Pre-Trained Language Models for Text Generation: A Survey

JUNYI LI, Renmin University of China, Beijing, China and Université de Montréal, Montréal, Canada
TIANYI TANG, Renmin University of China, Beijing, China
WAYNE XIN ZHAO, Renmin University of China, Beijing, China
JIAN-YUN NIE, Université de Montréal, Montréal, Canada
JI-RONG WEN, Renmin University of China, Beijing, China

Text Generation aims to produce plausible and readable text in human language from input data. The resurgence of deep learning has greatly advanced this field, in particular, with the help of neural generation models based on pre-trained language models (PLMs). Text generation based on PLMs is viewed as a promising approach in both academia and industry. In this article, we provide a survey on the utilization of PLMs in text generation. We begin with introducing two key aspects of applying PLMs to text generation: (1) how to design an effective PLM to serve as the generation model; and (2) how to effectively optimize PLMs given the reference text and to ensure that the generated texts satisfy special text properties. Then, we show the major challenges that have arisen in these aspects, as well as possible solutions for them. We also include a summary of various useful resources and typical text generation applications based on PLMs. Finally, we highlight the future research directions which will further improve these PLMs for text generation. This comprehensive survey is intended to help researchers interested in text generation problems to learn the core concepts, the main techniques and the latest developments in this area based on PLMs.

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Computing methodologies** → **Natural language generation;**

Additional Key Words and Phrases: Pre-trained language models, natural language processing

ACM Reference Format:
Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2024. Pre-Trained Language Models for Text Generation: A Survey. *ACM Comput. Surv.* 56, 9, Article 230 (April 2024), 39 pages. https://doi.org/10.1145/3649449

Junyi Li and Tianyi Tang contributed equally to this work.
We also appreciate the helpful suggestions from readers of the pre-print version. This work was partially supported by National Natural Science Foundation of China under Grant No. 62222215 and Beijing Natural Science Foundation under Grant No. 4222027 and L233008. Xin Zhao is the corresponding author.
Authors' addresses: J. Li, Gaoling School of Artificial Intelligence, Beijing Key Laboratory of Big Data Management and Analysis Methods, Renmin University of China, No. 59 Zhongguancun Street, Haidian District Beijing, China, 100872 and DIRO, Université de Montréal, 2900 Bd Édouard-Montpetit, Montréal (Québec), Canada, H3T1J4; e-mail: lijunyi@ruc.edu.cn; T. Tang and W. X. Zhao (Corresponding author), Gaoling School of Artificial Intelligence, Beijing Key Laboratory of Big Data Management and Analysis Methods, Renmin University of China, No. 59 Zhongguancun Street, Haidian District Beijing, China, 100872; e-mails: steventianyitang@outlook.com, batmanfly@gmail.com; J.-Y. Nie, DIRO, Université de Montréal, 2900 Bd Édouard-Montpetit, Montréal (Québec), Canada, H3C3J7; e-mail: nie@iro.umontreal.ca; J.-R. Wen, Gaoling School of Artificial Intelligence, School of Information, Beijing Key Laboratory of Big Data Management and Analysis Methods, Renmin University of China, No. 59 Zhongguancun Street, Haidian District Beijing, China, 100872; e-mail: jrwen@ruc.edu.cn.

ACM 0360-0300/2024/04-ART230
https://doi.org/10.1145/3649449

1 INTRODUCTION

Text generation, also known as *natural language generation*, has been one ofthe most important sub-fields in **natural language processing (NLP)**. It aims to produce plausible and readable text in a human language, from the input data in various forms including text, image, table and knowledge base. In the last decades, text generation techniques have been extensively applied to a wide range of applications. For example, they have been used in dialogue systems to generate responses to user utterances in a conversation [283], in machine translation to translate a text from one language into another [35]; and in text summarization to generate an abridged summary of the source text [48].

The primary goal of text generation is to automatically learn an input-to-output mapping from the data to build an end-to-end solution with minimal human intervention. This mapping function allows the generation system to generalize to a broader field and generate free text under the given input. Earlier approaches usually adopt statistical language models for modeling the conditional probabilities of words given an *n*-gram context [13, 15]. Such a statistical approach is known to suffer from the data sparsity issue. To better estimate the occurrence of unobserved terms, a number of smoothing methods have been developed [232, 268]. Still, words are used as the basic representation units in these approaches, which leads to the issue that similar words cannot be easily mapped with each other.

With the emergence of deep learning techniques [120], neural network models have dominated the mainstream methods in text generation and make exceptional success in generating natural language texts. Deep neural generation models usually adopt the sequence-to-sequence framework [230] based on the encoder-decoder scheme: the encoder first maps the input sequence into fix-sized low-dimensional vectors (called *input embeddings*), and then the decoder generates an output text based on the input embeddings. The distributed representation makes a key difference from earlier statistical approaches, making it easier to cope with the possible relations between inputs and outputs. Various neural models have been proposed with different designs for the encoder-decoder architecture, such as **graph neural networks (GNN)** for encoding graph inputs [126] and **recurrent neural networks (RNN)** for decoding texts [133]. Besides, the attention mechanism [5] and copy mechanism [211] are widely used to improve the performance of text generation models. An important merit of deep neural networks for text generation is that they enable end-to-end learning of semantic mappings from the input data to output texts without labor-intensive feature engineering. Moreover, deep neural models employ low-dimensional semantic representations [98] to capture linguistic features of language, which is useful to alleviate data sparsity.

Despite the success of deep neural models for text generation, a major performance bottleneck lies in the availability of large-scale labelled datasets. Most neural generation methods require substantial amounts of labelled data, which restricts their application to many domains that suffer from a dearth of annotated examples. To date, most existing labelled datasets for text generation tasks are usually small. In such cases, deep neural networks are likely to overfit on these small datasets and do not generalize well in practice. Moreover, the early neural models for text generation were still relatively shallow with only 1~3 neural layers. Therefore, these models have difficulties in modeling intricate relationships between the context and word meanings and deriving contextual word representations for better generation [191].

In recent years, the paradigm of **pre-trained language models (PLMs)** is thriving in NLP [191]. The basic idea is to first pre-train the models on large-scale unsupervised corpora and then fine-tune these models in downstream supervised task datasets. With the emergence of Transformer [237] and higher computational power, the architecture of PLMs has evolved from shallow to deeper architectures, such as BERT [43] and OpenAI GPT [16, 193]. More recently, the advent of ChatGPT and GPT-4 [175] has stirred public perception of AI. Generative models, especially conversational AI systems, have been considered as the paradigm to approach **artificial general intelligence (AGI)** [17]. Substantial work has shown that PLMs can encode massive amounts of linguistic knowledge from the pre-training corpora into their large-scale parameters and learn universal and contextual representations of the language with specially designed objectives such as next token prediction. Therefore, PLMs are generally beneficial for downstream tasks and can avoid training a new model from scratch. Following the success of PLMs in other NLP tasks, researchers have proposed to apply PLMs to text generation tasks [16, 123, 194]. Pre-trained on large-scale corpora, PLMs can understand natural language accurately and further express in human language fluently, both of which are critical abilities to fulfill text generation tasks. Grounding text generation on PLMs is seen as a promising direction in both academia and industry, which has much advanced the state of the art in this field. Thus, in this survey, we focus on text generation based on large PLMs.

There are a number of survey papers on text generation and PLMs. For example, Qiu et al. [191] summarized two generations of PLMs for the whole NLP domain and introduced various extensions and adaption approaches of PLMs. Kalyan et al. [106] gave a brief overview of the advances of self-supervised learning in PLMs. Han *et al.* [84] took a deep look into the history of pre-training, especially its special relation with transfer learning and self-supervised learning. Zhao et al. [280] presented a comprehensive survey of large language models including four major aspects, namely pre-training, adaptation tuning, utilization, and capacity evaluation. In contrast, we are more focused on in-depth analysis of research on PLMs for text generation, rather than paying attention to the entire NLP community. Besides, researchers in [48], [196], and [263], focused on specific applications, e.g., summarization, dialogue, and translation, but did not go deep into the core technique. As text generation is a key component in various applications, it is useful to provide a comprehensive survey on the topic of text generation based on PLMs. Different from existing surveys, this survey is intended to provide a more general description on this common task, rather than limiting it to specific applications. It is worth noting that this survey is an extended version of the short survey [131]. The extensions include: (1) This article covers a wider range of existing studies, evaluations, open-source libraries, and common applications of PLMs-based text generation, going far beyond the scope of the previous short survey; (2) This article provides a new schematic view involving two key aspects (i.e., model architecture, parameter optimization) about applying PLMs to text generation, which constitute the main content of this article; (3) To provide a better picture of existing solutions for various challenges, this article includes more detailed descriptions and discussions about their technical contributions.

The remainder of this survey is organized as follows: We first present the task formulation and an overview of PLMs in Section 2. Given the encoded input data, the goal of text generation is to optimize the generation function (i.e., PLMs) for generating satisfactory output text. Thus, two key points are involved when applying PLMs to text generation: (1) how to design an effective PLM to serve as the generation function (Section 3); and (2) how to optimize PLMs given the reference text and to ensure that the generated texts satisfy special text properties (Section 4). Then, we discuss several typical non-trivial challenges and solutions in Section 5. We present a summary of various useful resources to work with PLMs in Section 6 and common applications in Section 7. Finally, we summarize the contribution of this survey and describe future directions in Section 8.

2 PRELIMINARY

In this section, we first give a general task definition of text generation, then describe the background of PLMs, and finally introduce the paradigm of PLM-based text generation.

2.1 Text Generation

Generally, a text can be modeled as a sequence of tokens $y = \langle y_1, \ldots, y_j, \ldots, y_n \rangle$, where each token y_j is drawn from a vocabulary $\mathcal{V}$. The task of text generation aims to generate plausible and readable text in a human language. In most cases, text generation is conditioned on some input data (e.g., text, image, tabular, and knowledge base), which is denoted as x. In particular, the generated text is expected to satisfy some desired language properties such as fluency, naturalness, and coherence. We denote the desired properties for output text as a property set $\mathbb{P}$. Based on the above notations, the task of text generation can be formally described as:

$$y = f_{\mathcal{M}}(x, \mathbb{P}), \tag{1}$$

where the text generation model $f_{\mathcal{M}}$ produces the output text y given the input data x, satisfying some special proprieties from the property set $\mathbb{P}$. In this survey, the text generation model $f_{\mathcal{M}}$ is specially crafted based on a PLM $\mathcal{M}$.

Specifically, according to the type of the input data x and the property set $\mathbb{P}$, text generation can be instantiated into different kinds of real-world tasks:

— When the input data x is *not provided* or is *a random vector*, text generation will degenerate into language modeling or unconditional text generation [192, 193]. In this case, the output text is required to satisfy some common language properties, such as fluency and naturalness.

— When the input data x is a set of *discrete attributes* (e.g., topic words and sentiment labels), it can be exemplified by review generation [133] or essay generation [110]. The input data controls the content of the generated text and the output text should adhere to the input attributes.

— When the input data x is *structured data*, such as knowledge base or table, it is considered as data-to-text generation such as weather report generation [72, 130]. This task aims to generate a descriptive text about the input data. Thus, the output text should be faithful to the input data.

— When the input data x is *multimedia input*, such as image and speech, it becomes image captioning [251] or speech recognition [54]. We may expect that the caption text be lively for attracting children's attention, and the converted speech text be faithful to the original speech.

— The most common form of input data x is *a text sequence*. This form spans a number of applications such as machine translation [35], text summarization [205], and dialogue system [274]. For a specific task, the output text is expected to satisfy desired properties. For example, the summaries in text summarization should not contradict the facts described in the input text, and the responses in dialog should be relevant to the input dialogue history and context. To make a clear illustration of existing popular text generation tasks, we present the input and output for different kinds of tasks in Table 1.

2.2 Pre-Trained Language Models

Pre-trained language models (PLMs) are deep neural networks that are pre-trained on large-scale unlabelled corpora, which can be further fine-tuned on various downstream tasks. It has been shown that PLMs can encode a significant amount of linguistic knowledge into their vast amounts of parameters [131, 202]. Therefore, it is promising to apply PLMs to enhance the understanding of language and improve the generation quality.

Owing to the great success of Transformer [237], almost all PLMs employ it as the backbone. As two typical PLMs, GPT [192] and BERT [43] are first built upon Transformer decoder and encoder

Table 1. Summary of Common Text Generation Tasks w.r.t. Input/Output, Datasets, Metrics, and Our Recommended Model Regime and Tuning Method

Tasks	Input/Output	Model Regime	Tuning Method	Datasets	Metrics
Machine Translation	*I:* Text in language *A* *O:* Text in language *B*	Encoder-Decoder	Intermediate FT, Multi-task FT	WMT'14, 16 [35] IWSLT'14, 15 [149]	BLEU, COMET, METEOR, ChrF
Text Summarization	*I:* Long document *O:* Short summary	Encoder-Decoder Decoder-only	Prompt Tuning, Efficient FT	CNN/DailyMail [219], XSum [219]	ROUGE, BLEU, BERTScore
Dialogue System	*I:* Dialogue history *O:* Response utterance	Decoder-only	Prompt Tuning	PersonaChat [9], DailyDialogue [9]	BLEU, Distinct, Perplexity
Question Answering	*I:* Question (and passage) *O:* Answer	Encoder-Decoder Decoder-only	Prompt Tuning	TriviaQA [104], OpenBookQA [167]	Exact Match, Accuracy
Story/Essay Generation	*I:* Title or topics *O:* Story/Essay text	Encoder-Decoder Decoder-only	Property Tuning (Relevance)	ROCStories [79], WritingPrompts [197]	Perplexity, BLEU, Distinct
Data-to-text Generation	*I:* Structured data *O:* Description text	Encoder-Decoder	Property Tuning (Faithfulness)	WebNLG [202], WikiBio [33], E2E [29]	BLEU, ChrF++ METEOR
Image Captioning	*I:* Image *O:* Caption text	Encoder-Decoder	Vanilla FT	MS COCO [141], COCO Captions [30]	BLEU, ROUGE, METEOR

"FT" is short for fine-tuning.

respectively. Following GPT and BERT, PLMs such as XLNet [257], RoBERTa [151], ERNIE [276], T5 [194] and BART [123] are propopsed in the literature. Among them, XLNet, RoBERTa and ERNIE are developed based on the BERT model, while T5 and BART are encoder-decoder based PLMs. Researchers find that scaling PLM (e.g., scaling model size or data size [107]) often leads to an improved model capacity on downstream tasks. This triggers a number of studies to explore the performance limit by training ever larger PLMs (e.g., the 175B GPT-3 [16] and the 540B PaLM [11]). Based on these PLMs, there emerge several remarkable dialogue applications such as *ChatGPT* by OpenAI, *Bard* by Google, and *Claude* by Anthropic. Moreover, OpenAI developed GPT-4 [175], a large-scale, multimodal model which was considered as an early (yet still incomplete) version of an AGI system [17]. To promote transparency and access to large-scale PLMs, Meta opened source Llama 2 [235] and made it available free of charge for research and commercial use. We can say that the research areas of AI are being revolutionized by the rapid progress of generative PLMs. In addition, PLMs have been designed for other tasks such as named entity recognition [185], programming [57], and networking [155]. According to the backbone architectures, PLMs for text generation can be categorized as encoder-only LMs, decoder-only LMs, and encoder-decoder LMs, which will be detailed in Section 3.

2.3 PLM-Based Text Generation Methods

To effectively leverage PLMs for downstream text generation tasks, we need to consider two key aspects from the perspectives of model architecture and optimization algorithm, respectively:

— *Model Architecture. How to design an effective PLM $\mathcal{M}$ to serve as the generation model $f_{\mathcal{M}}$ and adapt to various text generation tasks?* In the literature, a number of PLMs have been developed with generalized architectures for general purposes (e.g., causal decoder [193]). While, these general architectures are focused on text input and auto-regressive decoding, which cannot cope with some special text generation cases. Many text generation tasks involve different kinds of input such as structured and multimodal data. Besides, online real-time applications such as query rewriting in search engines require efficient and low-latency decoding method. Therefore, it is important to make specific designs on the underlying PLMs for achieving good task performance.

— *Optimization Algorithm. How to optimize the text generation model (i.e., PLMs) $f_{\mathcal{M}}$ given the reference text y and ensure that the generated text satisfies special text properties $\mathbb{P}$?* In order to produce satisfactory text, it is critical to develop effective optimization algorithms for optimizing the text generation model. A major challenge stems from the fact that some desired properties for output text are difficult to be formulated or optimized. Besides, parameter-efficient optimization of PLMs (e.g., prompt tuning) has become a wide consensus for many recent studies, which aims to save the computational resource in a limited memory and GPU budget.

In the following sections, we will present recent research efforts on PLM-based text generation, with an emphasis on the two aforementioned aspects (Section 3 for model architecture and Section 4 for optimization algorithm). Figure 1 shows the overall organization of our survey.

3 DESIGNING PLM ARCHITECTURES FOR TEXT GENERATION

Given the breadth and depth of PLMs' capabilities, they have become the first choice to implement the text generation function $f_{\mathcal{M}}$, which models the text generation objective as the conditional probability of output text y given the input data x as follows:

$$\Pr_{\mathcal{M}}(y|x) = \prod_{j=1}^{n} \Pr_{\mathcal{M}}(y_j|y_{<j}, x), \tag{2}$$

where y_j denotes the jth output token, and $y_{<j}$ denotes the previous tokens $y_1, \ldots, y_{j-1}$.

With the excellent parallelization capacities, Transformer [237] has become the dominant framework for developing very large PLMs. According to the backbone of Transformers, PLMs for text generation encompass three basic architectures, namely encoder-only LMs, decoder-only LMs, and encoder-decoder LMs (see Section 3.1). To adapt to wider text generation application scenarios, these basic architectures have been advanced with respect to input modalities, module adaptation, and decoding mechanism (see Section 3.2).

3.1 Basic Architectures

Existing PLMs for text generation adopt either a single Transformer or a Transformer-based encoder-decoder as the backbone. PLMs, such as UniLM [45] and GPT-3 [16], use a single Transformer decoder to simultaneously implement the process of input encoding and output decoding. PLMs based on a single Transformer include two major variants: *encoder-only LMs* and *decoder-only LMs*, with different attention mask strategies. In contrast, PLMs built upon Transformer encoder-decoder, called *encoder-decoder LMs*, perform input encoding and output decoding separately. In the following, we will describe these four variants in detail. Table 2 summarizes three basic architectures and their corresponding pre-training objectives and representative PLMs.

3.1.1 Encoder-Only Language Models. Language models with the architecture of Transformer encoder are equipped with the full attention and usually pre-trained with **masked language modeling (MLM)** task, i.e., predicting the masked tokens using the bidirectional information. The most representative model is BERT [43], which is used extensively in language understanding.

However, due to the discrepancy between the pre-training task of masked LMs and the downstream generation function, masked LMs are rarely utilized for text generation tasks [257]. It is more common to use masked LMs as the encoder part for text generation, allowing to leverage the excellent bidirectional encoding capacities. For example, Rothe *et al.* [205] proposed to initialize both the encoder and decoder of the generation model with BERT [43], which yields comparable performance with other PLMs specially designed for text generation.

3.1.2 Decoder-Only Language Models. The decoder-only LMs are designed for language modeling, i.e., predicting the next word based on all previous words, which is natural for text generation. They have two variants: causal decoder and prefix decoder. The causal decoder adopts a

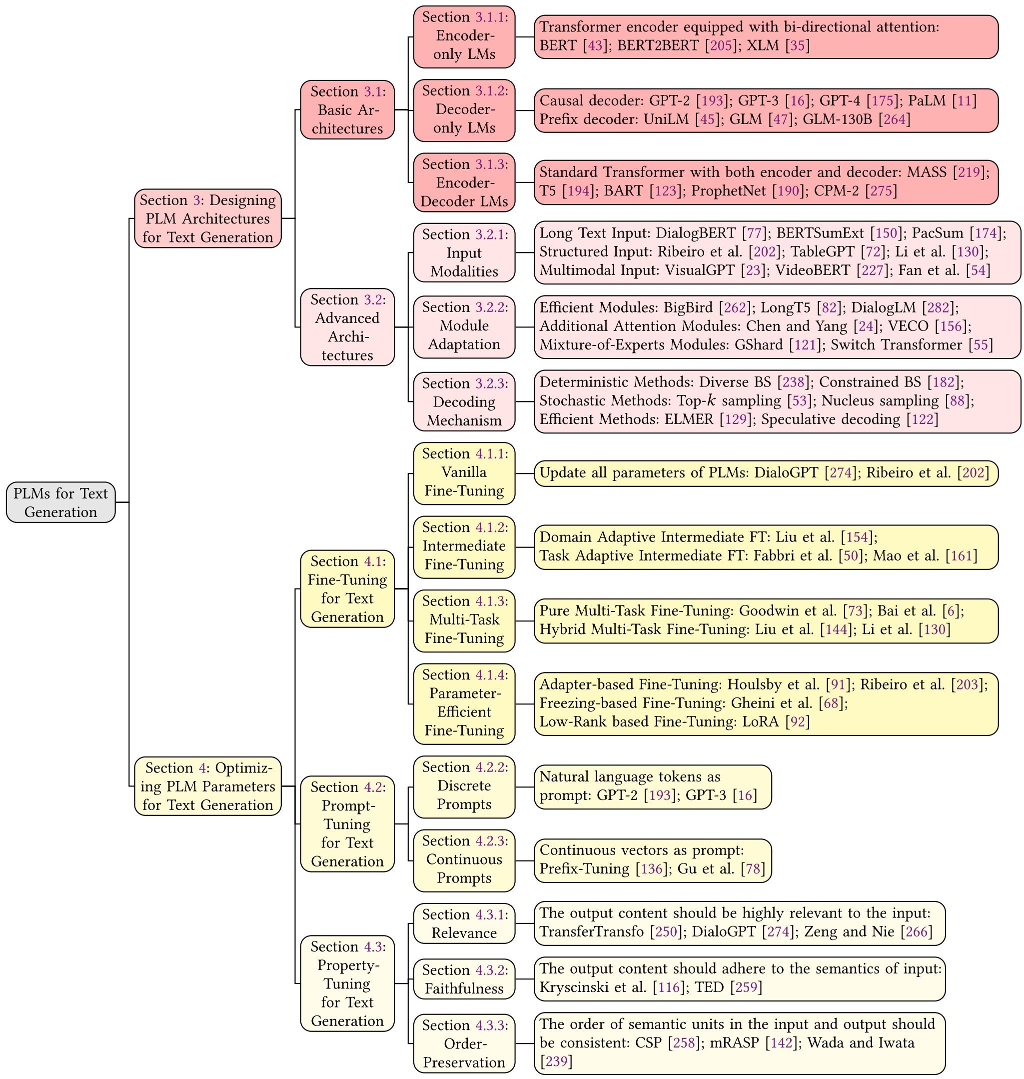

Fig. 1. The main content flow and categorization of this survey. We use color to indicate different sections.

Table 2. Basic Architectures and Their Pre-Training Objectives and Representative PLMs

Architectures	Pre-training Objectives	Pre-trained Language Models
Encoder-only LMs	$-\log \sum_{i\in M} \Pr(x_i \| x_{\setminus M})$	BERT [43], RoBERTa [151], XLM [35], ERNIE [276]
Decoder-only LM	$-\log \sum_{i=1}^{n} \Pr(y_i \| y_{<i})$	GPT-2 [193], GPT-3 [16], UniLM [45], CTRL [110]
Encoder-Decoder LMs	$-\log \sum_{i=1}^{n} \Pr(y_i \| y_{<i}, x)$	MASS [219], T5 [194], BART [123], PropherNet [190]

M denotes a masked position set, and $x_{\setminus M}$ means all the tokens except the ones with the masked position in M.

lower-diagonal attention matrix (i.e., 0s on the lower triangular and diagonal and $-\inf$ on the upper triangular), so that each token in the input and output texts can attend only to its preceding tokens. In contrast, the prefix decoder uses a mixed attention matrix (i.e., bi-directional attention matrix in the input part and lower-diagonal attention matrix in the output part), allowing tokens in the input to attend to each other, while tokens in the output attend to all input tokens and previous tokens.

In the literature, GPT [192] was the first causal LM for the text generation task. Then, GPT-2 [193] explored the transfer capacity of language models for zero-shot generation task. Furthermore, GPT-3 [16] showed that scaling model parameters can significantly improve the downstream generation tasks, with a few examples or prompts. So far, the causal LMs have been widely adopted as the dominated architecture of recent large LMs with over ten billion parameters, such as ChatGPT, GPT-4 [175], and PaLM [11]. In another line, UniLM [45] was the first prefix LM to solve conditional generation tasks. GLM [47] and GLM-130B [264] further scale prefix LM to obtain a bilingual LLM. Existing research has shown that the decoder-only LMs have better zero-shot performance than other architectures without multi-task fine-tuning [242]. CPM [277] and PanGu-α [265] practiced on training large-scale auto-regressive Chinese language models. More importantly, the performance of decoder-only LMs can be improved and indicated by the scaling law [107].

3.1.3 Encoder-Decoder Language Models. Encoder-decoder LMs follow the standard Transformer architecture for text generation, consisting of stacks of both encoder and decoder layers. During pre-training, MASS [219] and ProphetNet [190] took the sequence with one masked segment as the input of encoder and then the decoder generates the masked tokens in an auto-regressive way. T5 [194] randomly replaced several spans in the source text with different special tokens, and then the decoder predicted every replaced span in turn. BART [123] was pre-trained with **denoising auto-encoder (DAE),** i.e., the model learns to recover the original text from corrupted text, which is corrupted with different noising methods, such as sentence permutation and token deletion.

As shown in literature [233], encoder-decoder LMs should be preferred over decoder-only LMs if and only if there is no concern about storage, *i.e.,* parameter counts are generally less important than actual throughput. However, when there is a parameter constraint, the prefix LMs make for a suitable alternative. Tay et al. [233] compared GPT-like (decoder-only) and T5-like (encoder-decoder) models. The empirical analysis revealed that the prefix LMs and the span corruption encoder-decoder model (T5) have gives and takes across different sub-tasks and the decoder-only LM (GPT-like) setup appears to be the worse configuration. However, the recent significant advances in large LMs demonstrate some of the clear advantages of the use of GPT-like decoder-only LMs over other architectures with the increasing of model scales.

3.2 Advanced Architectures

To derive effective PLMs for text generation, most studies proposed to improve the basic architectures of PLMs from three major aspects of input modalities, module adaptation, and decoding mechanism. In this part, we will introduce these improvement advances.

3.2.1 Input Modalities. In many text generation tasks, the input text might be a long document consisting of multiple paragraphs, which is challenging for PLMs to model cross-sentence (paragraph) semantics and capture the most critical semantics [77]. Besides, structured data (e.g., table, graph, and tree) is also a critical kind of input data for text generation. However, there exists a semantic gap between structured data and PLMs since PLMs are typically pre-trained on natural language texts [86]. Finally, multimodal data (e.g., image) has become a critical kind of input for text generation applications, e.g., image captioning [23] and speech recognition [54].

Long Text Input. To capture the semantic dynamics across utterances, several studies [77, 281] proposed to learn document representations in a hierarchical way for subsequent accurate generation. Specifically, Gu et al. [77] represented the long dialogue context using sentence- and discourse-level Transformer encoders to respectively encode each dialogue utterance and the sequence of utterance vectors. On the other hand, long documents may contain complementary, overlapping, or contradictory contents. Therefore, it is necessary to retain the most critical contents and verbalize them in the generated text. To achieve this goal, Nguyen et al. [174] introduced a topic model to capture the global topic semantics of the document and utilized a gate mechanism to integrate the global semantic vectors into the text generation process.

Structured Input. In order to fit the structured input for PLMs, most studies directly linearized the input data into a sequence by concatenating the relational triples of KGs [202] or populating human-written heuristic templates with the attribute-value pair of tables [72]. During the serialization process, it would be better to explicitly provide the structural information for generating faithful text. For example, Ribeiro et al. [202] prepended "⟨H⟩", "⟨R⟩", and "⟨T⟩" tokens before the head entity, relation and tail entity of a KG triple. The semantic gap between structured data and PLMs makes it difficult to effectively inject structured data representations into PLMs while directly serializing structured data. Therefore, some people proposed to align the structured data representations with PLM-based word embeddings in semantic spaces. For example, Li et al. [130] utilized **graph neural networks (GNN)** and PLMs to project KG entities into embeddings, and then minimized the Euclidean distance between the GNN-based and PLM-based entity embeddings.

Multimodal Input. For the image input, Chen et al. [23] proposed an image captioning PLM, called *VisualGPT*. They designed an attention mechanism with **self-resurrecting activation units (SRAUs)**, which balances the multimodal input from the visual encoder and the language input from the decoder layer. For the video input, UniVL [157] employed two single-modal encoders to encode text and video separately and a sentence decoder to generate video captions. For the speech input, to alleviate the data-scarce issue, Fan et al. [54] pre-trained the speech encoder by predicting masked speech feature chunks with its context based on a large amount of unpaired speech. Besides a single modality, fusing multiple modalities as input is important for human perceptual learning [218]. Nagrani et al. [170] proposed a **multimodal bottleneck transformer (MBT)** to fuse audio and vision in videos, in which a tight fusion bottlenecks is used to collect and condense the most relevant inputs in each modality.

3.2.2 Module Adaptation. In addition to adapting PLMs to various input formats, it is also useful to modify the Transformer modules to fulfill diverse requirements. Efficient modules can be utilized to handle the long text input, additional attention modules to gather multiple input information, and **mixture-of-experts (MoE)** can further improve the models. Next, we mainly focus on these module adaptations inside the Transformer architecture for various text generation tasks.

Efficient Modules. To adapt PLMs to long-form text input and alleviate quadratic complexity of full-attention computation, efficient attention modules are proposed to modify the original self-attention modules. A common practice is to limit the view of attention matrix, *i.e.,* every token only attends to specific tokens using predefined or learnable patterns, rather than attending to all other tokens. Local attention allows each token to attend to its neighbors within a certain distance [160, 181, 262]. Global attention chooses special tokens which can see the whole text considering the long-term dependency [82, 181]. Moreover, random attention is proposed to learn non-local interactions [262], while Sinkhorn attention improves the local attention by attending to new blocks using neural sorting [282].

Additional Attention Modules. In practice, many text generation tasks need to process input data from multiple sources. It is common to leverage one or more encoders to encode multiple inputs. Therefore, several works proposed to utilize different strategies to aggregate multi-source inputs in the cross-attention module. Golovanov et al. [71] conducted mean pooling for dialogue history, current state and persona information. Chen and Yang [24] and Liu et al. [147] proposed multi-view attention and knowledge-aware attention to process embeddings from multiple views or knowledge sources. In addition, VECO [156] plugged a cross-attention technique into the Transformer encoder to explicitly build the inter-dependence between multiple languages. Zeng and Nie [267] appended the gating mechanism after self-attention to inject condition-aware information.

Mixture-of-Experts (MoE). Recently, some researchers incorporate MoE [99] into the Transformer architecture to scale the model capacity without increasing training and inference computation. MoE is commonly used to build a multilingual translation system since the multiple experts can learn knowledge from hundreds of languages in various domains. GShard [121] is the first work to extend the idea of MoE to Transformer that replace the dense feed forward network with the position-wise MoE layer with the top-2 gating network. Switch Transformer [55] further simplifies the routing algorithm that only routes to only one expert. The sparse MoE layer can reduce the computation budget while preserving the performance. NLLB-200 [36] tailors for low-resource languages and utilizes sparsely gated MoE to build a universal translation system for 200 languages.

3.2.3 Decoding Mechanism. After encoding the input texts, we need to utilize the PLM to produce desire output text in an auto-regressive manner with specific decoding strategies. A prominent method is greedy search, which selects the word with the highest generation probabilities in each step. Whereas, it faces the problems of repetitive generation and low diversity [88, 238]. The following work, which can be roughly categorized into deterministic decoding (e.g., beam search) and stochastic decoding (e.g., nucleus sampling), attempts to alleviate these issues. At the same time, considering the auto-regressive manner that can only generate one word each step, researchers also explore efficient methods to accelerate the decoding process.

Deterministic Methods. Since greedy search may be stuck in a local maximum and miss the sequence with higher global probability, beam search preserves the n most probable candidates and select the sentence with highest probability in the end. According to prior work [169], beam search is a decent strategy for tasks requiring accuracy, such as machine translation and text summarization. Considering the candidates may present minor variations with each other, diverse beam search [238] induces a dissimilarity term among the candidate sentences to increase the diversity while keeping the generation quality. To further alleviate the repetition issue, Paulus et al. [182] force the model to never produce the same n-grams. In addition, when it comes to the tasks that we can predefine some words that must be included in the output text, constraint beam search can be employed considering the constraint words [187].

Stochastic Methods. Compared to deterministic decoding, stochastic methods randomly sample the next word based on the probability distribution. It is a common option for open-ended generation tasks (e.g., dialogue and story generation) and widely applied in large language models. To mitigate the incoherent generation caused by sampling the word with extreme low probability, top-k sampling [53] only generates the next word with the top-k highest probability. Similarly, *nucleus sampling* [88] (a.k.a., *top-p sampling*) only considers the top words whose cumulative probability exactly exceeds p (e.g., 0.95). Contrastive search [226] further induces a degeneration term to penalize the generation of the word that has similar meaning with previous context. Moreover, typical sampling Meister et al. [164] constrains the sampling space within the expected information to produce more human-like text. Contrastive decoding [135] prefers the token that large LMs assign more probability than small LMs since larger LMs put more mass on desirable token.

Efficient Methods. Besides the accuracy and diversity of the decoding strategies, researchers also focus on speeding up the generation process with little sacrifice on the generation quality. **Compared to auto-regressive (AR)** paradigm which generates texts token-by-token, **non-autoregressive (NAR)** generation [75] is an initial idea to simultaneously produce all tokens in texts in parallel during inference. However, due to the independence assumption, NAR models lags much behind AR models. To bridge the gap between AR and NAR generation, Qi et al. [189] proposed to predict tokens with arbitrary length by utilizing different attention mechanisms. Furthermore, to learn bi-directional dependency during generation, researchers proposed an NAR language model [129], which leverages the early-exit technique to generate tokens at different layers and is pre-trained with a layer permutation language modeling objective. In another line, some work explore efficient decoding methods on existing autoregressive PLMs. Stern et al. [222] introduces a blockwise parallel decoding method that first predicts some words in parallel and then verify the output to correct the bad generation. Considering the difficulties of generation, speculative decoding [122] utilizes small but efficient models to assist the generation of PLMs.

4 OPTIMIZING PLM PARAMETERS FOR TEXT GENERATION

To obtain good performance, it is critical to develop effective optimization algorithms for PLM-based text generation models. We consider three main types of optimization methods, namely fine-tuning, prompt-tuning, and property-tuning. We will detail each optimization method below.

4.1 Fine-Tuning for Text Generation

During pre-training, PLMs are able to capture general linguistic knowledge from large-scale corpora. However, it requires task-specific knowledge to perform downstream text generation tasks. For this purpose, fine-tuning is a popular approach to incorporating task-specific information into PLMs by adjusting their weights using downstream text generation datasets [193].

According to how the parameters of PLMs are updated [106], exiting fine-tuning methods for text generation can be categorized as (1) vanilla fine-tuning, (2) intermediate fine-tuning, (3) parameter-efficient fine-tuning, and (4) multi-task fine-tuning. Compared with vanilla fine-tuning, intermediate and multi-task fine-tuning can alleviate the overfitting issue on small text generation datasets to some extent. As the vanilla fine-tuning requires adjusting the entire model, parameter-efficient methods such as adapters [91] can fine-tune PLMs in a lightweight manner.

4.1.1 Vanilla Fine-Tuning. Vanilla fine-tuning generally updates all parameters of PLMs using downstream text generation datasets with task-specific losses (e.g., cross-entropy loss [193]). Zhang et al. [274] trained the DialoGPT model on the basis of the GPT-2 architecture by modeling a multi-turn dialogue session as a long text and optimizing the generation model with language modeling objective. Ribeiro et al. [202] investigated two recent PLMs, BART and T5, for graph-to-text generation and fine-tuned them using the typical auto-regressive cross-entropy loss. A major issue of vanilla fine-tuning is that it is prone to overfitting on small downstream datasets.

4.1.2 Intermediate Fine-Tuning. The basic idea of intermediate fine-tuning is to incorporate an intermediate dataset consisting of sufficient labeled instances. The intermediate dataset can focus on the same target text generation task but from a different domain, or a similar NLP task from the same target domain. It is helpful to infuse domain- or task-specific knowledge from the intermediate dataset to alleviate the overfitting issue and enhance the performance on small target text generation datasets [184]. According to the relatedness between the intermediate dataset and the target text generation dataset [106], intermediate fine-tuning can be divided into two categories, i.e., **domain adaptive intermediate fine-tuning (DAIFT)** and **task adaptive intermediate fine-tuning (TAIFT)**.

Domain Adaptive Intermediate Fine-Tuning. According to Kalyan et al. [106], DAIFT utilizes an intermediate dataset, which focuses on a similar NLP task (not text generation tasks) from the same target domain, consisting of sufficient labeled instances. By leveraging such an intermediate dataset, PLMs can be enriched with domain-specific knowledge, which is helpful to improve the performance of the target text generation task within the same domain. DAIFT is commonly used in machine translation to eliminate the issue of unseen languages in translation pairs [39, 154]. For example, to improve the translation quality of the low-resource target language (e.g., Kazakh), Liu et al. [154] constructed a large-scale intermediate monolingual corpus of the target language and fine-tuned mBART by reconstructing the corrupted target-language text. The intermediate dataset comes from the same language domain as the target dataset (e.g., Kazakh), which can impart language-related linguistic knowledge to PLMs for a better translation performance.

Task Adaptive Intermediate Fine-Tuning. In contrast with DAIFT, TAIFT incorporates an intermediate dataset on the same target text generation task but from a different domain. It aims to infuse task-specific knowledge from the massive intermediate labeled dataset for improving the same target text generation task. For example, Maurya et al. [163] first fine-tuned mBART [149] with an intermediate task using monolingual data of three languages. The objective function of the intermediate task is close to the target tasks which enriches the multi-lingual latent representation of mBART and provides good initialization for target tasks. It has been shown that additionally training PLMs on the general domain text corpora (e.g., Wikipedia, WebText) on the same text generation task can improve the performance on a specific domain (e.g., Movie) [50, 161]. For example, Fabbri et al. [50] performed summarization on intermediate pseudo-summaries created from Wikipedia to improve the zero-shot and few-shot performance of abstractive summarization in CNN-DailyMail dataset. Furthermore, several researchers combined DAIFT and TAIFT in practice by dividing the parameters of PLMs into domain-related and task-related parameters and then using DAIFT and TAIFT to optimize them respectively [147].

4.1.3 Multi-Task Fine-Tuning. Multi-task fine-tuning can exploit cross-task knowledge to improve the primary text generation task by incorporating auxiliary tasks. Furthermore, by obtaining knowledge from related NLP tasks, multi-task fine-tuning can enhance the robustness of PLMs and reduce the need for large amounts of labeled instances in the text generation task. According to the similarity between the primary text generation task and auxiliary tasks, **multi-task fine-tuning (MTFT)** can be divided into two categories, i.e., pure MTFT and hybrid MTFT.

Pure Multi-Task Fine-Tuning. Pure MTFT incorporates auxiliary tasks that are the same as the primary text generation task but from different domains. Previous studies mainly utilized extra datasets to eliminate the data scarcity issue of the primary generation task [6, 73]. Specifically, Goodwin et al. [73] leveraged twenty-one additional summarization datasets to improve zero-shot summarization on unseen datasets. Bai et al. [6] used an auxiliary monolingual summarization task to improve the primary cross-lingual summarization task in a low-resource language.

Hybrid Multi-Task Fine-Tuning. Hybrid MTFT incorporates auxiliary tasks that are different from the primary text generation task. These diverse auxiliary tasks can enhance the primary generation task in different aspects. For example, Liu et al. [144] and Jin et al. [103] fine-tuned PLMs with auxiliary tasks (e.g., coherence detection, style-carrying text reconstruction) to control the content of the generated text according to the topic change and text style (humor, romance, and clickbait). Besides, to improve the faithfulness of the generated text, Li et al. [130] and Gong et al. [72] introduced auxiliary input data reconstruction tasks to reconstruct KG triples and table values for aligning the input information with the generated content.

4.1.4 Parameter-Efficient Fine-Tuning. As the above fine-tuning methods require updating all PLM parameters, it is time-consuming to perform the entire fine-tuning in resource-limited scenarios. Many studies developed **parameter-efficient fine-tuning (PEFT)** for text generation tasks by freezing most parameters of PLMs and updating a small account of parameters.

Adapter-Based Parameter-Efficient Fine-Tuning. Adapter is a special neural layer proposed by Houlsby et al. [91] to fine-tune PLMs in a parameter-efficient way. Specifically, the adapters first project the original d-dimensional features into a smaller dimension, m, apply a non-linearity, then project back to d dimensions. The total number of parameters added per layer, including biases, is $2md + d + m$. By setting $m \ll d$, we can limit the number of additional parameters per task. Thus, it is highly efficient to fix the parameters of original PLMs but only fine-tune the adapters [32, 223].

To address the inefficiency and overfitting issues in low-resource abstractive summarization, Chen and Shuai [32] inserted adapters into both encoder and decoder of PLMs and only fine-tuned the adapters. A number of studies have shown that adapters can help PLMs efficiently capture some input characteristics for generating more accurate output text with a low extra cost in terms of parameters [119, 203]. For example, Ribeiro et al. [203] utilized adapters to model the input graph structure effectively when fine-tuning PLMs on graph input.

Freezing-Based Parameter-Efficient Fine-Tuning. This approach refers to freezing most parameters and only updating a small proportion of PLM parameters. Recent studies have shown that not all parameters of PLMs are necessary to be fine-tuned for text generation tasks, and some of them can be fixed during fine-tuning without large impact on the model performance. Several studies also revealed that cross-attention (or encoder-decoder attention) layers are more important than self-attention layers when fine-tuning PLMs for machine translation [68, 260]. Therefore, Gheini et al. [68] only fine-tuned cross-attention layers while kept the encoder and decoder fixed. This approach achieved comparable translation performance to fine-tuning all parameters. Besides, Stickland et al. [223] froze most of the model parameters and added extra positional embeddings when fine-tuning BART on English monolingual data.

Low-Rank-Based Parameter-Efficient Fine-Tuning. **Low-rank adaptation (LoRA)** [92] is a technique that freezes the pre-trained model weights and injects trainable rank decomposition matrices into each layer of the Transformer architecture. For a pre-trained weight matrix $\mathbf{W} \in \mathbb{R}^{m \times n}$, the basic idea of LoRA is to constrains the update of weight by representing the change with a low-rank decomposition $\Delta\mathbf{W} = \mathbf{A} \cdot \mathbf{B}^\top$, where $\mathbf{A} \in \mathbb{R}^{m \times k}$ and $\mathbf{B} \in \mathbb{R}^{n \times k}$ are the trainable parameters for task adaptation and $k \ll \min(m, n)$ is the reduced rank. One of the primary advantages of LoRA lies in its ability to significantly reduce memory and storage usage (e.g., VRAM). Furthermore, this approach enables the retention of a single large model instance, alongside multiple task-specific low-rank decomposition matrices. This flexibility allows for effective adaptation to various downstream text generation tasks [3, 271].

4.2 Prompt-Tuning for Text Generation

Most generative PLMs are pre-trained using language modeling objectives but fine-tuned on text generation tasks with task-specific objectives. To alleviate this discrepancy between pre-training and fine-tuning, prompt learning [145] is proposed to reformulate the downstream tasks (text generation tasks) into the language modeling task as pre-training.

4.2.1 Background. According to Liu et al. [145], a prompt function $f_{prompt}(\cdot)$ converts the input text x into a prompt $x' = f_{prompt}(x)$ through a two-step process:

(1) Apply a textual *template* containing two slots: an input slot $[X]$ for input x and an answer slot $[Z]$ for an intermediate generated answer text z that will later be mapped into y.
(2) Fill the input slot $[X]$ with the input text x.

Here the prompt can be *cloze* or *prefix* style. The cloze-style prompt is usually adopted in language understanding tasks, where the empty slot $[Z]$ is either in the middle of the prompt or at the end. For example, in sentiment analysis where $x =$"*I love this movie*", the template may take a clozed form such as "$[X]$ It was a really $[Z]$ movie." to predict the answer in $[Z]$. While in the prefix-style prompt, the input text comes entirely before the empty slot $[Z]$ such as "English: $[X]$ German: $[Z]$" in machine translation. Prefix prompts are widely used in text generation, as they mesh well with the left-to-right nature of language modeling. In the above prompt examples, the template is composed of *discrete* natural language tokens, but the tokens can also be virtual words (e.g., represented by numeric IDs), which would be mapped into *continuous* embeddings later.

4.2.2 Discrete Prompts. Early prompting studies create prompts by manually designing templates based on human introspection. As a pioneering study, GPT-2 [193] performed text generation tasks using various manually created prompts. For example, the prompt "translate to french, [*input*], [*output*]" is used in machine translation. The prompt defines the semantic mapping from input data to output text in a specific text generation task. By utilizing diverse prompts, a single PLM is able to perform a number of different text generation tasks. These approaches heavily relied on manual efforts to create prompts; but PLMs are highly sensitive to prompts: improperly created prompts lead to low performance [100]. To avoid the need to manually specify prompts, several studies proposed to automatically generate prompts by searching tokens in discrete space [215], paraphrasing existing prompts [100], and generating prompts using PLMs [62]. In discrete prompts, there is no need for any model training and the parameters of PLMs will not be updated.

4.2.3 Continuous Prompts. Continuous prompts (a.k.a., soft prompts), consisting of embedding vectors, are widely explored for text generation tasks. Two major advantages are expected: (1) relaxing the constraint that the prompt template should be natural language words; (2) removing the restriction that the template is parameterized by PLMs' parameters. Instead, continuous prompts have their own parameters that will be optimized based on training data of the text generation tasks. The most well-known method using continuous prompts for text generation is prefix-tuning [136], which freezes the generative PLMs (e.g., GPT-2, BART) and optimizes a sequence of task-specific vectors (called *prefix*). In contrast to full-parameter fine-tuning, which requires storing a tuned copy of the model for each text generation task, prefix-tuning only optimizes the prefix for each text generation task. Similar to prefix-tuning, several studies used continuous prompts to solve other text generation tasks such as dialogue generation [78]. Notably, continuous prompts like prefix-tuning is considered as a type of parameter-efficient fine-tuning (PEFT) [87].

4.3 Property-Tuning for Text Generation

For different generation tasks, we need to consider specific language properties when tuning PLMs. In this section, we discuss three major properties that are widely desired for text generation.

4.3.1 Relevance. According to the linguistic literature [132], in text generation, *relevance* means that the topical semantics conveyed in output text is highly related to the input text. As a representative example, in dialogue systems, the generated responses should be relevant to the historical utterances and other conditions, such as speaker persona and discourse topic.

Compared with traditional neural generative models, PLMs utilize more powerful multi-layer cross-attention mechanism to model the semantic associations between input and output, which can enhance the relevance of generated text to the input data (*e.g.,*the dialogue systems [250, 274]). A good example is DialoGPT [274] based on an auto-regressive language model GPT-2. Specially, DialoGPT was first trained on large-scale dialogue pairs/sessions, which could enable DialoGPT to capture the joint distribution of $\Pr(history, response)$ in conversational flow for generating relevant responses to the history utterance. Furthermore, Zeng and Nie [266] proposed a TF-IDF based masked language modeling objective, which aims to generate the masked condition-related tokens rather than general language patterns. Besides, they adopted a non-parametric attention-based gating mechanism to switch between generating a general word or a condition-related word.

4.3.2 Faithfulness. *Faithfulness* is also an important language property to consider for text generation, which means the generated content should adhere to the semantics of input text. For example, text summarization must generate faithful text conveying the salient information of the input text. Faithfulness sometimes refers to the fact that the generated text is in accord with world facts.

Table 3. Summary of Major Challenges in the Three Aspects and Existing PLM-Based Solutions

Aspect	Challenge	Solution
Data	Lacking Enough Training Data	Knowledge transfer [147, 183, 288], data augmentation (e.g., model synthesis [44, 180], pertuebation [25, 158], multi-task learning [6, 73].
	Data Bias in Pretraining Corpus	Counterfactual data augmentation [287], dropout regularization [247], prompt-based self-debias [83, 208].
Model	Compression	Quantization by truncating PLMs weights [42, 59], pruning less critical weights [52, 74, 81, 90], knowledge distillation [31, 102, 127].
	Enhancement	World knowledge [229, 285], efficient framework [101, 285].
	Scaling	Large-scale training with 3D parallelism [97, 216], specialized adaptation [94], long text modeling [28, 224].
Optimization	Satisfying Text Properties	Enhance coherence [93, 132], preserve factuality [33, 46, 130, 172], improve controllablity [41, 111, 179].
	Mitigating Tuning Instabilities	Intermediate fine-tuning [154, 184], adaptive weight learning [91, 113], supervised contrastive learning [80].
	Aligning with Humans	Outcome-supervised RLHF [171, 177], process-supervised RLHF [139, 236].

To generate faithful texts, PLMs should be able to accurately understand the core semantics of input and acquire sufficient world knowledge for solving the downstream task. It has been shown that PLMs have excellent natural language understanding capacities in capturing core semantics from plain text [43], and they indeed encode a large amount of world knowledge [100]. For example, Kryscinski et al. [116] utilized a contextual network in the PLM decoder to retrieve the most salient parts from the source document to improve the level of faithfulness of generated summaries. Besides, several studies proposed to generate faithful texts by introducing additional tasks (losses) besides the basic text generation [205, 259]. Specifically, Yang et al. [259] fine-tuned PLMs through a theme modeling loss which aims to make the generated summary semantically close to the original article for achieving faithful generation. While, Liu et al. [153] proposed that different tasks can interact with each other and their information can be exchanged to perform faithful speech recognition.

4.3.3 Order-Preservation. In the NLP field, *order-preservation* is a special property that refers that the order of semantic units (word, phrase, etc.) in both input and output text is consistent. Such a property is key to several important text generation tasks, such as text paraphrasing and machine translation. In machine translation, it often requires preserving some order of phrases in the source and target language for ensuring the accuracy of the translation results.

In machine translation, word alignment is an extensively studied approach to achieve the order-preservation property. A representative study is **Code-Switching Pre-training (CSP)** [258]. CSP first automatically extracted the word-pair alignment information from the source and target monolingual corpora and then continually pre-trained PLMs by predicting the sentence fragment on the source side given the aligned fragment in the target language. Moreover, to relax the restriction of discrete word alignment, another line of research aims to conduct continuous representation alignment to improve the order-preservation property [142, 239]. For example, Wada and Iwata [239] focused on aligning word representations of each language by mapping word embeddings of each language into a common latent space.

5 CHALLENGES AND SOLUTIONS

The paradigm of PLM-based text generation involves three key components, namely data, model, and optimization. In this section, we further discuss the major challenges in each of the aspects and possible solutions. A summary of these challenges and solutions is presented in Table 3.

5.1 Data Challenges

We first discuss the challenges and solutions related to the data.

5.1.1 Lacking Sufficient Training Data. In a number of text generation tasks, it is difficult to obtain sufficient annotated data. Transfer learning provides an effective solution by transferring the knowledge of data-rich source tasks into data-scarce target text generation tasks. Besides, data augmentation and multi-task learning can be also used to address this problem.

Transfer Learning. To deal with the data scarcity issue, several studies proposed first fine-tuning PLMs on large amounts of external labeled corpora and then transferring into data-scarce target text generation tasks [147, 183, 288]. In particular, Peng et al. [183] and Zou et al. [288] first fine-tuned PLMs on substantial labeled dialog/summary data and then fine-tuned for the target dialog/summarization task in a new domain with limited labeled data. Similarly, Liu et al. [147] first trained models on large-scale ungrounded dialogs and unstructured knowledge base separately to improve the low-resource knowledge-grounded dialog generation task.

Data Augment. In recent literature, data augmentation has emerged as a critical method for increasing the amount of data by adding slightly modified copies of already existing data or newly created synthetic data from existing data. One line of research is to use another model/tool to synthesize the augmented data [44, 180]. For example, Pasunuru et al. [180] used a search engine, i.e., Bing, to retrieve the answer paragraph as the synthetic summary and used the top ranked documents as input text. Another line of work is to use perturbation-based methods by corrupting the original text [25, 158]. Chen and Yang [25] presented a set of perturbation methods for conversational summarization, such as randomly swapping or deleting utterances in conversations.

Multi-Task Learning. Leveraging other data-rich tasks and datasets can also overcome the data scarcity issue. Most studies usually incorporated similar auxiliary generation tasks for enhancing the primary text generation task [73]. However, these methods usually adopt independent decoders for each task, thus breaking the semantic connections between high- and low-resource text generation tasks. To bridge this gap, Bai et al. [6] employed a unified decoder which learns the alignments and patterns across multiple languages in machine translation.

5.1.2 Data Bias from Pre-Training Corpora. In sociology, bias is an unjustified prejudice in favour of or against a person, group, or thing [65]. PLMs are generally trained using real-world data, so they eventually inherit the biases and stereotypes that are common in the data [65]. These biases can result in unexpected ethical issues in downstream text generation tasks [16].

Counterfactual data augmentation (CDA) is a data-based debiasing strategy that has been used to mitigate gender bias [287]. CDA usually re-balances a corpus for training by “swapping” bias attribute words (e.g., he/she) in a dataset. From a model perspective, researchers explored using dropout regularization to mitigate bias through increasing the dropout ratio for PLM’s attention weights and hidden activations and performing additional pre-training [247]. However, these approaches require modifying the training corpus (which might be secret for the public) or altering PLM’s internal representations or parameters. Therefore, Self-Debias [208] is a post-hoc debiasing technique that first uses hand-crafted prompts to encourage a PLM to generate biased texts and then scales down the probabilities of biased tokens. Similarly, Auto-Debias [83] searches for biased prompts from PLMs and then probes the biased content with such prompts to correct them.

5.2 Model Challenges

In this section, we present the challenges and solutions from the architecture design.

5.2.1 Model Compression. Although PLMs have achieved great success on text generation, the backbone Transformers are still bulky and resource-hungry, resulting in high memory consumption, computational overhead, and energy cost. To address these issues, more and more approaches are proposed to compress PLMs [61], such as quantization, pruning, and knowledge distillation.

Quantization. Quantization refers to the mapping process from floating-point numbers to integers [69]. There are generally two major quantization approaches, namely *quantization-aware training (QAT)* (requiring additional full model retraining) and *post-training quantization (PTQ)* (requires no model retraining). Due to a much lower computational cost, PTQ is more preferred than QAT for PLMs. As shown in [42], extreme large values occur in hidden activations (called *outliers*) when PLMs reach 6.7B size or above. Therefore, LLM.int8() [42] is proposed to separate the dimensions of outliers and the rest of the dimensions in matrix multiplication and then the two parts are computed with 16-bit floating numbers and 8-bit integers, respectively. Considering that weights are easier to be quantized than activations, SmoothQuant [252] has been proposed to migrate the difficulty from activations to weights. To find optimal quantized weights that minimize a layerwise reconstruction loss, GPTQ [59] improves the original optimal brain quantization [58] method by fixing the quantization order of weights for all rows.

Pruning. Pruning refers to identifying and removing redundant and/or less important weights [61]. Pruning methods for text generation fall into two categories [61]. First, unstructured pruning prunes individual weights by locating the set of least important weights in PLMs. The importance of weights can be measured by specific metrics such as absolute values [74] and gradients [81]. Second, structured pruning prunes structured blocks of weights or even complete components of PLMs by reducing and simplifying certain modules such as attention heads [90] and Transformer layers [52].

Knowledge Distillation. Knowledge distillation refers to training a smaller model (called the *student*) using the output of PLMs (called the *teacher*). First, the student model can directly learn from the output word distribution of the final softmax layer in PLMs, which allows the student to mimic the generated text of the teacher by replicating the word distribution across the whole vocabulary [31]. Second, the student can also learn from the output tensors of PLMs encoders [127]. Intuitively, the representations of PLMs encoder may contain meaningful semantics and contextual relationships between input tokens, which is helpful for generating accurate text. Third, by replicating attention distributions between input data and output text, the student can also learn the contextual dependency between input and output [102].

5.2.2 Model Enhancement. Although PLMs have achieved great success nowadays, they are still far from our expectations. Recently, there has been a surge of interest in the research community to strengthen existing PLMs through injecting knowledge or building efficient training framework.

Knowledge-Enriched PLMs. Recent work has found that integrating knowledge from external knowledge sources can enhance the text generation performance of PLMs [229, 285]. Specifically, ERNIE 3.0 [229] was pretrained on a 4-TB corpus consisting of plain texts and a large-scale knowledge graph for both language understanding and generation tasks. Without injecting explicit knowledge, CALM [285] can encode commonsense knowledge into parameters by learning to write and reason with concepts via pre-training strategies, yielding better performance on text generation tasks.

Efficient PLMs. Pre-training PLMs on large-scale text data is prohibitively expensive. Recently, it has been demonstrated that by meticulously structuring the model architecture, it is possible to obtain equivalent or higher text generation performance with less pre-training data [285] or lower pre-training costs [101]. For example, CALM [285] developed a mutually reinforced pre-training framework with generative and contrastive objectives, thus achieving comparable results to other larger PLMs such as T5 while only being pre-trained on a small corpus for a few steps.

5.2.3 Model Scaling. Kaplan et al. [107] have shown that the performance of PLMs can be boosted by scaling up the amount of PLMs' parameters. This observation sparked the development of large-scale PLMs (a.k.a., **large language models, LLMs**) in text generation [11, 16]. The most representative LLM is GPT-3 [16], which contains 175 billion parameters, 10× more than any previous PLMs. The large-scale parameters endow GPT-3 the emergent ability [248] to perform text generation tasks only using a few examples without any gradient updates, which is known as in-context learning.

Large-Scale Training. With the parameter scaling of PLMs, the training loss spike is also more likely to appear. To address this issue, several studies proposed to restart the training process from an earlier checkpoint before the spike [11] and shrink the embedding layer gradients [47]. Moreover, to train PLMs under limited computational resource, many efficient and scalable techniques are proposed, exemplified by 3D parallelism [97, 216], ZeRO [195], mixed precision training [166], and LoRA [92]. Especially, 3D parallelism usually includes data parallelism, pipeline parallelism, and tensor parallelism to train large-scale models in a parallel way. Focused on memory redundancy in data parallelism, ZeRO aims to retain only a fraction of data on each GPU, while the rest of the data can be retrieved from other GPUs when required. LoRA exerts the low-rank constraint for approximating the update matrix at each layer to reduce the trainable parameters.

Specialized Adaptation. Researchers have shown great interest in the challenging text generation across various expressions, specific languages, and specialized domains due to its widespread application. Huang et al. [94] proposed a generic yet effective template for assigning LLMs to the roles of diverse task experts to perform various generation tasks in multiple languages. Moreover, several work leverages instruction tuning for LLMs to acquire domain knowledge in an efficient manner. Med-PaLM [217] instruction-tunes the model using clinical data and make it capable to answer medical questions. InstructCTG [284] verbalizes constraints into manual instructions and further fine-tunes an LLM to perform controlled text generation in various expressions.

Long Text Modeling. Early PLMs (e.g., GPT-2 [193]) usually adopt absolute **position embedding (PE)** such as sinusoidal and learned PE. However, more and more studies employ relative position embedding [194, 257], which show better generalization to texts longer than those texts for training. Certain position embeddings have shown the capacity to generalize to text beyond the training length, referred to as *extrapolation capability*, including ALiBi [188], xPos [228], and even NoPE [109]. Due to the excellent performance and the long-term decay property, RoPE [225] has been widely adopted in the latest PLMs [11, 234], To scale RoPE to longer texts, a method of position interpolation [28] downscales the unseen position indices within the original length by multiplying all position indices with a coefficient, but this way might hurt the model performance on shorter texts. Moreover, ReRoPE and LeakyReRoPE [224] introduce *position truncation*, where position indices within a pre-defined window are kept, while indices beyond the window are either truncated to the pre-defined length or interpolated to align with the maximum training length.

5.3 Optimization Challenges

In this part, we discuss challenges and solutions about the optimization of PLMs for text generation.

5.3.1 Satisfying Special Text Properties. In Section 4.3, we introduced three basic text properties. In this section, we will present three more difficult properties for text generation tasks, i.e., coherence, factuality, and controllability.

Coherence. In linguistics [125], language coherence is what makes a multi-sentence text meaningful, both logically and syntactically. An essential technique to improve coherence is to elaborately plan the generated content, known as text planning [93, 132, 244]. For example, Li et al. [132] designed a text generation model based on a two-level text plan: (1) the document plan is modeled as a sequence of sentence plans in order, and (2) the sentence plan is modeled as an entity-based subgraph from KG. The local coherence is naturally enforced by KG subgraphs, and the global coherence can be improved by generating a coherent sequence of subgraphs. For video captioning, Wang et al. [243] proposed to segment the video into a number of event pieces under the holistic understanding of the video, which can effectively increase the coherence of generated caption.

Factuality. The input data (e.g., infobox) for text generation tasks (e.g., table-to-text generation) usually contains some factual information. In such cases, the generated content should adhere to the original input facts. However, lacking direct access to the input facts or explicit supervision makes PLMs unable to retain text factuality in generation process. For data-to-text generation, the pointer generator [211] is usually adopted to copy the input facts into output for preserving factuality [33, 130]. Furthermore, in text summarization, some studies proposed evaluation metrics or correction methods to measure and revise the generated text for preserving factuality [46, 172].

Controllability. In text generation, many applications need a good control over the output text. For example, to generate reading materials for kids, we would like to guide the output stories to be safe, educational, and easily understandable by children. The **Plug and Play Language Model**, also known as **PPLM** [41], is an example of a controllable PLM that combines a PLM with one or more simple attribute classifiers that direct text generation without further PLM training. Several studies achieved controllablility from a distributional view [111, 179]. Pascual et al. [179] described a plug-and-play decoding approach in a single sentence: given a topic or keyword, the model adds a shift to the probability distribution over the vocabulary towards semantically similar words.

5.3.2 Mitigating Tuning Instabilities. Due to the catastrophic forgetting nature of PLMs and the small size of text generation datasets, tuning PLMs for text generation is usually unstable, i.e., fine-tuning the model with different random seeds results in a wide variance of performance. The possible solutions include intermediate fine-tuning, mixout and using supervised contrastive loss.

Intermediate Fine-Tuning. Recent studies have shown that first training PLMs on data-rich intermediate labeled datasets (e.g., a similar NLP task from the same target domain) before fine-tuning them on data-scarce target text generation tasks can achieve better performance in target tasks [154, 184]. For example, Liu et al. [154] constructed an intermediate monolingual corpus of the target language (e.g., Kazakh) and fine-tuned mBART to reconstruct the corrupted monolingual text for improving the translation quality of the low-resource target language.

Adaptive Weight Learning. To avoid the catastrophic forgetting issue in continual learning, Kirkpatrick et al. [113] introduced **elastic weight consolidation (EWC)**, which adaptively slows down the learning of certain weights for PLMs based on how important they are to previously seen tasks, so that the knowledge of previous tasks can be protected during new learning. Moreover, adapter [91] is also an effective solution for catastrophic forgetting, which only updates the task-specific adapters without changing the original PLMs.

Contrastive Learning. The most used cross-entropy loss in text generation, i.e., the KL-divergence between one-hot vectors of labels and the distribution of model's outputs, lacks robustness to noise labels [278] or adversarial examples [49]. Thus, fine-tuning PLMs with cross-entropy loss tends to be unstable, especially when labeled data is limited. An effective solution is to capture the similarity between examples in one class and contrast them with examples in other classes [80]. To this end, Gunel et al. [80] combined the cross-entropy loss with a supervised contrastive learning loss that pushes the words from the same class close and the words from different classes further apart.

5.3.3 Aligning with Humans. To optimize PLMs for real-world deployment, the most important consideration is to align the behaviors of PLMs with human expectations [177]. Recently, there has been a growing focus on the development of diverse criteria aimed at governing the behaviors of

PLMs in generating texts. Among them, *helpfulness*, *honesty*, and *harmlessness* are three commonly used alignment criteria [4]. To be helpful, the generated texts should assist users in solving their tasks in a concise and efficient manner. Besides, an honest PLM should generate accurate content to users instead of fabricating information. Finally, the generated texts should not contain offensive or discriminatory content that may result in potential ethical issue to society.

Reinforcement learning from human feedback (RLHF) [177] has been proposed to fine-tune PLMs with human feedback data. The RLHF technique involves three steps: supervised fine-tuning, reward model training, and RL fine-tuning. For example, InstructGPT [177] first collected a human-written supervised dataset to fine-tune 175 B GPT-3, then trained another 6 B GPT-3 as reward model (RM) using the response ranking as label, and finally optimize the 175 B GPT-3 against the RM using the PPO algorithm [209]. Following this method, WebGPT [171], GopherCite [165], and Sparrow [70] has also been aligned with human expectations. These studies are focused on outcome-supervised RLHF, which aims to assess the quality of the whole text after generated by LLMs. In contrast, process-supervised RLHF [139, 236] evaluates each individual component (e.g., sentence, word, or reasoning step) within the generated text, which can provide fine-grained supervision signals to guide the training. More details about RLHF can be accessed in Zhao et al. [280]'s survey.

6 EVALUATION AND RESOURCES

In this section, we will discuss several commonly used evaluation metrics and resources with respect to PLMs for text generation.

6.1 Evaluation

With the growing variety of text generation applications and datasets, there are several advantages of automatic evaluation: it is potentially much cheaper and quicker than human evaluation and repeatable [10]. However, we should be aware of that most automatic metrics have weaker correlation with human evaluation [214] and certain aspects of texts such as fluency are hard to be evaluated by automatic metrics [34]. For this reason, human evaluation is still viewed as the most important form of evaluation for text generation and held as the gold standard when developing new automatic metrics. In this part, we mainly focus on automatic evaluation metrics and present four categories of metrics, i.e., n-gram overlap metrics, neural-based metrics, diversity metrics, and logit-based metrics. We list the metrics used in each text generation task in Table 1.

6.1.1 N-Gram Overlap Metrics. These metrics measure the degree of word "matching" between machine-generated and ground-truth texts at the word level.

BLEU. The **Bilingual Evaluation Understudy (BLEU)** [178] is proposed as the machine translation metric by comparing a candidate translation of text with one or more reference translations. BLEU-n measures the precision of the co-occurrences of n-grams between the generated and real text and conducts length penalty on shorter generated text. BLEU has been successfully applied to short text generation like sentence-level machine translation, since it is fast and easy to calculate. While, for those text generation tasks involving contextual understanding and reasoning (e.g., story generation), BLEU neglects semantic meaning and sentence structure and shows limitations in morphologically complex languages [162].

ROUGE. **Recall-Oriented Understudy for Gisting Evaluation (ROUGE)** [140] is a set of metrics for measuring automatic summarization of long texts consisting of multiple sentences. ROUGE-n counts the F1 score of the overlapping n-grams between generated and ground-truth texts. ROUGE-L measures the longest matching sequence of words using longest common sub-sequence. Compared to BLEU, ROUGE focuses on recall instead of precision and is more interpretable than BLEU. However, for long text generation tasks, ROUGE's reliance on *n*-gram matching fails to provide insights into the coherence of narrative flow, grammatical accuracy, or topical continuity exhibited by the generated text [112].

METEOR. The **Metric for Evaluation of Translation with Explicit ORdering (METEOR)** [7] is proposed to address some issues found in BLEU. Compared to BLEU, METEOR is computed based on the harmonic mean of the unigram precision and recall, and measures word-to-word matches between generated and real text based on WordNet. METEOR has been found to demonstrate a strong correlation with human evaluations at the sentence or segment level because it is explicitly designed to compare at the sentence level rather than the corpus level [2]. Therefore, METEOR has been widely used in short text generation such as image captioning and question generation. Since it is also based on exact word matching, METEOR still suffers from reference translation variability.

ChrF. **Character *n*-gram F-score (ChrF)** [186] is an automatic evaluation metric for machine translation. Unlike the word level co-occurrence of BLEU, ChrF is mainly focused on the character-level matching so as to consider morpheme overlapping. To improve the correlations with direct human assessments, word unigram and bigram are added to ChrF, which is called *ChrF++*. Recent studies [12] have shown that ChrF is especially suitable for morphologically rich target languages. Besides, ChrF is fast, does not require any additional tools or information, language- and tokenization-independent, and it correlates very well with human relative rankings [20].

6.1.2 Neural-Based Metrics. The above metrics do not require training and mostly assume that the generated text has significant *n*-gram overlap with the reference text. However, this assumption does not hold for text generation tasks that have multiple diverse and plausible outputs. Therefore, metrics based on neural models are proposed to mimic human judges.

BERTScore. Given the excellent performance of BERT across many tasks, BERTScore [272] leverages the pre-trained contextual embeddings from BERT and compares words in candidate and reference texts by cosine similarity. BERTScore has proven to correspond well with human judgments on sentence-level and system-level evaluations [22]. Compared to *n*-gram based metrics, BERTScore can robustly match semantically correct paraphrases and capture distant dependencies and semantically critical ordering changes.

COMET. **Cross-lingual Optimized Metric for Evaluation of Translation (COMET)** is a neural framework for training highly multilingual and adaptable machine translation evaluation models that can function as metrics [199]. COMET is trained in a supervised manner using the segment-level scores of human judgments from WMT, such as the **direct assessments (DA), human-mediated translation edit rate (HTER),** and **multidimensional quality metrics (MQM)**. COMET utilizes two training objectives: estimator modeling and translation ranking modeling. The estimator model is trained to minimize the mean squared error between the predicted scores and quality assessments. The translation ranking model is trained to minimize the distance between the "better" hypothesis and the "anchors" (source and reference) using the triplet margin loss. Notably, COMET can even work for specific languages in machine translation tasks such as IndicCOMET [206].

BLEURT. **Bilingual Evaluation Understudy with Representations from Transformers (BLEURT)** is also a BERT-based machine-learned evaluation metric [212]. To train the evaluation model, BERT is first fine-tuned on synthetic sentence pairs, which are generated using automatic evaluation metrics such as BLEU based on random perturbations. Then, the model is further fine-tuned on system-generated outputs and human-written references using human ratings and automatic metrics as labels. The fine-tuning on synthetic sentence pairs constitutes a crucial stage owing to its ability to enhance the robustness of generation systems against quality drifts.

6.1.3 Diversity Metrics. Lexical diversity is desirable in many text generation tasks, such as dialogue systems and story generation. For these tasks, it is necessary to conduct diversity evaluation.

Distinct. Distinct-n measures the degree of diversity by calculating the number of distinct n-grams in generated text [124]. This metric is scaled by total number of generated tokens to avoid favoring long sentences. However, the scaling method has significant potential to assign higher penalties to longer sequences [146]. Moreover, the distinct value cannot stay invariant and presents a sharp decrease with increasing utterance length.

Self-BLEU. Self-BLEU is proposed as a corpus-level diversity metric [286], which calculates a BLEU score for every generated sentence by treating the other generated sentences as references and then averages these BLEU scores. The lower the self-BLEU score, the higher the diversity of the generated text. Many studies have demonstrated that self-BLEU achieves good generation diversity [27, 286]. However, self-BLEU has limitations in generating diverse output and detecting mode collapse in text generation with GANs [213].

Logit-Based Metrics. In text generation, the probability of a generated text $y = \langle y_1, \dots, y_n \rangle$ can be formulated as $\Pr(y) = \prod_{j=1}^{n} \Pr(y_j | y_{1:j-1}; x)$, where x is the input data, and $y_{1:j-1}$ is the previous tokens. Logit-based metrics evaluate the generated text from a probabilistic view.

NLL. **Negative log-likelihood (NLL)** is originally introduced in SeqGAN [261] to tell how good the generated data is fitted by the oracle language model. In SeqGAN, a randomly initialized LSTM is regarded as a true model, and the text generation model needs to minimize the average negative log-likelihood of generate data on oracle LSTM, *i.e.,*$\mathbb{E}_{y \sim q} \log \Pr(y)$, where y denotes the generated text. Since an LSTM is regarded as a true model, NLL can calculate the average loss on every sentence, word by word:

$$\mathrm{NLL}(y) = -\mathbb{E}_{y \sim G_\theta} \sum_{j=1}^{n} \log(G_{oracle}(y_j | y_{1:j-1})), \tag{3}$$

where G_{oracle} denotes the oracle LSTM, and G_θ denotes the generative model.

Perplexity. In information theory, **perplexity (PPL)** is a measurement of how well a probability distribution or probability model predicts a sample compared with the ground-truth [14]. A low perplexity indicates the probability distribution is good at predicting the sample. Therefore, the perplexity of the discrete probability distribution $\Pr(\cdot)$ is defined as:

$$\mathrm{PPL}(\Pr(y)) \coloneqq e^{\mathrm{H}(\Pr(y))} = e^{-\sum_y \Pr(y) \ln \Pr(y)} = \prod_y \Pr(y)^{-\Pr(y)}, \tag{4}$$

where $\mathrm{H}(\Pr(y))$ is the entropy of the distribution $\Pr(\cdot)$. Empowered by robust language models, such as GPT-2, PPL has been widely applied to assess the open-ended generation tasks. However, PPL has been criticized for exhibiting a bias to shorter or repetitive texts [246].

6.2 Resources

In this section, we will introduce some available open-source libraries and benchmarks.

6.2.1 Open-Source Libraries. There are a number of public text generation libraries that can be used to implement PLM-based text generation models. Transformers [249] and Fairseq [176] are all-featured libraries for reproducing and implementing Transformer-based PLMs for a wide range of text generation tasks. Besides, nanoT5 [173] and nanoGPT [108] implement the pre-training and fine-tuning of T5 and GPTs, and picoGPT [168] is a tiny and minimal implementation of GPT-2 in plain NumPy with only 40 lines of code for the entire forward pass. More specifically, yanmtt [37] and joeyNMT [114] are specially designed toolkits for **neural machine translation (NMT),** which supports the pre-training, fine-tuning, and decoding of various NMT models such as mBART and a wide range of tasks such as multi-source and document-level translation. Finally, some of libraries like FastSeq [255], DeepSpeed [198], and LightSeq [245] are useful to increase the inference speed of models. TextBox [128] supports 21 text generation models, including several prevalent PLMs, and diverse generation strategies (e.g., top-k, beam search) and evaluation metrics (e.g., BLEU, Distinct). One can easily choose different PLMs, optimization methods, and evaluation metrics by setting corresponding hyper-parameters with just a few lines of code.

6.2.2 Evaluation Benchmarks. In order to evaluate the comprehensive capacities of PLMs, several important evaluation benchmarks are created and released. Similar to NLU benchmarks like GLUE [241] and SuperGLUE [240], an increasing number of benchmarks for text generation have been proposed. Liu et al. [143] introduced **General Language Generation Evaluation (GLGE)**, a multi-task text generation benchmark containing eight English language generation tasks. For each task, they design three task difficulties, i.e., Easy, Medium, and Hard. GEM [66, 67] is a benchmark environment for text generation with a focus on its evaluation, both through human annotations and automatic metrics, covering 40 tasks and 51 languages and models. BIG-bench [220] consists of 204 tasks covering topics from math, biology, physics, and beyond. These tasks are assumed to be beyond the capabilities of current PLMs. To test text generation models on specific languages, IndicNLG Benchmark [117] collects approximately 8 M examples and focuses on five text generation tasks for 11 Indic languages. In another line, we also need meta-evaluation benchmarks to access the correlation between automatic metrics and human judgements. Representative benchmarks include WMT Metrics Shared Tasks [60] for translation and SummEval [51] for summarization.

7 APPLICATION

As discussed in Section 2, text generation can be instantiated into different kinds of applications. To summarize existing text generation applications, we present an overview of different tasks (as well as corresponding common datasets and metrics) in Table 1. In what follows, we will highlight three classic applications, *i.e.,* machine translation, text summarization, and dialogue system, and briefly discuss how to design a task-specific PLM to adapt to specific text generation tasks.

7.1 Machine Translation

With the advent of deep learning, **Neural Machine Translation (NMT)** has emerged as the dominant method in both academic research and commercial use [38]. Machine translation can be classified into two types: *unsupervised machine translation* and *supervised machine translation*, depending on whether parallel corpora are available for fine-tuning PLMs.

7.1.1 Unsupervised Machine Translation. **Unsupervised Machine Translation (UMT)** refers to the use of solely monolingual corpora without any parallel data for both pre-training and fine-tuning PLMs. UMT enables machine translation to no longer rely on large-scale annotated corpora, and also widely applied to low-resource language translation. When using PLMs for UMT, there are typically two steps involved [118]: (1) PLMs are pre-trained on monolingual corpora in a variety of languages; (2) Iterative back-translation is then leveraged to combine the source-to-target and target-to-source model with the denoising auto-encoding and back-translation objectives.

Pre-training on Monolingual Corpora. Recent PLM-based research has mainly focused on the first step of UMT. Specifically, XLM [35] and mBERT [43] were pre-trained on multiple monolingual data using MLM task, and then the PLM was used to initialize both the encoder and decoder for machine translation. mBART [149] followed the pre-training scheme of BART [123] on multiple languages. Researchers further continually pre-trained mBART on specific language families like IndoBART [19], IndicBART [40], and AfroBART [200]. However, these mBART models do not consider the relationship between languages. Thus, CMLM [201] and CSP [258] randomly masked tokens in monolingual sentences and predicted corresponding translation candidates. In this way, they are able to align the embeddings of different languages.

Fine-Tuning with Iterative Back-Translation. In the back-translation stage, Garcia *et al.* [64] proposed using multi-task learning. They investigated *multilingual UNMT*, which involved the use of a third language when translating one language into another. The extra language can provide auxiliary monolingual data or parallel data containing only one language in the source or target language. They aggregated back-translation loss and introduced a cross-translation term to incorporate the auxiliary corpus. Li *et al.* [138] also applied the cross-translation term and additionally included a knowledge distillation objective for the third (intermediate) language.

7.1.2 Supervised Machine Translation. **Supervised machine translation (SMT)** refers to fine-tuning PLMs based on parallel corpora. Here, we will discuss how to utilize existing self-supervised PLMs and how to design PLMs for parallel corpora.

Pre-Training on Parallel Corpora. Most of PLMs are pre-trained on monolingual corpora using self-supervised pre-training tasks such as MLM and DAE. Nevertheless, these pre-training objectives are different from the downstream translation task. Hence, mRASP [142] pre-trained the model on bilingual pairs with supervised Seq2Seq loss by randomly replacing the words in the source sentence with the words which have the same meaning in other languages. As a result, words with similar meaning across different languages are encouraged to share similar representations.

Directly Fine-Tuning Unsupervised PLMs. Almost all PLMs mentioned above using unsupervised (self-supervised) pre-training, such as XLM [35] and mBART [149], can be directly fine-tuned with bilingual pairs. Moreover, considering the excellent encoding capability of BERT, CTNMT [256] used asymptotic distillation and dynamic switching gate to integrate BERT embeddings. Tang *et al.* [231] fine-tuned mBART on multiple language pairs, called *multilingual fine-tuning*.

7.2 Text Summarization

Text summarization is the process of condensing text into a brief summary that retains key information from the source [48]. The mainstream approaches to text summarization based on PLMs are either extractive or abstractive. Extractive method selects a subset of sentences from the source text and concatenates them to form the summary [150, 273]. In contrast, abstractive method generates the summary automatically from the abstract representation of input texts [211, 269]. As abstractive method is more related to text generation, we only discuss abstractive summarization in this section.

7.2.1 Document Summarization. Document is a widely used literary form, such as news, opinions, reviews, and scientific papers. Some PLMs can be directly fine-tuned for document summarization (e.g., T5 [194], BART [123]). During pre-training, these models learn to predict the masked sentences in the input text based on remaining ones, which shares the similar idea of summarization.

Without directly generating summaries, several studies first extracted keywords, key sentences or relations as guidance and then combined these with PLMs for generation. CIT [207] employed RoBERTa [151] to extract important words and sentences from the input document. In addition, topic models are used to capture the global topic semantics of the document, which can be integrated into the summarization model [174]. Apart from external guidance, several tricks can be applied to document summarization. Refactor [148] first generated multiple summaries under different setups and then scored them and finally selected an optimal candidate summary.

7.2.2 Dialogue Summarization. Dialogues, such as chat and medical conversation, consist of multi-turn utterances by two or more individuals. Thus, it is critical to capture the semi-structured dialogue content and users' interactions in dialogue [56]. For dialogue summarization, it is straightforward to directly reuse document summarization models [270].

Meanwhile, several studies also explored some specific characteristics of dialogue for improving dialogue summarization. Chen and Yang [24] first extracted different topic views from conversations, and then utilized a multi-view decoder to combine these views for generating summaries. Furthermore, Chen and Yang [26] constructed discourse relation graphs and action graphs of conversations, focusing on the most salient utterances and concrete details of users' action. Considering the low information density, topic drifts, and frequent coreferences of dialogue [56], some researchers conducted auxiliary tasks to extract intrinsic information of dialogue.

7.3 Dialogue System

Dialogue system (a.k.a., *conversational agent*) aims to make machines communicate with human fluently. Technically, machines are required to generate a response conditioned on history contexts. According to downstream applications, dialogue systems are commonly categorized into open-domain and task-oriented dialogue systems. The former intends to converse with humans engaged on open topics such as daily life, sports, and entertainment [95], while the latter is focused on assisting users to complete specific tasks, such as hotel reservation and product purchase [279].

7.3.1 Open-Domain Dialogue System. Open-domain dialogue system is also known as chat-bots focusing on daily chat. For example, Microsoft XiaoIce is a well-known open-domain dialogue system to satisfy human needs for communication, affection, and social belonging [283].

Continuous Pre-Training with Dialogue Corpora. PLMs, such as GPT-2, are pre-trained on general text corpora, thus various studies continually pre-trained general-purpose PLMs to fit dialogue systems. Owing to the difficulty in obtaining large-scale dialogue corpora, informal text resources (such as forum posts and comments in Reddit, Twitter, and Weibo) are usually employed for continual pre-training. As two typical models, DialoGPT [274] and Meena [1] used English or Chinese dialogue corpora to continually pre-train casual LMs like GPT-2. Besides, Blender [204] and PLATO [9] utilized the Seq2Seq loss to generate the next utterance based on previous utterances. Moreover, PLATO [9] incorporated the **next utterance classification (NUC)** loss to judge whether the response is relevant to history dialogues to enhance the coherence of utterances.

Directly Fine-Tuning Existing PLMs. Besides pre-training on dialogue corpora, researchers also directly fine-tuned existing PLMs on dialogue tasks. TransferTransfo [250] adapted GPT to the dialogue task through multi-task learning. To capture the hierarchical structure of dialogue, hierarchical encoders have been proposed to model the dialogue input [77, 137]. Gu et al. [77] proposed a hierarchical framework, dialogueBERT, that uses sentence- and discourse-level Transformer encoders to encode each dialogue utterance and the sequence of utterance vectors, respectively. Furthermore, controllability is also important to consider in dialogue systems. Zeng and Nie [267] utilized condition-aware Transformer block to steer the response in a specific topic label.

7.3.2 Task-Oriented Dialogue System. Task-oriented (a.k.a., goal-oriented) dialogue system is a widely used text generation application in real life, such as helping users order tickets. Generally, task-oriented dialogue system was divided into four modules, *i.e.,* natural language understanding, dialogue state tracking, dialogue policy learning and natural language generation [279].

Most previous work only focused on the last generation module in task-oriented dialogue system by using generative PLMs (e.g., GPT). For example, SC-GPT [183] used the ground-truth results of previous three modules (e.g., dialogue state) and serialized them as input of the last generation module to generate response. Kale and Rastogi [105] further designed a manual schema to better convert previous results into a natural language. PRAL [76] utilized two separate GPT-2s to model the user and system, and adopted a third GPT-2 to perform knowledge distillation and incorporate commonsense knowledge into the final dialogue generation. Besides, more and more studies proposed to jointly learn these four modules based on a shared PLM. Budzianowski and Vulic [18] and Hosseini-Hosseini-Asl et al. [89] generated the dialogue state, system action and final response successively, based on the original dialogue history.

7.4 Question Generation

Question generation can be seen as a dual task of **question answering (QA),** i.e., generate coherent questions based on given passages and answers. Existing PLMs, such as UniLM [8, 45] and ProphetNet [190], can be employed for this task by taking as input the concatenation of the passage and answer. Moreover, researchers explored this task in different QA settings. For example, Huang et al. [96] proposed a two-stage model to solve multi-hop question generation, and Cao and Wang [21] attempted to generate open-ended questions which are answered by multiple sentences. Moreover, Majumder et al. [159] proposed a clarification question generation task to ask questions about the missing information in the passage in order to reduce the ambiguity.

7.5 Story Generation

Story (or narrative, news) generation requires the generation of a long-form open-ended text leveraging the given title or premise. It is challenging to produce a coherent and informative text based on limited input [63]. To enrich the content of generated text, some work aimed to incorporate external knowledge into PLMs. Guan et al. [79] and Mao et al. [161] utilized a commonsense knowledge base to fine-tune PLMs to generate reasonable stories. MEGATRON-CNTRL [253] used extracted keywords to retrieve knowledge sentences and then selected top-ranked sentences for story generation. Besides, to generate coherent long-form text, Rashkin et al. [197] extracted keywords from input as outline to organize the output structure, and Guan et al. [79] leveraged the contrastive learning to judge whether two sentences are consecutive in original text.

7.6 Data-to-Text Generation

The above tasks take unstructured text as input, while the data-to-text generation task generates descriptive text about structured input data, such as table, **knowledge graph (KG)** and **abstract meaning representation (AMR)**. First, a naive and straightforward approach is to directly linearize the structured table [33, 72] and KG [85, 202] into textual form as the input of PLMs. Considering the graph structure of KG and AMR, Li et al. [130] and Ribeiro et al. [203] employed graph neural network to learn a better representation for each node. Moreover, to cope with the structural information, a typical approach is to incorporate auxiliary training objectives such as predicting the value of the table [72] and the relation of knowledge graph [130].

7.7 More Kinds of Generation Tasks

Besides the aforementioned tasks, there are also other text generation applications. ColdGANs [210] explored the unconditional language generation. KG-BART [152] investigates the commonsense generation, i.e., generating a natural language consisting of provided commonsense concept (word), which can be considered as the hard-constrained conditional generation [63]. Explanation generation is designed to interpret AI algorithms, elucidating how models arrive at specific decisions to enhance users' trust and augment the usability of the algorithms [221]. Moreover, text style transfer aims to convert a text into another style while preserving the basic semantics of input [63], such as sentiment transfer and writing style transfer [115]. In addition, some researchers devoted to literary creation, such as poem [134] and lyric [254].

8 CONCLUSION AND FUTURE DIRECTIONS

In this survey, we presented an overview of current representative research efforts on PLMs-based text generation, and expect it can facilitate future research. We began with introducing three key aspects when applying PLMs to text generation, based on which the main content of our survey is divided into three sections from the view of input representation learning, model architecture design, and parameter optimization. Besides, we discussed several non-trivial challenges related to the above three aspects. Finally, we reviewed various evaluation metrics, open-source libraries, and common applications to help practitioners evaluate, choose and employ PLMs for text generation.

Despite the great progress made in recent years, we are faced with several open problems and several future directions are promising to deal with them.

Controllable Generation. Controllable text generation with PLMs is an interesting direction but still at a very early stage. Controlling some attributes of the generated text has many practical use cases, such as generating positive responses to patients suffering from depression in dialogue systems. However, PLMs are usually pre-trained in universal corpora, which is difficult to control the multi-grained attributes of the generated text (e.g., sentiment, topic, and coherence). Keskar et al. [110] has explored text generation with control codes that govern style, content and task-specific behavior. However, these control codes are preset and coarse-grained. Future work can explore multi-grained control and develop PLMs that are sufficiently steerable.

Optimization Exploration. Fine-tuning is the predominant optimization way to distill the linguistic knowledge stored in PLMs to downstream generation tasks. Now, prompt-based learning has become a performant and lightweight optimization method [145]. Future work can explore a broader range of optimization approaches that can combine the advantages of current methods.

Language-Agnostic PLMs. Nowadays, almost all the PLMs for text generation are mainly for the English language. These PLMs will encounter challenges when dealing with non-English generation tasks. Therefore, language-agnostic PLMs are worthy to be investigated. This requires us to capture universal linguistic and semantic features across different languages. An interesting direction to explore is how to reuse existing English-based PLMs for text generation in non-English languages.

Generation with LLMs. Recently, LLMs [280] have shown excellent capabilities in various text generation tasks. After fine-tuning LLMs with instructions, it becomes flexible to directly utilize LLMs with natural language instructions without any fine-tuning. In the future, researchers can further investigate how to design instructions or demonstrations to satisfy general or specialized generation scenarios.

Ethical Concern. Currently, PLMs are pre-trained on large-scale corpora crawled from web without fine-grained filtering, potentially causing ethical issues such as generating private content about users. Therefore, researchers should try their best to prevent misusing PLMs. Besides, the text generated by PLMs might be prejudiced, which is in line with the bias in training data along the dimensions of gender, race, and religion [16]. As a result, we should intervene PLMs for preventing such biases. The research on the general approach is extensive but still preliminary for PLMs.

In conclusion, text generation based on PLMs has greatly contributed to the advance of the state of the art in this field. However, the current state of the art in text generation is still far from what we expect. Extensive research efforts are needed to better adapt PLMs to text generation tasks.

ACKNOWLEDGEMENTS

We thank all anonymous reviewers for valuable comments.

REFERENCES

[1] Daniel Adiwardana, Minh-Thang Luong, David R. So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V. Le. 2020. Towards a human-like open-domain chatbot. *CoRR* abs/2001.09977 (2020).

[2] Abhaya Agarwal and Alon Lavie. 2008. Meteor, M-BLEU and M-TER: Evaluation metrics for high-correlation with human rankings of machine translation output. In *WMT@ACL*.

[3] Alpaca-LoRA. 2023. Instruct-Tune LLaMA on Consumer Hardware. https://github.com/tloen/alpaca-lora. (2023).

[4] Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Benjamin Mann, Nova DasSarma, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Jackson Kernion, Kamal Ndousse, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Chris Olah, and Jared Kaplan. 2021. A general language assistant as a laboratory for alignment. *CoRR* abs/2112.00861 (2021).

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

[6] Yu Bai, Yang Gao, and Heyan Huang. 2021. Cross-lingual abstractive summarization with limited parallel resources. In *ACL*.

[7] Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *IEEvaluation@ACL*.

[8] Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, and Hsiao-Wuen Hon. 2020. UniLMv2: Pseudo-masked language models for unified language model pre-training. In *ICML (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, 642–652.

[9] Siqi Bao, Huang He, Fan Wang, Hua Wu, and Haifeng Wang. 2020. PLATO: Pre-trained dialogue generation model with discrete latent variable. In *ACL*.

[10] Anja Belz and Ehud Reiter. 2006. Comparing automatic and human evaluation of NLG systems. In *EACL*.

[11] Bin Bi, Chenliang Li, Chen Wu, Ming Yan, Wei Wang, Songfang Huang, Fei Huang, and Luo Si. 2020. PALM: Pre-training an autoencoding & autoregressive language model for context-conditioned generation. In *EMNLP*.

[12] Ondrej Bojar, Yvette Graham, Amir Kamran, and Milos Stanojevic. 2016. Results of the WMT16 metrics shared task. In *1st Conference on Machine Translation (WMT 2016, colocated with ACL 2016)*.

[13] Peter F. Brown, John Cocke, Stephen Della Pietra, Vincent J. Della Pietra, Frederick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Comput. Linguistics* (1990).

[14] Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, Jennifer C. Lai, and Robert L. Mercer. 1992. An estimate of an upper bound for the entropy of english. *Comput. Linguistics* (1992).

[15] Ralf Brown and Robert Frederking. 1995. Applying statistical english language modeling to symbolic machine translation. In *TMI*.

[16] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NeurIPS*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.).

[17] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott M. Lundberg, Harsha Nori, Hamid Palangi, Marco Túlio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with GPT-4. *CoRR* abs/2303.12712 (2023).

[18] Pawel Budzianowski and Ivan Vulic. 2019. "Hello, it's GPT-2 - How can I help you?" Towards the use of pretrained language models for task-oriented dialogue systems. In *NGT@EMNLP-IJCNLP*.

[19] Samuel Cahyawijaya, Genta Indra Winata, Bryan Wilie, Karissa Vincentio, Xiaohong Li, Adhiguna Kuncoro, Sebastian Ruder, Zhi Yuan Lim, Syafri Bahar, Masayu Leylia Khodra, Ayu Purwarianti, and Pascale Fung. 2021. IndoNLG: Benchmark and resources for evaluating indonesian natural language generation. In *EMNLP*.

[20] Chris Callison-Burch, Cameron S. Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. 2008. Further meta-evaluation of machine translation. In *t3rd Workshop on Statistical Machine Translation, WMT@ACL 2008*.

[21] Shuyang Cao and Lu Wang. 2021. Controllable open-ended question generation with a new question type ontology. In *ACL/IJCNLP*.

[22] Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of text generation: A survey. *arXiv* (2020).

[23] Jun Chen, Han Guo, Kai Yi, Boyang Li, and Mohamed Elhoseiny. 2021. VisualGPT: Data-efficient adaptation of pretrained language models for image captioning. *arXiv* (2021).

[24] Jiaao Chen and Diyi Yang. 2020. Multi-view sequence-to-sequence models with conversational structure for abstractive dialogue summarization. In *EMNLP*.

[25] Jiaao Chen and Diyi Yang. 2021. Simple conversational data augmentation for semi-supervised abstractive dialogue summarization. In *EMNLP*.
[26] Jiaao Chen and Diyi Yang. 2021. Structure-aware abstractive conversation summarization via discourse and action graphs. In *NAACL-HLT*.
[27] Liqun Chen, Shuyang Dai, Chenyang Tao, Haichao Zhang, Zhe Gan, Dinghan Shen, Yizhe Zhang, Guoyin Wang, Ruiyi Zhang, and Lawrence Carin. 2018. Adversarial text generation via feature-mover's distance. In *NeurIPS*.
[28] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. Extending context window of large language models via positional interpolation. *CoRR* abs/2306.15595 (2023).
[29] Wenhu Chen, Yu Su, Xifeng Yan, and William Yang Wang. 2020. KGPT: Knowledge-grounded pre-training for data-to-text generation. In *EMNLP*.
[30] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C. Lawrence Zitnick. 2015. Microsoft COCO captions: Data collection and evaluation server. *arXiv* (2015).
[31] Yen-Chun Chen, Zhe Gan, Yu Cheng, Jingzhou Liu, and Jingjing Liu. 2020. Distilling knowledge learned in BERT for text generation. In *ACL*.
[32] Yi-Syuan Chen and Hong-Han Shuai. 2021. Meta-transfer learning for low-resource abstractive summarization. In *AAAI*.
[33] Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. 2020. Few-shot NLG with pre-trained language model. In *ACL*.
[34] David Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations?. In *ACL*.
[35] Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *NeurIPS*.
[36] Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Y. Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loïc Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation. *CoRR* abs/2207.04672 (2022).
[37] Raj Dabre. YANMTT: Yet Another Neural Machine Translation Toolkit. (n.d.). https://github.com/prajdabre/yanmtt
[38] Raj Dabre, Chenhui Chu, and Anoop Kunchukuttan. 2020. A survey of multilingual neural machine translation. *CSUR* (2020).
[39] Raj Dabre, Atsushi Fujita, and Chenhui Chu. 2019. Exploiting multilingualism through multistage fine-tuning for low-resource neural machine translation. In *EMNLP*.
[40] Raj Dabre, Himani Shrotriya, Anoop Kunchukuttan, Ratish Puduppully, Mitesh M. Khapra, and Pratyush Kumar. 2022. IndicBART: A pre-trained model for indic natural language generation. In *Findings of ACL*.
[41] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *ICLR*.
[42] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. LLM.int8(): 8-Bit matrix multiplication for transformers at scale. *CoRR* abs/2208.07339 (2022).
[43] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.
[44] Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq R. Joty, Luo Si, and Chunyan Miao. 2020. DAGA: Data augmentation with a generation approach for low-resource tagging tasks. *arXiv* (2020).
[45] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *NeurIPS*.
[46] Yue Dong, Shuohang Wang, Zhe Gan, Yu Cheng, Jackie Chi Kit Cheung, and Jingjing Liu. 2020. Multi-fact correction in abstractive text summarization. In *EMNLP*.
[47] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2021. All NLP tasks are generation tasks: A general pretraining framework. *arXiv preprint arXiv:2103.10360* (2021).
[48] Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed. 2021. Automatic text summarization: A comprehensive survey. *Expert Syst. Appl.* (2021).
[49] Gamaleldin F. Elsayed, Dilip Krishnan, Hossein Mobahi, Kevin Regan, and Samy Bengio. 2018. Large margin deep networks for classification. In *NeurIPS*.
[50] Alexander R. Fabbri, Simeng Han, Haoyuan Li, Haoran Li, Marjan Ghazvininejad, Shafiq R. Joty, Dragomir R. Radev, and Yashar Mehdad. 2021. Improving zero and few-shot abstractive summarization with intermediate fine-tuning and data augmentation. In *NAACL-HLT*.

[51] Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *TACL* (2021).
[52] Angela Fan, Edouard Grave, and Armand Joulin. 2020. Reducing transformer depth on demand with structured dropout. In *ICLR*.
[53] Angela Fan, Mike Lewis, and Yann N. Dauphin. 2018. Hierarchical neural story generation. In *ACL*.
[54] Zhiyun Fan, Shiyu Zhou, and Bo Xu. 2019. Unsupervised pre-training for sequence to sequence speech recognition. *arXiv preprint arXiv:1910.12418* (2019).
[55] William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.* 23 (2022).
[56] Xiachong Feng, Xiaocheng Feng, and Bing Qin. 2021. A survey on dialogue summarization: Recent advances and new frontiers. *arXiv preprint arXiv:2107.03175* (2021).
[57] Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020. CodeBERT: A pre-trained model for programming and natural languages. In *EMNLP Findings (Findings of ACL, Vol. EMNLP 2020)*, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, 1536–1547.
[58] Elias Frantar and Dan Alistarh. 2022. Optimal brain compression: A framework for accurate post-training quantization and pruning. In *NeurIPS*.
[59] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. GPTQ: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323* (2022).
[60] Markus Freitag, Nitika Mathur, Chi-kiu Lo, Eleftherios Avramidis, Ricardo Rei, Brian Thompson, Tom Kocmi, Frédéric Blain, Daniel Deutsch, Craig Stewart, Chrysoula Zerva, Sheila Castilho, Alon Lavie, and George F. Foster. 2023. Results of WMT23 metrics shared task: Metrics might be guilty but references are not innocent. In *Proceedings of the Eighth Conference on Machine Translation, WMT 2023, Singapore, December 6-7, 2023*, Philipp Koehn, Barry Haddon, Tom Kocmi, and Christof Monz (Eds.). Association for Computational Linguistics, 578–628.
[61] Prakhar Ganesh, Yao Chen, Xin Lou, Mohammad Ali Khan, Yin Yang, Hassan Sajjad, Preslav Nakov, Deming Chen, and Marianne Winslett. 2020. Compressing large-scale transformer-based models: A case study on BERT. *Trans. Assoc. Comput. Linguistics* 9 (2021), 1061–1080.
[62] Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *ACL*.
[63] Cristina Garbacea and Qiaozhu Mei. 2020. Neural language generation: Formulation, methods, and evaluation. *arXiv preprint arXiv:2007.15780* (2020).
[64] Xavier Garcia, Pierre Foret, Thibault Sellam, and Ankur P. Parikh. 2020. A multilingual view of unsupervised machine translation. In *EMNLP Findings*.
[65] Ismael Garrido-Muñoz, Arturo Montejo-Ráez, Fernando Martínez-Santiago, and L. Alfonso Ureña-López. 2021. A survey on bias in deep NLP. *Applied Sciences* (2021).
[66] Sebastian Gehrmann, Tosin P. Adewumi, Karmanya Aggarwal, Pawan Sasanka Ammanamanchi, Aremu Anuoluwapo, Antoine Bosselut, Khyathi Raghavi Chandu, Miruna-Adriana Clinciu, Dipanjan Das, Kaustubh D. Dhole, Wanyu Du, Esin Durmus, Ondrej Dusek, Chris Emezue, Varun Gangal, Cristina Garbacea, Tatsunori Hashimoto, Yufang Hou, Yacine Jernite, Harsh Jhamtani, Yangfeng Ji, Shailza Jolly, Dhruv Kumar, Faisal Ladhak, Aman Madaan, Mounica Maddela, Khyati Mahajan, Saad Mahamood, Bodhisattwa Prasad Majumder, Pedro Henrique Martins, Angelina McMillan-Major, Simon Mille, Emiel van Miltenburg, Moin Nadeem, Shashi Narayan, Vitaly Nikolaev, Rubungo Andre Niyongabo, Salomey Osei, Ankur P. Parikh, Laura Perez-Beltrachini, Niranjan Ramesh Rao, Vikas Raunak, Juan Diego Rodriguez, Sashank Santhanam, João Sedoc, Thibault Sellam, Samira Shaikh, Anastasia Shimorina, Marco Antonio Sobrevilla Cabezudo, Hendrik Strobelt, Nishant Subramani, Wei Xu, Diyi Yang, Akhila Yerukola, and Jiawei Zhou. 2021. The gem benchmark: Natural language generation, its evaluation and metrics. *CoRR* abs/2102.01672 (2021).
[67] Sebastian Gehrmann, Abhik Bhattacharjee, Abinaya Mahendiran, Alex Wang, Alexandros Papangelis, Aman Madaan, Angelina McMillan-Major, Anna Shvets, Ashish Upadhyay, and Bernd Bohnet. 2022. Gemv2: Multilingual NLG benchmarking in a single line of code. In *Proceedings of the The 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022 - System Demonstrations, Abu Dhabi, UAE, December 7-11, 2022*, Wanxiang Che and Ekaterina Shutova (Eds.). Association for Computational Linguistics, 266–281.
[68] Mozhdeh Gheini, Xiang Ren, and Jonathan May. 2021. On the strengths of cross-attention in pretrained transformers for machine translation. *arXiv preprint arXiv:2104.08771* (2021).
[69] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. 2021. A survey of quantization methods for efficient neural network inference. *arXiv* (2021).
[70] Amelia Glaese, Nat McAleese, Maja Trebacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin J. Chadwick, Phoebe Thacker, Lucy Campbell-Gillingham, Jonathan Uesato, Po-Sen Huang, Ramona

Comanescu, Fan Yang, Abigail See, Sumanth Dathathri, Rory Greig, Charlie Chen, Doug Fritz, Jaume Sanchez Elias, Richard Green, Sona Mokrá, Nicholas Fernando, Boxi Wu, Rachel Foley, Susannah Young, Iason Gabriel, William Isaac, John Mellor, Demis Hassabis, Koray Kavukcuoglu, Lisa Anne Hendricks, and Geoffrey Irving. 2022. Improving alignment of dialogue agents via targeted human judgements. *CoRR* abs/2209.14375 (2022).

[71] Sergey Golovanov, Rauf Kurbanov, Sergey I. Nikolenko, Kyryl Truskovskyi, Alexander Tselousov, and Thomas Wolf. 2019. Large-scale transfer learning for natural language generation. In *ACL*.

[72] Heng Gong, Yawei Sun, Xiaocheng Feng, Bing Qin, Wei Bi, Xiaojiang Liu, and Ting Liu. 2020. TableGPT: Few-shot table-to-text generation with table structure reconstruction and content matching. In *COLING*.

[73] Travis R. Goodwin, Max E. Savery, and Dina Demner-Fushman. 2020. Towards zero shot conditional summarization with adaptive multi-task fine-tuning. In *EMNLP Findings*.

[74] Mitchell A. Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing BERT: Studying the effects of weight pruning on transfer learning. In *RepL4NLP@ACL*.

[75] Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *ICLR (Poster)*.

[76] Jing Gu, Qingyang Wu, Chongruo Wu, Weiyan Shi, and Zhou Yu. 2021. PRAL: A tailored pre-training model for task-oriented dialog generation. In *ACL/IJCNLP Short*.

[77] Xiaodong Gu, Kang Min Yoo, and Jung-Woo Ha. 2021. DialogBERT: Discourse-aware response generation via learning to recover and rank utterances. In *AAAI*.

[78] Xiaodong Gu, Kang Min Yoo, and Sang-Woo Lee. 2021. Response generation with context-aware prompt learning. *arXiv preprint arXiv:2111.02643* (2021).

[79] Jian Guan, Fei Huang, Minlie Huang, Zhihao Zhao, and Xiaoyan Zhu. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *TACL* (2020).

[80] Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *ICLR*.

[81] Fu-Ming Guo, Sijia Liu, Finlay S. Mungall, Xue Lin, and Yanzhi Wang. 2019. Reweighted proximal pruning for large-scale language representation. *arXiv preprint arXiv:1909.12486* (2019).

[82] Mandy Guo, Joshua Ainslie, David C. Uthus, Santiago Ontañón, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. LongT5: Efficient text-to-text transformer for long sequences. In *NAACL-HLT (Findings)*.

[83] Yue Guo, Yi Yang, and Ahmed Abbasi. 2022. Auto-Debias: Debiasing masked language models with automated biased prompts. In *ACL*.

[84] Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, Wentao Han, Minlie Huang, Qin Jin, Yanyan Lan, Yang Liu, Zhiyuan Liu, Zhiwu Lu, Xipeng Qiu, Ruihua Song, Jie Tang, Ji-Rong Wen, Jinhui Yuan,Wayne Xin Zhao, and Jun Zhu. 2021. Pre-trained models: Past, present and future. *AI Open* 2 (2021), 225–250.

[85] Hamza Harkous, Isabel Groves, and Amir Saffari. 2020. Have your text and use it too! End-to-end neural data-to-text generation with semantic fidelity. In *COLING*.

[86] Sadid A. Hasan and Oladimeji Farri. 2019. Clinical natural language processing with deep learning. In *Data Science for Healthcare - Methodologies and Applications*, Sergio Consoli, Diego Reforgiato Recupero, and Milan Petkovic (Eds.). Springer, 147–171.

[87] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *ICLR*.

[88] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *ICLR*.

[89] Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. In *NeurIPS*.

[90] Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. 2020. DynaBERT: Dynamic BERT with adaptive width and depth. In *NeurIPS*.

[91] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *ICML*.

[92] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *ICLR*.

[93] Xinyu Hua, Ashwin Sreevatsa, and Lu Wang. 2021. DYPLOC: Dynamic planning of content using mixed language models for text generation. In *ACL/IJCNLP*.

[94] Haoyang Huang, Tianyi Tang, Dongdong Zhang, Wayne Xin Zhao, Ting Song, et al. 2023. Not all languages are created equal in LLMs: Improving multilingual capability by cross-lingual-thought prompting. *arXiv* (2023).

[95] Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. 2020. Challenges in building intelligent open-domain dialog systems. *TOIS* (2020).

[96] Xinting Huang, Jianzhong Qi, Yu Sun, and Rui Zhang. 2021. Latent reasoning for low-resource question generation. In *ACL/IJCNLP Findings*.

[97] Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Xu Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V. Le, Yonghui Wu, and Zhifeng Chen. 2019. GPipe: Efficient training of giant neural networks using pipeline parallelism. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019 NeurIPS 2019, December 8-14, 2019*, Vancouver, BC, Canada. 103–112.

[98] Touseef Iqbal and Shaima Qureshi. 2022. The survey: Text generation models in deep learning. *J. King Saud Univ. Comput. Inf. Sci.* 34, 6 Part A (2022), 2515–2528.

[99] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive mixtures of local experts. *Neural Comput.* (1991).

[100] Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know. *TACL* (2020).

[101] Zihang Jiang, Weihao Yu, Daquan Zhou, Yunpeng Chen, Jiashi Feng, and Shuicheng Yan. 2020. ConvBERT: Improving BERT with span-based dynamic convolution. In *NeurIPS*.

[102] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *EMNLP Findings*.

[103] Di Jin, Zhijing Jin, Joey Tianyi Zhou, Lisa Orii, and Peter Szolovits. 2020. Hooks in the headline: Learning to generate headlines with controlled styles. In *ACL*.

[104] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

[105] Mihir Kale and Abhinav Rastogi. 2020. Template guided text generation for task-oriented dialogue. In *EMNLP*.

[106] Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. 2021. AMMUS : A survey of transformer-based pretrained models in natural language processing. *arXiv preprint arXiv:2108.05542* (2021).

[107] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv* (2020).

[108] Andrej karpathy. 2023. *nanoGPT: The Simplest, Fastest Repository for Training/Finetuning Medium-Sized GPTs*.

[109] Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2023. The impact of positional encoding on length generalization in transformers. *CoRR* abs/2305.19466 (2023).

[110] Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858* (2019).

[111] Muhammad Khalifa, Hady Elsahar, and Marc Dymetman. 2021. A distributional approach to controlled text generation. In *ICLR*.

[112] Mert Kilickaya, Aykut Erdem, Nazli Ikizler-Cinbis, and Erkut Erdem. 2017. Re-evaluating automatic metrics for image captioning. In *EACL*.

[113] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2016. Overcoming catastrophic forgetting in neural networks. *CoRR* abs/1612.00796 (2016).

[114] Julia Kreutzer, Jasmijn Bastings, and Stefan Riezler. 2019. Joey NMT: A minimalist NMT toolkit for novices. In *EMNLP-IJCNLP: System Demonstrations*.

[115] Kalpesh Krishna, John Wieting, and Mohit Iyyer. 2020. Reformulating unsupervised style transfer as paraphrase generation. In *EMNLP*.

[116] Wojciech Kryscinski, Romain Paulus, Caiming Xiong, and Richard Socher. 2018. Improving abstraction in text summarization. In *EMNLP*.

[117] Aman Kumar, Himani Shrotriya, Prachi Sahu, Amogh Mishra, Raj Dabre, Ratish Puduppully, Anoop Kunchukuttan, Mitesh M. Khapra, and Pratyush Kumar. 2022. Indicnlg benchmark: Multilingual datasets for diverse nlg tasks in indic languages. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022.* Association for Computational Linguistics, 5363–5394.

[118] Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Phrase-based & neural unsupervised machine translation. In *EMNLP*.

[119] Hang Le, Juan Miguel Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. 2021. Lightweight adapter tuning for multilingual speech translation. In *ACL/IJCNLP Short*.

[120] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. 2015. Deep learning. *Nat.* (2015).

[121] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. GShard: Scaling giant models with conditional computation and automatic sharding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021.* OpenReview.net.

[122] Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*.
[123] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*.
[124] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *NAACL-HLT*.
[125] Jiwei Li and Eduard H. Hovy. 2014. A model of coherence based on distributed sentence representation. In *EMNLP*.
[126] Junyi Li, Siqing Li, Wayne Xin Zhao, Gaole He, Zhicheng Wei, Nicholas Jing Yuan, and Ji-Rong Wen. 2020. Knowledge-enhanced personalized review generation with capsule graph neural network. In *CIKM*.
[127] Jianquan Li, Xiaokang Liu, Honghong Zhao, Ruifeng Xu, Min Yang, and Yaohong Jin. 2020. BERT-EMD: Many-to-many layer mapping for BERT compression with earth mover's distance. In *EMNLP*.
[128] Junyi Li, Tianyi Tang, Gaole He, Jinhao Jiang, Xiaoxuan Hu, Puzhao Xie, Zhipeng Chen, Zhuohao Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2021. TextBox: A unified, modularized, and extensible framework for text generation. In *ACL Demonstration*.
[129] Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2022. ELMER: A non-autoregressive pre-trained language model for efficient and effective text generation. In *EMNLP*.
[130] Junyi Li, Tianyi Tang, Wayne Xin Zhao, Zhicheng Wei, Nicholas Jing Yuan, and Ji-Rong Wen. 2021. Few-shot knowledge graph-to-text generation with pretrained language models. In *ACL/IJCNLP Findings*.
[131] Junyi Li, Tianyi Tang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Pretrained language models for text generation: A survey. *arXiv preprint arXiv:2105.10311* (2021).
[132] Junyi Li, Wayne Xin Zhao, Zhicheng Wei, Nicholas Jing Yuan, and Ji-Rong Wen. 2021. Knowledge-based review generation by coherence enhanced text planning. In *SIGIR*.
[133] Junyi Li, Wayne Xin Zhao, Ji-Rong Wen, and Yang Song. 2019. Generating long and informative reviews with aspect-aware coarse-to-fine decoding. In *ACL*.
[134] Piji Li, Haisong Zhang, Xiaojiang Liu, and Shuming Shi. 2020. Rigid formats controlled text generation. In *ACL*.
[135] Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023. Contrastive decoding: Open-ended text generation as optimization. In *ACL (1)*. Association for Computational Linguistics, 12286–12312.
[136] Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing continuous prompts for generation. In *ACL*.
[137] Zekang Li, Jinchao Zhang, Zhengcong Fei, Yang Feng, and Jie Zhou. 2021. Conversations are not flat: Modeling the dynamic information flow across dialogue utterances. In *ACL/IJCNLP*.
[138] Zuchao Li, Hai Zhao, Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2020. Reference language based unsupervised neural machine translation. In *EMNLP Findings*.
[139] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *CoRR* abs/2305.20050 (2023).
[140] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*.
[141] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common objects in context. In *ECCV*.
[142] Zehui Lin, Xiao Pan, Mingxuan Wang, Xipeng Qiu, Jiangtao Feng, Hao Zhou, and Lei Li. 2020. Pre-training multilingual neural machine translation by leveraging alignment information. In *EMNLP*.
[143] Dayiheng Liu, Yu Yan, Yeyun Gong, Weizhen Qi, Hang Zhang, Jian Jiao, Weizhu Chen, Jie Fu, Linjun Shou, Ming Gong, Pengcheng Wang, Jiusheng Chen, Daxin Jiang, Jiancheng Lv, Ruofei Zhang, Winnie Wu, Ming Zhou, and Nan Duan. 2021. GLGE: A new general language generation evaluation benchmark. In *ACL/IJCNLP Findings*.
[144] Junpeng Liu, Yanyan Zou, Hainan Zhang, Hongshen Chen, Zhuoye Ding, Caixia Yuan, and Xiaojie Wang. 2021. Topic-aware contrastive learning for abstractive dialogue summarization. In *EMNLP Findings*.
[145] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv* (2021).
[146] Siyang Liu, Sahand Sabour, Yinhe Zheng, Pei Ke, Xiaoyan Zhu, and Minlie Huang. 2022. Rethinking and refining the distinct metric. In *ACL*.
[147] Shilei Liu, Xiaofeng Zhao, Bochao Li, Feiliang Ren, Longhui Zhang, and Shujuan Yin. 2021. A three-stage learning framework for low-resource knowledge-grounded dialogue generation. In *EMNLP*.
[148] Yixin Liu, Zi-Yi Dou, and Pengfei Liu. 2021. RefSum: Refactoring neural summarization. In *NAACL-HLT*.
[149] Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *TACL* (2020).
[150] Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *EMNLP/IJCNLP*.

[151] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv* (2019).

[152] Ye Liu, Yao Wan, Lifang He, Hao Peng, and Philip S. Yu. 2021. KG-BART: Knowledge graph-augmented BART for generative commonsense reasoning. In *AAAI*.

[153] Yuchen Liu, Jiajun Zhang, Hao Xiong, Long Zhou, Zhongjun He, Hua Wu, Haifeng Wang, and Chengqing Zong. 2020. Synchronous speech recognition and speech-to-text translation with interactive decoding. In *AAAI*.

[154] Zihan Liu, Genta Indra Winata, and Pascale Fung. 2021. Continual mixed-language pre-training for extremely low-resource neural machine translation. In *ACL/IJCNLP Findings*.

[155] Antoine Louis. 2020. *NetBERT: A Pre-trained Language Representation Model for Computer Networking*. Ph.D. Dissertation.

[156] Fuli Luo, Wei Wang, Jiahao Liu, Yijia Liu, Bin Bi, Songfang Huang, Fei Huang, and Luo Si. 2021. VECO: Variable and flexible cross-lingual pre-training for language understanding and generation. In *ACL/IJCNLP*.

[157] Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Xilin Chen, and Ming Zhou. 2020. UNIVL: A unified video and language pre-training model for multimodal understanding and generation. *CoRR* abs/2002.06353 (2020).

[158] Ahmed Magooda and Diane J. Litman. 2021. Mitigating data scarceness through data synthesis, augmentation and curriculum for abstractive summarization. In *EMNLP Findings*.

[159] Bodhisattwa Prasad Majumder, Sudha Rao, Michel Galley, and Julian J. McAuley. 2021. Ask what's missing and what's useful: Improving clarification question generation using global knowledge. In *NAACL-HLT*.

[160] Potsawee Manakul and Mark J. F. Gales. 2021. Long-span summarization via local attention and content selection. In *ACL/IJCNLP*.

[161] Huanru Henry Mao, Bodhisattwa Prasad Majumder, Julian J. McAuley, and Garrison W. Cottrell. 2019. Improving neural story generation by targeted common sense grounding. In *EMNLP/IJCNLP*.

[162] Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *ACL*.

[163] Kaushal Kumar Maurya, Maunendra Sankar Desarkar, Yoshinobu Kano, and Kumari Deepshikha. 2021. ZmBART: An unsupervised cross-lingual transfer framework for language generation. In *ACL/IJCNLP Findings*.

[164] Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2023. Locally typical sampling. *TACL* (2023).

[165] Jacob Menick, Maja Trebacz, Vladimir Mikulik, John Aslanides, H. Francis Song, Martin J. Chadwick, Mia Glaese, Susannah Young, Lucy Campbell-Gillingham, Geoffrey Irving, and Nat McAleese. 2022. Teaching language models to support answers with verified quotes. *CoRR* abs/2203.11147 (2022).

[166] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory F. Diamos, Erich Elsen, David García, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2017. Mixed precision training. *arXiv* (2017).

[167] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.

[168] Jay Mody. picoGPT: An unnecessarily tiny implementation of GPT-2 in NumPy. (n.d.).

[169] Kenton Murray and David Chiang. 2018. Correcting length bias in neural machine translation. *arXiv* (2018).

[170] Arsha Nagrani, Shan Yang, Anurag Arnab, Aren Jansen, Cordelia Schmid, and Chen Sun. 2021. Attention bottlenecks for multimodal fusion. In *NeurIPS*.

[171] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. WebGPT: Browser-assisted question-answering with human feedback. *arXiv* (2021).

[172] Feng Nan, Cícero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Kathleen R. McKeown, Ramesh Nallapati, Dejiao Zhang, Zhiguo Wang, Andrew O. Arnold, and Bing Xiang. 2021. Improving factual consistency of abstractive summarization via question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume1: Long Papers), Virtual Event, August 1-6, 2021*, Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (Eds.). Association for Computational Linguistics, 6881–6894.

[173] Piotr Nawrot. 2023. nanoT5: Fast & simple repository for pre-training and fine-tuning T5-style models. (2023).

[174] Thong Nguyen, Anh Tuan Luu, Truc Lu, and Tho Quan. 2021. Enriching and controlling global semantics for text summarization. In *EMNLP*.

[175] OpenAI. 2023. GPT-4 technical report. *OpenAI* (2023).

[176] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. Fairseq: A fast, extensible toolkit for sequence modeling. In *NAACL-HLT Demonstrations*.

[177] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda

Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022.*

[178] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *ACL*.

[179] Damian Pascual, Beni Egressy, Clara Meister, Ryan Cotterell, and Roger Wattenhofer. 2021. A plug-and-play method for controlled text generation. In *EMNLP Findings*.

[180] Ramakanth Pasunuru, Asli Celikyilmaz, Michel Galley, Chenyan Xiong, Yizhe Zhang, Mohit Bansal, and Jianfeng Gao. 2021. Data augmentation for abstractive query-focused multi-document summarization. In *AAAI*.

[181] Ramakanth Pasunuru, Mengwen Liu, Mohit Bansal, Sujith Ravi, and Markus Dreyer. 2021. Efficiently summarizing text and graph encodings of multi-document clusters. In *NAACL-HLT*.

[182] Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *ICLR (Poster)*.

[183] Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020. Few-shot natural language generation for task-oriented dialog. In *EMNLP Findings*.

[184] Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088* (2018).

[185] Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT?. In *ACL*.

[186] Maja Popovic. 2017. chrF++: Words helping character *n*-grams. In *WMT*.

[187] Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *NAACL-HLT*.

[188] Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. In *ICLR*.

[189] Weizhen Qi, Yeyun Gong, Jian Jiao, Yu Yan, Weizhu Chen, Dayiheng Liu, Kewen Tang, Houqiang Li, Jiusheng Chen, Ruofei Zhang, Ming Zhou, and Nan Duan. 2021. BANG: Bridging autoregressive and non-autoregressive generation with large scale pretraining. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021,18-24 July 2021, Virtual Event (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 8630–8639.

[190] Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. ProphetNet: Predicting future n-gram for sequence-to-sequence pre-training. In *EMNLP Findings*.

[191] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *arXiv preprint arXiv:2003.08271* (2020).

[192] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-Training. (2018).

[193] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog* (2019).

[194] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR* (2020).

[195] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. ZeRO: Memory optimizations toward training trillion parameter models. In *SC*. 20.

[196] Surangika Ranathunga, En-Shiun Annie Lee, Marjana Prifti Skenduli, Ravi Shekhar, Mehreen Alam, and Rishemjit Kaur. 2023. Neural machine translation for low-resource languages: A survey. *ACM Comput. Surv.* (2023).

[197] Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. 2020. PlotMachines: Outline-conditioned generation with dynamic plot state tracking. In *EMNLP*.

[198] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. DeepSpeed: System optimizations enable training deep learning models with over 100 billion parameters. In *SIGKDD*.

[199] Ricardo Rei, Craig Stewart, Ana C. Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *EMNLP*.

[200] Machel Reid, Junjie Hu, Graham Neubig, and Yutaka Matsuo. 2021. AfroMT: Pretraining strategies and reproducible benchmarks for translation of 8 African languages. In *EMNLP*.

[201] Shuo Ren, Yu Wu, Shujie Liu, Ming Zhou, and Shuai Ma. 2019. Explicit cross-lingual pre-training for unsupervised machine translation. In *EMNLP/IJCNLP*.

[202] Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2020. Investigating pretrained language models for graph-to-text generation. *arXiv preprint arXiv:2007.08426* (2020).

[203] Leonardo F. R. Ribeiro, Yue Zhang, and Iryna Gurevych. 2021. Structural adapters in pretrained language models for amr-to-text generation. In *EMNLP*.

[204] Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. 2021. Recipes for building an open-domain chatbot. In *EACL*.

[205] Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *TACL* (2020).

[206] Ananya B. Sai, Tanay Dixit, Vignesh Nagarajan, Anoop Kunchukuttan, Pratyush Kumar, Mitesh M. Khapra, and Raj Dabre. 2023. IndicMT Eval: A dataset to meta-evaluate machine translation metrics for indian languages. In *ACL*.

[207] Itsumi Saito, Kyosuke Nishida, Kosuke Nishida, and Junji Tomita. 2020. Abstractive summarization with combination of pre-trained sequence-to-sequence and saliency models. *arXiv preprint arXiv:2003.13028* (2020).

[208] Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in NLP. *Trans. Assoc. Comput. Linguistics* 9 (2021), 1408–1424.

[209] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[210] Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2020. ColdGANs: Taming language GANs with cautious sampling strategies. In *NeurIPS*.

[211] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*.

[212] Thibault Sellam, Dipanjan Das, and Ankur P. Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *ACL*.

[213] Stanislau Semeniuta, Aliaksei Severyn, and Sylvain Gelly. 2018. On accurate evaluation of GANs for language generation. *arXiv preprint arXiv:1806.04936* (2018).

[214] Anastasia Shimorina. 2018. Human vs automatic metrics: On the importance of correlation design. *arXiv* (2018).

[215] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting knowledge from language models with automatically generated prompts. In *EMNLP*.

[216] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-LM: Training multi-billion parameter language models using model parallelism. *arXiv* (2019).

[217] Karan Singhal, Shekoofeh Azizi, Tao Tu, S. Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Kumar Tanwani, Heather Cole-Lewis, Stephen Pfohl, Perry Payne, Martin Seneviratne, Paul Gamble, Chris Kelly, Nathaneal Schärli, Aakanksha Chowdhery, Philip Andrew Mansfield, Blaise Agüera y Arcas, Dale R.Webster, Gregory S. Corrado, Yossi Matias, Katherine Chou, Juraj Gottweis, Nenad Tomasev, Yun Liu, Alvin Rajkomar, Joelle K. Barral, Christopher Semturs, Alan Karthikesalingam, and Vivek Natarajan. 2022. Large language models encode clinical knowledge. *CoRR* abs/2212.13138 (2022).

[218] Linda B. Smith and Michael Gasser. 2005. The Development of Embodied Cognition: Six Lessons from Babies. (2005).

[219] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *ICML*.

[220] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Santilli, Andreas Stuhlmüller, Andrew M. Dai, Andrew La, Andrew K. Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, and Ayla Karakas. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *CoRR* abs/2206.04615 (2022).

[221] Ilia Stepin, Jose M. Alonso, Alejandro Catala, and Martín Pereira-Fariña. 2021. A survey of contrastive and counterfactual explanation generation methods for explainable artificial intelligence. *IEEE Access* 9 (2021), 11974–12001.

[222] Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. 2018. Blockwise parallel decoding for deep autoregressive models. In *NeurIPS*.

[223] Asa Cooper Stickland, Xian Li, and Marjan Ghazvininejad. 2021. Recipes for adapting pre-trained monolingual and multilingual models to machine translation. In *EACL*.

[224] Jianlin Su. 2023. *Transformer Upgrade Path: 12, Infinite Extrapolation of ReRoPE?*

[225] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. RoFormer: Enhanced transformer with rotary position embedding. *CoRR* abs/2104.09864 (2021).

[226] Yixuan Su, Tian Lan, Yan Wang, Dani Yogatama, Lingpeng Kong, and Nigel Collier. 2022. A contrastive framework for neural text generation. In *NeurIPS*.

[227] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. 2019. VideoBERT: A joint model for video and language representation learning. In *ICCV*.

[228] Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. 2022. A length-extrapolatable transformer. *arXiv* (2022).
[229] Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, Weixin Liu, Zhihua Wu, Weibao Gong, Jianzhong Liang, Zhizhou Shang, Peng Sun, Wei Liu, Xuan Ouyang, Dianhai Yu, Hao Tian, Hua Wu, and Haifeng Wang. 2021. ERNIE 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv* (2021).
[230] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*.
[231] Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2021. Multilingual translation from denoising pre-training. In *ACL/IJCNLP Findings*.
[232] Tao Tao, Xuanhui Wang, Qiaozhu Mei, and ChengXiang Zhai. 2006. Language model information retrieval with document expansion. In *NAACL*.
[233] Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. 2023. UL2: Unifying language learning paradigms. In *ICLR*.
[234] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and efficient foundation language models. *CoRR* abs/2302.13971 (2023).
[235] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu,Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR* abs/2307.09288 (2023).
[236] Jonathan Uesato, Nate Kushman, Ramana Kumar, H. Francis Song, Noah Y. Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. Solving math word problems with process- and outcome-based feedback. *CoRR* abs/2211.14275 (2022).
[237] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.
[238] Ashwin K. Vijayakumar, Michael Cogswell, Ramprasath R. Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv* (2016).
[239] Takashi Wada and Tomoharu Iwata. 2018. Unsupervised cross-lingual word embedding by multilingual neural language models. *arXiv preprint arXiv:1809.02306* (2018).
[240] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *NeurIPS*.
[241] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.
[242] Thomas Wang, Adam Roberts, Daniel Hesslow, Teven Le Scao, Hyung Won Chung, Iz Beltagy, Julien Launay, and Colin Raffel. 2022. What language model architecture and pretraining objective works best for zero-shot generalization?. In *ICML*.
[243] Teng Wang, Ruimao Zhang, Zhichao Lu, Feng Zheng, Ran Cheng, and Ping Luo. 2021. End-to-end dense video captioning with parallel decoding. In *ICCV*. IEEE, 6827–6837.
[244] Wei Wang, Piji Li, and Hai-Tao Zheng. 2021. Consistency and coherency enhanced story generation. In *ECIR*.
[245] Xiaohui Wang, Ying Xiong, Yang Wei, Mingxuan Wang, and Lei Li. 2021. LightSeq: A high performance inference library for transformers. In *NAACL-HLT Industry*.
[246] Yequan Wang, Jiawen Deng, Aixin Sun, and Xuying Meng. 2022. Perplexity from PLM is unreliable for evaluating text quality. *arXiv preprint arXiv:2210.05892* (2022).
[247] Kellie Webster, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, and Slav Petrov. 2020. Measuring and reducing gendered correlations in pre-trained models. *CoRR* abs/2010.06032 (2020).
[248] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Trans. Mach. Learn. Res.* 2022 (2022).

[249] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, Qun Liu and David Schlangen (Eds.). Association for Computational Linguistics, 38–45.

[250] Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. TransferTransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149* (2019).

[251] Qiaolin Xia, Haoyang Huang, Nan Duan, Dongdong Zhang, Lei Ji, Zhifang Sui, Edward Cui, Taroon Bharti, and Ming Zhou. 2021. XGPT: Cross-modal generative pre-training for image captioning. In *NLPCC*.

[252] Guangxuan Xiao, Ji Lin, Mickaël Seznec, Julien Demouth, and Song Han. 2022. SmoothQuant: Accurate and efficient post-training quantization for large language models. *arXiv* (2022).

[253] Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. 2020. MEGATRON-CNTRL: Controllable story generation with external knowledge using large-scale language models. In *EMNLP*.

[254] Lanqing Xue, Kaitao Song, Duocai Wu, Xu Tan, Nevin L. Zhang, Tao Qin, Wei-Qiang Zhang, and Tie-Yan Liu. 2021. DeepRapper: Neural rap generation with rhyme and rhythm modeling. In *ACL/IJCNLP*.

[255] Yu Yan, Fei Hu, Jiusheng Chen, Nikhil Bhendawade, Ting Ye, Yeyun Gong, Nan Duan, Desheng Cui, Bingyu Chi, and Ruifei Zhang. 2021. FastSeq: Make sequence generation faster. *arXiv preprint arXiv:2106.04718* (2021).

[256] Jiacheng Yang, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Weinan Zhang, Yong Yu, and Lei Li. 2020. Towards making the most of BERT in neural machine translation. In *AAAI*.

[257] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

[258] Zhen Yang, Bojie Hu, Ambyera Han, Shen Huang, and Qi Ju. 2020. CSP: Code-switching pre-training for neural machine translation. In *EMNLP*.

[259] Ziyi Yang, Chenguang Zhu, Robert Gmyr, Michael Zeng, Xuedong Huang, and Eric Darve. 2020. TED: A pretrained unsupervised summarization model with theme modeling and denoising. In *EMNLP Findings*.

[260] Weiqiu You, Simeng Sun, and Mohit Iyyer. 2020. Hard-coded gaussian attention for neural machine translation. In *ACL*.

[261] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence generative adversarial nets with policy gradient. In *AAAI*.

[262] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big Bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, Virtual*.

[263] Munazza Zaib, Quan Z. Sheng, and Wei Emma Zhang. 2020. A short survey of pre-trained language models for conversational AI-A new age in NLP. In *ACSW*.

[264] Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, et al. 2023. GLM-130B: An open bilingual pre-trained model. In *ICLR*.

[265] Wei Zeng, Xiaozhe Ren, Teng Su, Hui Wang, Yi Liao, Zhiwei Wang, Xin Jiang, ZhenZhang Yang, Kaisheng Wang, Xiaoda Zhang, Chen Li, Ziyan Gong, Yifan Yao, Xinjing Huang, Jun Wang, Jianfeng Yu, Qi Guo, Yue Yu, Yan Zhang, Jin Wang, Hengtao Tao, Dasen Yan, Zexuan Yi, Fang Peng, Fangqing Jiang, Han Zhang, Lingfeng Deng, Yehong Zhang, Zhe Lin, Chao Zhang, Shaojie Zhang, Mingyue Guo, Shanzhi Gu, Gaojun Fan, Yaowei Wang, Xuefeng Jin, Qun Liu and , Yonghong Tian. 2021. PanGu-α: Large-scale autoregressive pretrained chinese language models with auto-parallel computation. *arXiv preprint arXiv:2104.12369* (2021).

[266] Yan Zeng and Jian-Yun Nie. 2020. Generalized conditioned dialogue generation based on pre-trained language model. *arXiv preprint arXiv:2010.11140* (2020).

[267] Yan Zeng and Jian-Yun Nie. 2021. A simple and efficient multi-task learning approach for conditioned dialogue generation. In *NAACL-HLT*.

[268] ChengXiang Zhai and John D. Lafferty. 2001. Model-based feedback in the language modeling approach to information retrieval. In *CIKM*.

[269] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *ICML*.

[270] Longxiang Zhang, Renato Negrinho, Arindam Ghosh, Vasudevan Jagannathan, Hamid Reza Hassanzadeh, Thomas Schaaf, and Matthew R. Gormley. 2021. Leveraging pretrained models for automatic summarization of doctor-patient conversations. In *EMNLP Findings*.

[271] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512* (2023).

[272] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating text generation with BERT. In *ICLR*.
[273] Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *ACL*.
[274] Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *ACL*.
[275] Zhengyan Zhang, Yuxian Gu, Xu Han, Shengqi Chen, Chaojun Xiao, Zhenbo Sun, Yuan Yao, Fanchao Qi, Jian Guan, Pei Ke, Yanzheng Cai, Guoyang Zeng, Zhixing Tan, Zhiyuan Liu, Minlie Huang, Wentao Han, Yang Liu, Xiaoyan Zhu, and Maosong Sun. 2021. CPM-2: Large-scale cost-effective pre-trained language models. *AI Open* 2 (2021), 216–224.
[276] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *ACL*.
[277] Zhengyan Zhang, Xu Han, Hao Zhou, Pei Ke, Yuxian Gu, Deming Ye, Yujia Qin, YuSheng Su, Haozhe Ji, Jian Guan, Fanchao Qi, Xiaozhi Wang, Yanan Zheng, Guoyang Zeng, Huanqi Cao, Shengqi Chen, Daixuan Li, Zhenbo Sun, Zhiyuan Liu, Minlie Huang, Wentao Han, Jie Tang, Juanzi Li, Xiaoyan Zhu, and Maosong Sun. 2020. CPM: A large-scale generative chinese pre-trained language model. *arXiv preprint arXiv:2012.00413* (2020).
[278] Zhilu Zhang and Mert R. Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. In *NeurIPS*.
[279] Zheng Zhang, Ryuichi Takanobu, Qi Zhu, MinLie Huang, and XiaoYan Zhu. 2020. Recent advances and challenges in task-oriented dialog systems. *Sci. China Technol. Sci.* (2020).
[280] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).
[281] Hao Zheng and Mirella Lapata. 2019. Sentence centrality revisited for unsupervised summarization. In *ACL*.
[282] Ming Zhong, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. DialogLM: Pre-trained model for long dialogue understanding and summarization. *arXiv preprint arXiv:2109.02492* (2021).
[283] Li Zhou, Jianfeng Gao, Di Li, and Heung-Yeung Shum. 2020. The design and implementation of xiaoice, an empathetic social chatbot. *Comput. Linguistics* (2020).
[284] Wangchunshu Zhou, Yuchen Eleanor Jiang, Ethan Wilcox, Ryan Cotterell, and Mrinmaya Sachan. 2023. Controlled text generation with natural language instructions. *CoRR* abs/2304.14293 (2023).
[285] Wangchunshu Zhou, Dong-Ho Lee, Ravi Kiran Selvam, Seyeon Lee, and Xiang Ren. 2021. Pre-training text-to-text transformers for concept-centric common sense. In *ICLR*.
[286] Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *SIGIR*.
[287] Ran Zmigrod, S. J. Mielke, Hanna M. Wallach, and Ryan Cotterell. 2019. Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology. In *ACL*.
[288] Yicheng Zou, Bolin Zhu, Xingwu Hu, Tao Gui, and Qi Zhang. 2021. Low-resource dialogue summarization with domain-agnostic multi-source pretraining. In *EMNLP*.

Received 13 May 2022; revised 24 January 2024; accepted 31 January 2024