, A. Zhang, L. Zhang, *et al.*, "Pre-trained models: Past, present and future", *AI Open*, 2, 2021, 225–50.

[56] L. He, S. Zheng, T. Yang, and F. Zhang, "KLMo: Knowledge Graph Enhanced Pretrained Language Model with Fine-Grained Relationships", in *Findings of the Association for Computational Linguistics: EMNLP 2021*, Punta Cana, Dominican Republic: Association for Computational Linguistics, November 2021, 4536–42, DOI: 10.18653/v1/2021.findings-emnlp.384, https://aclanthology.org/2021.findings-emnlp.384.

[57] J. Hewitt and C. D. Manning, "A structural probe for finding syntax in word representations", in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, 4129–38.

[58] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions", *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02), 1998, 107–16.

[59] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration", *arXiv preprint arXiv:1904.09751*, 2019.

[60] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for NLP", in *International Conference on Machine Learning*, PMLR, 2019, 2790–9.

[61] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models", *arXiv preprint arXiv:2106.09685*, 2021.

[62] E. J. Hu, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, *et al.*, "LoRA: Low-Rank Adaptation of Large Language Models", in *International Conference on Learning Representations*.

[63] Z. Hu, Y. Xu, W. Yu, S. Wang, Z. Yang, C. Zhu, K.-W. Chang, and Y. Sun, "Empowering language models with knowledge graph reasoning for open-domain question answering", in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022, 9562–81.

[64] L. Huang, L. Wu, and L. Wang, "Knowledge Graph-Augmented Abstractive Summarization with Semantic-Driven Cloze Reward", in *Proceed-*

ings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, 5094–107.

[65] X. Huang, J. Zhang, D. Li, and P. Li, "Knowledge graph embedding based question answering", in *Proceedings of the twelfth ACM international conference on web search and data mining*, 2019, 105–13.

[66] Z. Huang, G. Zweig, and B. Dumoulin, "Cache based recurrent neural network language model inference for first pass speech recognition", in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2014, 6354–8.

[67] K. Hwang and W. Sung, "Character-level language modeling with hierarchical recurrent neural networks", in *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, IEEE, 2017, 5720–4.

[68] D. Ippolito, D. Duckworth, C. Callison-Burch, and D. Eck, "Automatic Detection of Generated Text is Easiest when Humans are Fooled", in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, 1808–22.

[69] D. Ippolito, D. Grangier, D. Eck, and C. Callison-Burch, "Toward Better Storylines with Sentence-Level Language Models", in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, 7472–8.

[70] G. Jawahar, M. Abdul-Mageed, and V. Laks Lakshmanan, "Automatic Detection of Machine Generated Text: A Critical Survey", in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, 2296–309.

[71] F. Jelinek, B. Merialdo, S. Roukos, M. Strauss, *et al.*, "Self-organized language modeling for speech recognition", in *Readings in speech recognition*, Citeseer, 1990.

[72] F. Jelinek, "Continuous speech recognition by statistical methods", *Proceedings of the IEEE*, 64(4), 1976, 532–56.

[73] F. Jelinek, "Interpolated estimation of Markov source parameters from sparse data", in *Proc. Workshop on Pattern Recognition in Practice, 1980*, 1980.

[74] F. Jelinek, L. Bahl, and R. Mercer, "Design of a linguistic statistical decoder for the recognition of continuous speech", *IEEE Transactions on Information Theory*, 21(3), 1975, 250–6.

[75] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation", *ACM Computing Surveys*, 55(12), 2023, 1–38.

[76] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "TinyBERT: Distilling BERT for Natural Language Understanding", in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, 4163–74.

[77] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits, "Is bert really robust? a strong baseline for natural language attack on text classification and entailment", in *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34, No. 05, 2020, 8018–25.

[78] X. Jin, D. Zhang, H. Zhu, W. Xiao, S.-W. Li, X. Wei, A. Arnold, and X. Ren, "Lifelong Pretraining: Continually Adapting Language Models to Emerging Corpora", in *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, virtual+Dublin: Association for Computational Linguistics, May 2022, 1–16, DOI: 10.18653/v1/2022.bigscience-1.1, https://aclanthology.org/2022.bigscience-1.1.

[79] W. E. Johnson, "Probability: The deductive and inductive problems", *Mind*, 41(164), 1932, 409–23.

[80] M. Kang, T. Ng, and L. Nguyen, "Mandarin word-character hybrid-input neural network language model", in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[81] A. Kannan, Y. Wu, P. Nguyen, T. N. Sainath, Z. Chen, and R. Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model", in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2018, 1–5828.

[82] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models", *arXiv preprint arXiv:2001.08361*, 2020.

[83] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer", *IEEE transactions on acoustics, speech, and signal processing*, 35(3), 1987, 400–1.

[84] D. Kiecza, T. Schultz, and A. Waibel, "Data-driven determination of appropriate dictionary units for Korean LVCSR", in *Proceedings of ICASSP*, 1999, 323–7.

[85] T. Kim, J. Choi, D. Edmiston, and S.-g. Lee, "Are pre-trained language models aware of phrases? simple but strong baselines for grammar induction", *arXiv preprint arXiv:2002.00737*, 2020.

[86] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models", in *Thirtieth AAAI conference on artificial intelligence*, 2016.

[87] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.*, "Overcoming catastrophic forgetting in neural networks", *Proceedings of the national academy of sciences*, 114(13), 2017, 3521–6.

[88] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling", in *1995 international conference on acoustics, speech, and signal processing*, Vol. 1, IEEE, 1995, 181–4.

[89] S. Kombrink, T. Mikolov, M. Karafiát, and L. Burget, "Recurrent Neural Network Based Language Modeling in Meeting Recognition.", in *Interspeech*, Vol. 11, 2011, 2877–80.

[90] T. Kudo, "Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates", in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, 66–75.

[91] T. Kuhn, H. Niemann, and E. G. Schukat-Talamazzini, "Ergodic hidden Markov models and polygrams for language modeling", in *Proceedings of ICASSP'94. IEEE International Conference on Acoustics, Speech and Signal Processing*, Vol. 1, IEEE, 1994, I–357.

[92] I. Kulikov, A. Miller, K. Cho, and J. Weston, "Importance of Search and Evaluation Strategies in Neural Dialogue Modeling", in *Proceedings of the 12th International Conference on Natural Language Generation*, 2019, 76–87.

[93] C.-C. J. Kuo and A. M. Madni, "Green learning: Introduction, examples and outlook", *Journal of Visual Communication and Image Representation*, 2022, 103685.

[94] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations", *arXiv preprint arXiv:1909.11942*, 2019.

[95] A. Lazaridou, A. Kuncoro, E. Gribovskaya, D. Agrawal, A. Liska, T. Terzi, M. Gimenez, C. de Masson d'Autume, T. Kocisky, S. Ruder, *et al.*, "Mind the gap: Assessing temporal generalization in neural language models", *Advances in Neural Information Processing Systems*, 34, 2021, 29348–63.

[96] H. Lee, D. A. Hudson, K. Lee, and C. D. Manning, "SLM: Learning a Discourse Language Representation with Sentence Unshuffling", in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, 1551–62.

[97] B. Lester, R. Al-Rfou, and N. Constant, "The Power of Scale for Parameter-Efficient Prompt Tuning", in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, 3045–59.

[98] M. Levit, S. Parthasarathy, S. Chang, A. Stolcke, and B. Dumoulin, "Word-phrase-entity language models: Getting more mileage out of n-grams", in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[99] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension", in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, 7871–80.

[100] J. Li, W. Monroe, and D. Jurafsky, "A simple, fast diverse decoding algorithm for neural generation", *arXiv preprint arXiv:1611.08562*, 2016.

[101] J. Li, T. Tang, W. X. Zhao, J.-Y. Nie, and J.-R. Wen, "ELMER: A Non-Autoregressive Pre-trained Language Model for Efficient and Effective Text Generation", *arXiv preprint arXiv:2210.13304*, 2022.

[102] X. L. Li and P. Liang, "Prefix-Tuning: Optimizing Continuous Prompts for Generation", in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, 4582–97.

[103] Z. Li, E. Wallace, S. Shen, K. Lin, K. Keutzer, D. Klein, and J. Gonzalez, "Train big, then compress: Rethinking model size for efficient training and inference of transformers", in *International Conference on Machine Learning*, PMLR, 2020, 5958–68.

[104] G. J. Lidstone, "Note on the general case of the Bayes-Laplace formula for inductive or a posteriori probabilities", *Transactions of the Faculty of Actuaries*, 8(182-192), 1920, 13.

[105] K. Liu, J. Jiang, and F. Lyu, "A Domain Knowledge Enhanced Pre-Trained Language Model for Vertical Search: Case Study on Medicinal Products", in *Proceedings of the 29th International Conference on Computational Linguistics*, 2022, 1014–23.

[106] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing", *ACM Computing Surveys*, 55(9), 2023, 1–35.

[107] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach", *arXiv preprint arXiv:1907.11692*, 2019.

[108] R. Luo, L. Sun, Y. Xia, T. Qin, S. Zhang, H. Poon, and T.-Y. Liu, "BioGPT: generative pre-trained transformer for biomedical text generation and mining", *Briefings in Bioinformatics*, 23(6), 2022.

[109] K. Luu, D. Khashabi, S. Gururangan, K. Mandyam, and N. A. Smith, "Time Waits for No One! Analysis and Challenges of Temporal Misalignment", in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Seattle, United States: Association for Computational Linguistics, July 2022, 5944–58, DOI: 10.18653/v1/2022.naacl-main.435, https://aclanthology.org/2022.naacl-main.435.

[110] X. Ma, P. Zhang, S. Zhang, N. Duan, Y. Hou, M. Zhou, and D. Song, "A tensorized transformer for language modeling", *Advances in neural information processing systems*, 32, 2019.

[111] A. Madotto, Z. Liu, Z. Lin, and P. Fung, "Language models as few-shot learner for task-oriented dialogue systems", *arXiv preprint arXiv:2008.06239*, 2020.

[112] C. May, A. Wang, S. Bordia, S. R. Bowman, and R. Rudinger, "On Measuring Social Biases in Sentence Encoders", in *Proceedings of NAACL-HLT*, 2019, 622–8.

[113] E. McDermott, H. Sak, and E. Variani, "A density ratio approach to language model fusion in end-to-end automatic speech recognition", in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, IEEE, 2019, 434–41.

[114] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model.", in *Interspeech*, Vol. 2, No. 3, Makuhari, 2010, 1045–8.

[115] T. Mikolov, S. Kombrink, L. Burget, J. Černockỳ, and S. Khudanpur, "Extensions of recurrent neural network language model", in *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, IEEE, 2011, 5528–31.

[116] T. Mikolov, I. Sutskever, A. Deoras, H.-S. Le, S. Kombrink, and J. Cernocky, "Subword language modeling with neural networks", *preprint (http://www. fit. vutbr. cz/imikolov/rnnlm/char. pdf)*, 8(67), 2012.

[117] P. Mirowski and A. Vlachos, "Dependency Recurrent Neural Language Models for Sentence Completion", in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 2015, 511–7.

[118] Y. Miyamoto and K. Cho, "Gated word-character recurrent language model", in *2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016*, Association for Computational Linguistics (ACL), 2016, 1992–7.

[119] M. Nadeem, A. Bethke, and S. Reddy, "StereoSet: Measuring stereotypical bias in pretrained language models", in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, 5356–71.

[120] D. Narayanan, M. Shoeybi, J. Casper, P. LeGresley, M. Patwary, V. Korthikanti, D. Vainbrand, P. Kashinkunti, J. Bernauer, B. Catanzaro, *et al.*, "Efficient large-scale language model training on gpu clusters using megatron-lm", in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2021, 1–15.

[121] J. Ni, T. Young, V. Pandelea, F. Xue, and E. Cambria, "Recent advances in deep learning based dialogue systems: A systematic survey", *Artificial intelligence review*, 2022, 1–101.

[122] Y. Nie, A. Williams, E. Dinan, M. Bansal, J. Weston, and D. Kiela, "Adversarial NLI: A New Benchmark for Natural Language Understanding", in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, 4885–901.

[123] T. R. Niesler and P. C. Woodland, "A variable-length category-based n-gram language model", in *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, Vol. 1, IEEE, 1996, 164–7.

[124] F. J. Och, N. Ueffing, and H. Ney, "An efficient A* search algorithm for statistical machine translation", in *Proceedings of the ACL 2001 Workshop on Data-Driven Methods in Machine Translation*, 2001.

[125] M. Omar, S. Choi, D. Nyang, and D. Mohaisen, "Robust natural language processing: Recent advances, challenges, and future directions", *IEEE Access*, 2022.

[126] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep Contextualized Word Representations", in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana: Association for Computational Linguistics, June 2018, 2227–37, DOI: 10.18653/v1/N18-1202, https://aclanthology.org/N18-1202.

[127] F. Petroni, P. Lewis, A. Piktus, T. Rocktäschel, Y. Wu, A. H. Miller, and S. Riedel, "How Context Affects Language Models' Factual Predictions", in *Automated Knowledge Base Construction*, 2020, https://openreview.net/forum?id=025X0zPfn.

[128] F. Petroni, T. Rocktäschel, S. Riedel, P. Lewis, A. Bakhtin, Y. Wu, and A. Miller, "Language Models as Knowledge Bases?", in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, 2463–73.

[129] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, and I. Gurevych, "AdapterFusion: Non-Destructive Task Composition for Transfer Learning", in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2021, 487–503.

[130] R. Pryzant, D. Iter, J. Li, Y. T. Lee, C. Zhu, and M. Zeng, "Automatic prompt optimization with" gradient descent" and beam search", *arXiv preprint arXiv:2305.03495*, 2023.

[131] G. Qin and J. Eisner, "Learning How to Ask: Querying LMs with Mixtures of Soft Prompts", in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, 5203–12.

[132] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, "Pre-trained models for natural language processing: A survey", *Science China Technological Sciences*, 63(10), 2020, 1872–97.

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

[134] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners", *OpenAI blog*, 1(8), 2019, 9.

[135] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer.", *J. Mach. Learn. Res.*, 21(140), 2020, 1–67.

[136] R. Al-Rfou, D. Choe, N. Constant, M. Guo, and L. Jones, "Character-level language modeling with deeper self-attention", in *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33, No. 01, 2019, 3159–66.

[137] M. T. Ribeiro, C. Guestrin, and S. Singh, "Are red roses red? evaluating consistency of question-answering models", in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, 6174–84.

[138] K. Ries, F. D. Buo, and A. Waibel, "Class phrase models for language modeling", in *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP'96*, Vol. 1, IEEE, 1996, 398–401.

[139] A. Rogers, O. Kovaleva, and A. Rumshisky, "A primer in bertology: What we know about how bert works", *Transactions of the Association for Computational Linguistics*, 8, 2020, 842–66.

[140] R. Rosenfeld, "A whole sentence maximum entropy language model", in *1997 IEEE workshop on automatic speech recognition and understanding proceedings*, IEEE, 1997, 230–7.

[141] R. Rosenfeld, S. F. Chen, and X. Zhu, "Whole-sentence exponential language models: a vehicle for linguistic-statistical integration", *Computer Speech & Language*, 15(1), 2001, 55–73.

[142] R. Rosenfeld, "A maximum entropy approach to adaptive statistical language modeling", 1996.

[143] S. Rothe, S. Narayan, and A. Severyn, "Leveraging pre-trained checkpoints for sequence generation tasks", *Transactions of the Association for Computational Linguistics*, 8, 2020, 264–80.

[144] T. Rotovnik, M. S. Maučec, and Z. Kačič, "Large vocabulary continuous speech recognition of an inflected language using stems and endings", *Speech communication*, 49(6), 2007, 437–52.

[145] B. Rychalska, D. Basaj, A. Gosiewska, and P. Biecek, "Models in the wild: On corruption robustness of neural nlp systems", in *Neural Information Processing: 26th International Conference, ICONIP 2019, Sydney, NSW, Australia, December 12–15, 2019, Proceedings, Part III 26*, Springer, 2019, 235–47.

[146] H. Sak, M. Saraclar, and T. Güngör, "Morphology-based and sub-word language modeling for Turkish speech recognition", in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2010, 5402–5.

[147] J. Salazar, D. Liang, T. Q. Nguyen, and K. Kirchhoff, "Masked Language Model Scoring", in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, 2699–712.

[148] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter", *arXiv preprint arXiv:1910.01108*, 2019.

[149] G. Saon and M. Padmanabhan, "Data-driven approach to designing compound words for continuous speech recognition", *IEEE transactions on Speech and audio processing*, 9(4), 2001, 327–32.

[150] R. Sarikaya, M. Afify, and Y. Gao, "Joint morphological-lexical language modeling (JMLLM) for Arabic", in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, Vol. 4, IEEE, 2007, IV–181.

[151] N. Saunshi, S. Malladi, and S. Arora, "A mathematical exploration of why language models help solve downstream tasks", *arXiv preprint arXiv:2010.03648*, 2020.

[152] T. Schick and H. Schütze, "Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference", in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2021, 255–69.

[153] T. Schick and H. Schütze, "Few-shot text generation with natural language instructions", in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, 390–402.

[154] M. Schuster and K. Nakajima, "Japanese and korean voice search", in *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, IEEE, 2012, 5149–52.

[155] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green ai", *Communications of the ACM*, 63(12), 2020, 54–63.

[156] H. Schwenk, "Continuous space language models", *Computer Speech & Language*, 21(3), 2007, 492–518.

[157] H. Schwenk and J.-L. Gauvain, "Training neural network language models on very large corpora", in *Proceedings of human language technology conference and conference on empirical methods in natural language processing*, 2005, 201–8.

[158] R. Sennrich, B. Haddow, and A. Birch, "Neural Machine Translation of Rare Words with Subword Units", in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, 1715–25.

[159] C. Shan, C. Weng, G. Wang, D. Su, M. Luo, D. Yu, and L. Xie, "Component fusion: Learning replaceable language model component for end-to-end speech recognition system", in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2019, 5361–635.

[160] O. Sharir, B. Peleg, and Y. Shoham, "The cost of training nlp models: A concise overview", *arXiv preprint arXiv:2004.08900*, 2020.

[161] J. Shin, Y. Lee, and K. Jung, "Effective sentence scoring method using bert for speech recognition", in *Asian Conference on Machine Learning*, PMLR, 2019, 1081–93.

[162] T. Shin, Y. Razeghi, R. L. Logan IV, E. Wallace, and S. Singh, "AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts", in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, 4222–35.

[163] M. Siu and M. Ostendorf, "Variable n-grams and extensions for conversational speech language modeling", *IEEE Transactions on Speech and Audio Processing*, 8(1), 2000, 63–75.

[164] A. Sriram, H. Jun, S. Satheesh, and A. Coates, "Cold Fusion: Training Seq2Seq Models Together with Language Models", *Proc. Interspeech 2018*, 2018, 387–91.

[165] H. Stiff and F. Johansson, "Detecting computer-generated disinformation", *International Journal of Data Science and Analytics*, 13(4), 2022, 363–83.

[166] E. Strubell, A. Ganesh, and A. McCallum, "Energy and Policy Considerations for Deep Learning in NLP", in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, 3645–50.

[167] Y. Su, D. Cai, Y. Wang, D. Vandyke, S. Baker, P. Li, and N. Collier, "Non-Autoregressive Text Generation with Pre-trained Language Models", in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2021, 234–43.

[168] B. Suhm and A. Waibel, "Towards better language models for spontaneous speech", 1994.

[169] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling", in *Thirteenth annual conference of the international speech communication association*, 2012.

[170] I. Sutskever, J. Martens, and G. E. Hinton, "Generating text with recurrent neural networks", in *ICML*, 2011.

[171] I. Tenney, D. Das, and E. Pavlick, "BERT Rediscovers the Classical NLP Pipeline", in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, 4593–601.

[172] I. Tenney, P. Xia, B. Chen, A. Wang, A. Poliak, R. T. McCoy, N. Kim, B. Van Durme, S. R. Bowman, D. Das, *et al.*, "What do you learn from context? Probing for sentence structure in contextualized word representations", in *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[173] N. C. Thompson, K. Greenewald, K. Lee, and G. F. Manso, "The computational limits of deep learning", *arXiv preprint arXiv:2007.05558*, 2020.

[174] R. Thoppilan, D. De Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos, L. Baker, Y. Du, *et al.*, "Lamda: Language models for dialog applications", *arXiv preprint arXiv:2201.08239*, 2022.

[175] J. Thorne and A. Vlachos, "Automated Fact Checking: Task Formulations, Methods and Future Directions", in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, 3346–59.

[176] S. Toshniwal, A. Kannan, C.-C. Chiu, Y. Wu, T. N. Sainath, and K. Livescu, "A comparison of techniques for language model integration in encoder-decoder speech recognition", in *2018 IEEE spoken language technology workshop (SLT)*, IEEE, 2018, 369–75.

[177] I. Turc, M.-W. Chang, K. Lee, and K. Toutanova, "Well-read students learn better: On the importance of pre-training compact models", *arXiv preprint arXiv:1908.08962*, 2019.

[178] A. Uchendu, T. Le, K. Shu, and D. Lee, "Authorship attribution for neural text generation", in *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

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

[180] L. Verwimp, J. Pelemans, P. Wambacq, *et al.*, "Character-Word LSTM Language Models", in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 2017, 417–27.

[181] A. Vijayakumar, M. Cogswell, R. Selvaraju, Q. Sun, S. Lee, D. Crandall, and D. Batra, "Diverse beam search for improved description of complex scenes", in *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32, No. 1, 2018.

[182] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned", in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, 5797–808.

[183] A. Wang and K. Cho, "BERT has a Mouth, and It Must Speak: BERT as a Markov Random Field Language Model", in *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, 2019, 30–6.

[184] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "Superglue: A stickier benchmark for general-purpose language understanding systems", *Advances in neural information processing systems*, 32, 2019.

[185] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding", in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2018, 353–5.

[186] B. Wang and C.-C. J. Kuo, "SBERT-WK: A sentence embedding method by dissecting bert-based word models", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28, 2020, 2146–57.

[187] B. Wang and H. Li, "Relational Sentence Embedding for Flexible Semantic Matching", *arXiv preprint arXiv:2212.08802*, 2022.

[188] B. Wang, G. Wang, J. Huang, J. You, J. Leskovec, and C.-C. J. Kuo, "Inductive learning on commonsense knowledge graph completion", in *2021 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2021, 1–8.

[189] B. Wang, C. Zhang, C. Wei, and H. Li, "A Focused Study on Sequence Length for Dialogue Summarization", *arXiv preprint arXiv:2209.11910*, 2022.

[190] B. Wang, C. Zhang, Y. Zhang, Y. Chen, and H. Li, "Analyzing and Evaluating Faithfulness in Dialogue Summarization", in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, December 2022, 4897–908, https://aclanthology.org/2022.emnlp-main.325.

[191] Q. Wang, B. Li, T. Xiao, J. Zhu, C. Li, D. F. Wong, and L. S. Chao, "Learning Deep Transformer Models for Machine Translation", in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, 1810–22.

[192] X. Wang, Q. Liu, T. Gui, Q. Zhang, Y. Zou, X. Zhou, J. Ye, Y. Zhang, R. Zheng, Z. Pang, *et al.*, "Textflint: Unified multilingual robustness evaluation toolkit for natural language processing", in *Proceedings of*

the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations, 2021, 347–55.

[193] X. Wang, T. Gao, Z. Zhu, Z. Zhang, Z. Liu, J. Li, and J. Tang, "KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation", *Transactions of the Association for Computational Linguistics*, 9, 2021, 176–94, DOI: 10.1162/tacl_a_00360, https://aclanthology.org/2021.tacl-1.11.

[194] Y.-C. Wang, X. Ge, B. Wang, and C.-C. J. Kuo, "GreenKGC: A lightweight knowledge graph completion method", *arXiv preprint arXiv: 2208.09137*, 2022.

[195] Y.-C. Wang, X. Ge, B. Wang, and C.-C. J. Kuo, "KGBoost: A classification-based knowledge base completion method with negative sampling", *Pattern Recognition Letters*, 157, 2022, 104–11.

[196] Y.-C. Wang, J. Xue, C. Wei, and C.-C. J. Kuo, "An Overview on Generative AI at Scale with Edge-Cloud Computing", 2023.

[197] Z. Wang, J. Wohlwend, and T. Lei, "Structured Pruning of Large Language Models", in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, 6151–62.

[198] C. Wei, B. Wang, and C.-C. J. Kuo, "Synwmd: Syntax-aware word mover's distance for sentence similarity evaluation", *Pattern Recognition Letters*, 170, 2023, 48–55.

[199] C. Wei, B. Wang, and C.-C. J. Kuo, "Task-Specific Dependency-based Word Embedding Methods", *Pattern Recognition Letters*, 2022.

[200] C. Wei, S. M. Xie, and T. Ma, "Why do pretrained language models help in downstream tasks? an analysis of head and prompt tuning", *Advances in Neural Information Processing Systems*, 34, 2021, 16158–70.

[201] M. Weiss, "Deepfake bot submissions to federal public comment websites cannot be distinguished from human submissions", *Technology Science*, 2019.

[202] C.-J. Wu, R. Raghavendra, U. Gupta, B. Acun, N. Ardalani, K. Maeng, G. Chang, F. Aga, J. Huang, C. Bai, *et al.*, "Sustainable ai: Environmental implications, challenges and opportunities", *Proceedings of Machine Learning and Systems*, 4, 2022, 795–813.

[203] C.-S. Wu and C. Xiong, "Probing Task-Oriented Dialogue Representation from Language Models", in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online: Association for Computational Linguistics, November 2020, 5036–51, DOI: 10.18653/v1/2020.emnlp-main.409, https://aclanthology.org/2020.emnlp-main.409.

[204] W. Xiong, J. Du, W. Y. Wang, and V. Stoyanov, "Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model", *arXiv preprint arXiv:1912.09637*, 2019.

[205] J. Xu, W. Zhou, Z. Fu, H. Zhou, and L. Li, "A survey on green deep learning", *arXiv preprint arXiv:2111.05193*, 2021.

[206] Z. Xu, B. Liu, B. Wang, C.-J. Sun, X. Wang, Z. Wang, and C. Qi, "Neural response generation via gan with an approximate embedding layer", in *Proceedings of the 2017 conference on empirical methods in natural language processing*, 2017, 617–26.

[207] L. Xue, A. Barua, N. Constant, R. Al-Rfou, S. Narang, M. Kale, A. Roberts, and C. Raffel, "Byt5: Towards a token-free future with pre-trained byte-to-byte models", *Transactions of the Association for Computational Linguistics*, 10, 2022, 291–306.

[208] K. Yamada and K. Knight, "A decoder for syntax-based statistical MT", in *Proceedings of the 40th Annual meeting of the Association for Computational Linguistics*, 2002, 303–10.

[209] Y. Yang, C. Huang, L. Xia, and C. Li, "Knowledge graph contrastive learning for recommendation", in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022, 1434–43.

[210] Z. Yang, Z. Dai, R. Salakhutdinov, and W. W. Cohen, "Breaking the softmax bottleneck: A high-rank RNN language model", *arXiv preprint arXiv:1711.03953*, 2017.

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

[212] M. Yasunaga, A. Bosselut, H. Ren, X. Zhang, C. D. Manning, P. S. Liang, and J. Leskovec, "Deep bidirectional language-knowledge graph pretraining", *Advances in Neural Information Processing Systems*, 35, 2022, 37309–23.

[213] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, and J. Leskovec, "QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering", in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online: Association for Computational Linguistics, June 2021, 535–46, DOI: 10.18653/v1/2021.naacl-main.45, https://aclanthology.org/2021.naacl-main.45.

[214] Y. Zang, F. Qi, C. Yang, Z. Liu, M. Zhang, Q. Liu, and M. Sun, "Word-level Textual Adversarial Attacking as Combinatorial Optimization", in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, 6066–80.

[215] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi, "Defending against neural fake news", *Advances in neural information processing systems*, 32, 2019.

[216] D. Zhang, Z. Yuan, Y. Liu, F. Zhuang, H. Chen, and H. Xiong, "E-BERT: A phrase and product knowledge enhanced language model for e-commerce", *arXiv preprint arXiv:2009.02835*, 2020.

[217] X. Zhang, A. Bosselut, M. Yasunaga, H. Ren, P. Liang, C. D. Manning, and J. Leskovec, "GreaseLM: Graph REASoning Enhanced Language Models", in *International Conference on Learning Representations*, 2022, https://openreview.net/forum?id=41e9o6cQPj.

[218] Y. Zhang, A. Warstadt, X. Li, and S. Bowman, "When Do You Need Billions of Words of Pretraining Data?", in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, 1112–25.

[219] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, "ERNIE: Enhanced Language Representation with Informative Entities", in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy: Association for Computational Linguistics, July 2019, 1441–51, DOI: 10.18653/v1/P19-1139, https://aclanthology.org/P19-1139.

[220] T. Zhao, R. Zhao, and M. Eskenazi, "Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders", in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, 654–64.

[221] Z. Zhong, D. Friedman, and D. Chen, "Factual Probing Is [MASK]: Learning vs. Learning to Recall", in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, 5017–33.

[222] Y. Zhou, A. I. Muresanu, Z. Han, K. Paster, S. Pitis, H. Chan, and J. Ba, "Large language models are human-level prompt engineers", *arXiv preprint arXiv:2211.01910*, 2022.

[223] J. E. Zini and M. Awad, "On the explainability of natural language processing deep models", *ACM Computing Surveys*, 55(5), 2022, 1–31.