All NLP Tasks Are Generation Tasks: A General Pretraining Framework

Zhengxiao Du [*12] **Yujie Qian** [*3] **Xiao Liu** [12] **Ming Ding** [12] **Jiezhong Qiu** [12] **Zhilin Yang** [42] **Jie Tang** [12]

Abstract

There have been various types of pretraining architectures including autoregressive models (e.g., GPT), autoencoding models (e.g., BERT), and encoder-decoder models (e.g., T5). On the other hand, NLP tasks are different in nature, with three main categories being classification, unconditional generation, and conditional generation. However, none of the pretraining frameworks performs the best for all tasks, which introduces inconvenience for model development and selection. We propose a novel pretraining framework GLM (General Language Model) to address this challenge. Compared to previous work, our architecture has three major benefits: (1) it performs well on classification, unconditional generation, and conditional generation tasks with one single pretrained model; (2) it outperforms BERT-like models on classification due to improved pretrain-finetune consistency; (3) it naturally handles variable-length blank filling which is crucial for many downstream tasks. Empirically, GLM substantially outperforms BERT on the SuperGLUE natural language understanding benchmark with the same amount of pre-training data. Moreover, GLM with 1.25× parameters of $\text{BERT}_{\text{Large}}$ achieves the best performance in NLU, conditional and unconditional generation at the same time, which demonstrates its generalizability to different downstream tasks.[1]

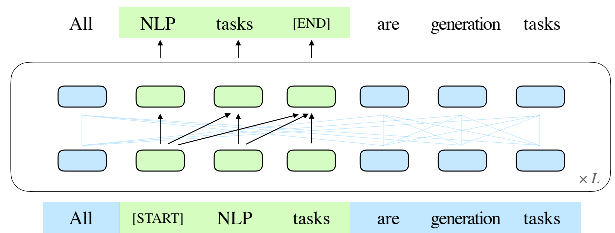

Figure 1. Illustration of GLM. We blank out text spans (green part) and GLM is trained to generate them in an autoregressive fashion. (Some attention edges are omitted; cf. Figure 2.)

1. Introduction

Large-scale language models pre-trained on web texts have substantially advanced the state of the art in various NLP tasks, such as natural langage understanding and text generation (Radford et al., 2018a; Devlin et al., 2019; Yang et al., 2019; Radford et al., 2018b; Raffel et al., 2020; Lewis et al., 2019; Brown et al., 2020). The downstream task performance as well as the scale of the parameters have also constantly increased in the past few years.

In general, existing pretraining frameworks can be categorized into three families: *augoregressive models*, *autoencoding models*, and *encoder-decoder models*. Autoregressive models, such as GPT (Radford et al., 2018a), learn left-to-right language models. While they have succeeded in long-text generation and shown strong few-shot learning ability when scaled to billions of parameters (Radford et al., 2018b; Brown et al., 2020), the inherent disadvantage is that the unidirectional attention mechanism cannot fully capture the interaction of the context tokens. Autoencoding models, such as BERT (Devlin et al., 2019), learn bidirectional Transformers as context encoders via denoising objectives. These encoders generate contexualized representations which excel at natural langugage understanding tasks, but could not be directly applied for text generation. Encoder-decoder models adopt bidirectional attention for the encoder model, unidirectional attention for the decoder model, and cross attention to connect them (Song et al., 2019; Bi et al., 2020). They are typically deployed in conditional text generation tasks such as text summarization and response generation. Table 1 compares different pretraining frameworks.

[*]Equal contribution [1]Department of Computer Science and Technology, Tsinghua Univerisity, Beijing, China [2]Beijing Academy of Artificial Intelligence, Beijing, China [3]Massachusetts Institute of Technology, Cambridge, U.S.A. [4]Recurrent AI, Ltd.. Correspondence to: Zhilin Yang <kimi_yang@rcrai.com>, Jie Tang <jietang@tsinghua.edu.cn>.

[1]The codes and pre-trained models are available at https://github.com/THUDM/GLM

Table 1. Summary of the pre-training frameworks. "Cond. Gen." and "Uncond. Gen." refer to conditional and unconditional text generation, respectively. "✓" means "is good at", "—" means "could be adapted to", and "×" means "cannot be directly applied to". We define unconditional generation as the task of generating text without further training as in a standard language model, while conditional generation refers to seq2seq tasks such as text summarization.

Framework	NLU	Cond. Gen.	Uncond. Gen.
Autoregressive	—	—	✓
Autoencoding	✓	×	×
Encoder-Decoder	—	✓	—
GLM	✓	✓	✓

None of these pretraining frameworks performs best for all the NLP tasks. Previous works have tried to unify different frameworks by combining their objectives via multi-task learning (Dong et al., 2019; Bao et al., 2020). However, the autoencoding and autoregressive objectives differ by nature, and simple unification cannot fully inherit the advantages of both frameworks.

In this paper, we propose a novel pre-training method, called GLM, based on autoregressive blank-filling. We randomly blank out continuous spans of tokens from the input text, following the idea of autoencoding, and train the model to reconstruct the spans, following the idea of autoregressive pre-training. To learn both the bidirectional and unidirectional attention mechanism in a single framework, we divide the input text into two parts, where the unmasked tokens can attend to each other, but the masked tokens cannot attend to subsequent masked tokens. We also propose a 2D positional encoding technique to indicate inter- and intra- span position information. Figure 1 illustrates our pre-training objective. As a result, GLM learns both contextual representation and autoregressive generation during pre-training.

When finetuning our models on downstream tasks, we reformulate them as blank-filling generation, inspired by (Schick & Schütze, 2020a;b). Each task is associated with a human-crafted cloze question, and the model predicts the answer to the cloze. For example, a sentiment classification task is reformulated as filling the blank in "[SENTENCE]. It's really __". The prediction of "good" or "bad" indicates the sentiment being positive or negative. With such formulation, GLM benefits from the consistency between pretraining and finetuning, because both pretraining and finetuning involves training the model to generate text given context. As a result, GLM is more suitable for downstream classification tasks compared to BERT-like models. To make our pre-training method better suited for text generation tasks, we also study a multi-task pre-training setup, where the model is jointly trained to reconstruct masked spans and generate longer text.

Empirically, we show that with the same pre-training data and a close amount of computational cost, GLM significantly outperforms BERT on the SuperGLUE natural language understanding benchmark by a large margin of 4.6% – 5.0%. GLM also outperforms RoBERTa, T5, and BART when pre-trained on the same, larger corpus (158GB). Moreover, compared with standalone baselines, GLM with multi-task pre-training can achieves improvements in comprehension, conditional generation, and language modeling tasks with shared parameters.

2. GLM Pre-training Framework

In this section, we describe the model architecture and the pre-training method of GLM. We also introduce how our proposed model is finetuned for downstream natural language understanding and generation tasks.

2.1. Model Architecture

GLM uses the Transformer architecture similar to BERT (Devlin et al., 2019). First, the input tokens $[x_1, x_2, \cdots, x_n]$ are projected into embedding vectors $\mathbf{H}^0 = [\boldsymbol{h}_1^0, \boldsymbol{h}_2^0, \cdots, \boldsymbol{h}_n^0]$ via a learnable embedding table. Then L transformer layers are applied to compute the hidden states of the tokens. Each layer consists of a multi-head self-attention layer and a position-wise fully connected feed-forward network. Specifically, a self-attention head in layer l is defined as

$$\begin{aligned} &\mathbf{Q}^l = \mathbf{H}^l\mathbf{W}_Q^l, \mathbf{K}^l = \mathbf{H}^l\mathbf{W}_K^l, \mathbf{V}^l = \mathbf{H}^l\mathbf{W}_V^l \\ &\mathbf{A}^l = \mathrm{softmax}\left(\frac{\mathbf{Q}^l(\mathbf{K}^l)^T}{\sqrt{d_k}} + \mathbf{M}\right)\mathbf{V}^l \end{aligned} \tag{1}$$

where $\mathbf{W}_Q^l, \mathbf{W}_K^l, \mathbf{W}_V^l \in \mathbb{R}^{d_h \times d_k}$ are model parameters. $\mathbf{M} \in \mathbb{R}^{n\times n}$ is the matrix of self-attention mask, $M_{ij} = 0$ indicates that token x_i is allowed to attend to token x_j and $M_{ij} = -\infty$ indicates prevention.

Following Megatron-LM (Shoeybi et al., 2019), we make two modifications to the BERT architecture. (1) We rearrange the order of layer normalization and the residual connection, which has been shown critical when scaling to large BERT-style models. (2) We replace the feed-forward network for token prediction with a linear layer, therefore the output position i is defined as

$$\boldsymbol{p}_i = \mathrm{softmax}(\boldsymbol{h}_i^L \mathbf{W}_o) \tag{2}$$

where $\mathbf{W}_o \in \mathbb{R}^{d_h \times |\mathcal{V}|}$ and $|\mathcal{V}|$ is the vocabulary size.

2.2. Autoregressive Blank Infilling

GLM is trained by optimizing an *autoregressive blank infilling* task. Given an input text $\boldsymbol{x} = [x_1, \cdots, x_n]$, multiple text spans $\{\boldsymbol{s}_1, \cdots, \boldsymbol{s}_m\}$ are sampled, where each span $\boldsymbol{s}_i$

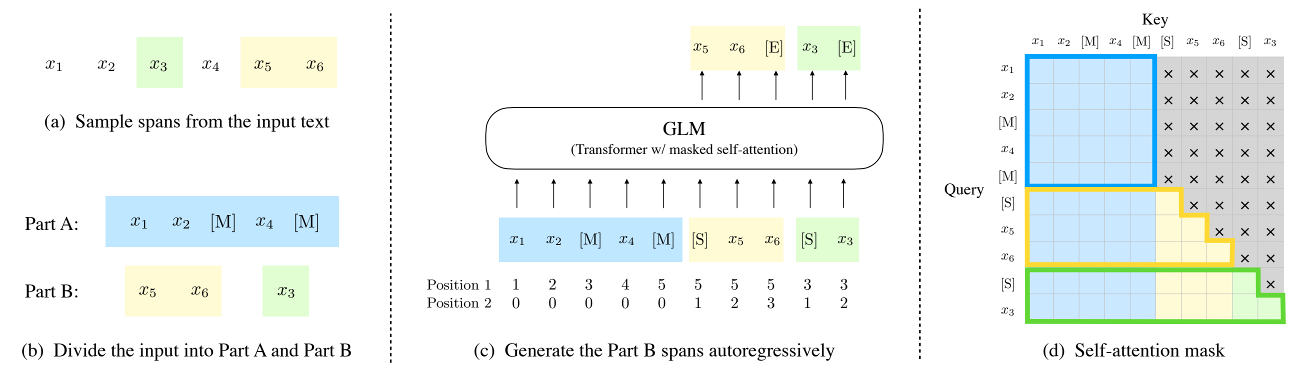

Figure 2. GLM pre-training framework. (a) The original text is $[x_1, x_2, x_3, x_4, x_5, x_6]$, and two spans $[x_3]$ and $[x_5, x_6]$ are sampled. (b) Replace the sampled spans with [MASK] tokens to form Part A, and shuffle the sampled spans to form Part B. (c) GLM is trained to autorergessively generate Part B. Each span is prepended with a [START] token as input and appended with a [END] token as output. 2D positional encoding is used to represent inter- and intra- span position. (d) Self-attention mask controls the attention mechanism, where the gray areas are masked out. Part A tokens can attend to A (the blue-framed area) but not B. Part B tokens can attend to A and their antecedent tokens in B (the yellow- and green-framed areas denote the tokens which the two spans in B can attend to). [M], [S], and [E] represents [MASK], [START], and [END] respectively.

corresponds to a series of consecutive tokens $[s_{i,1}, \cdots, s_{i,l_i}]$ in $\boldsymbol{x}$. The number and length of text spans depend on the pre-training objective, which is described in Section 2.3. Each span is replaced with a single [MASK] token, forming the corrupted text $\boldsymbol{x}_{\text{corrupt}}$. The model predicts the missing tokens in the spans from the corrupted text in an autoregressive manner, which means when predicting the missing tokens in a span, the model has access to the corrupted text *and* the previously predicted spans. To fully capture the interdependence between different spans, we randomly permute the order of the spans, similar to (Yang et al., 2019). Formally, let Z_m be the set of all possible permutations of the length-m index sequence $[1, 2, \cdots, m]$, and $\boldsymbol{s}_{\boldsymbol{z}_{<i}}$ be $[\boldsymbol{s}_{z_1}, \cdots, \boldsymbol{s}_{z_{i-1}}]$, we define the pre-training objective as

$$\max_{\theta} \mathbb{E}_{\boldsymbol{z}\sim Z_m}\left[\sum_{i=1}^{m} \log p_\theta(\boldsymbol{s}_{z_i} | \boldsymbol{x}_{\text{corrupt}}, \boldsymbol{s}_{\boldsymbol{z}_{<i}})\right] \quad (3)$$

The task differs from that of SpanBERT (Joshi et al., 2020) in that the number of missing tokens in a span is unknown to the model. Also, GLM predicts the missing tokens also in an autoregressive manner. We always generate the tokens in each blank following the left-to-right order, i.e. the probability of generating the span $\boldsymbol{s}_i$ is factorized as:

$$p_\theta(\boldsymbol{s}_i | \boldsymbol{x}_{\text{corrupt}}, \boldsymbol{s}_{\boldsymbol{z}_{<i}}) = \prod_{j=1}^{l_i} p(s_{i,j} | \boldsymbol{x}_{\text{corrupt}}, \boldsymbol{s}_{\boldsymbol{z}_{<i}}) \quad (4)$$

Specifically, we implemented the autoregressive blank infilling task with the following tricks. The input tokens are divided into two parts. Part A consists of the corrupted text $\boldsymbol{x}_{\text{corrupt}}$ where the sampled text spans are replaced by [MASK] tokens. Part B consists of the tokens in the masked spans. Tokens in Part A can attend to all the tokens in A, but cannot attend to any token in B. Tokens in Part B can attend to tokens in A as well as its antecedents in B, but cannot attend to any subsequent positions in B. Similar to the decoder in the original Transformer model, the tokens in each span are padded with two special tokens [START] and [END] at the beginning and the end, respectively. In this way, our model automatically learns a bidirectional encoder (Part A) and a unidirectional decoder (Part B) in a single model. The whole implementation is illustrated in Figure 2.

2.2.1. 2D Positional Encoding

One of the challenges in the above task is how to encode the positional information. Transformers rely on positional encodings added to the input embeddings to inject the absolute and relative positions of the tokens. As a similar autoregressive model, XLNet (Yang et al., 2019) encodes the original positions for tokens in Part A and B. As a result, the model can perceive the number of missing tokens in a span.

We propose a novel 2D positional encodings to address the challenge. Specifically, each token is encoded with two position ids. The first position id represents the position in the corrupted text $\boldsymbol{x}_{\text{corrupt}}$. For tokens in B, it is the position of the corresponding [MASK] token. The second position id represents the intra-span position. For tokens in A, the second position id is 0. For tokens in B, it ranges from 1 to the length of the span. The two position ids are projected into two position vectors via two separate embedding tables and added to the input embeddings.

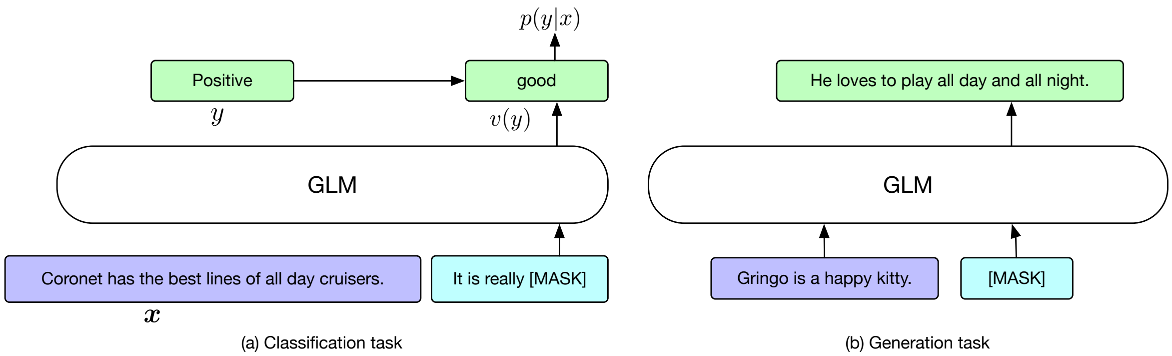

Figure 3. GLM finetune framework. (a) Formulation of the sentiment classification task as blank infilling with GLM. (b) GLM for text generation given the context. This can be the language modeling in the zero-shot setting, or seq2seq with fine-tuning.

2.3. Pre-Training Objectives

As described in Section 2.2, the pre-training objective of GLM is defined as autoregressive generation of the masked spans. Following BERT, the masked spans make up 15% of the original tokens. Empirically, we have found that the ratio is critical for good performance on downstream natural langauge understanding tasks. The span lengths are drawn from a Poisson distribution with $\lambda = 3$, following BART (Lewis et al., 2019). We repeatedly sample new spans until more than 15% of the original tokens are masked.

Similar to other BERT-style models, GLM masks short spans and is suited for NLU tasks. However, we are interested in pre-training a single model that can handle both NLU and text generation. We further study a *multi-task pre-training* setup, in which a second objective of generating longer text is jointly optimized with GLM. Specifically, we sample a single span that covers 50%–100% of the original tokens. The span length is sampled from a uniform distribution. The new objective is defined in the same way as the original objective. The only difference is there is only one but much longer span.

2.4. Finetuning GLM

Previously, for downstream NLU tasks, a linear classifier takes the representations produced by pre-trained models as inputs and predicts the correct answer. For token classification tasks, the inputs are representations of the target tokens. For sequence classification tasks, the input is the representation of the [CLS] token, which works as the representation of the sequence. The practices are different from the pre-training task of cloze filling, leading to inconsistency between pre-training and finetuning.

Instead, we formulate classification tasks in NLU as generation tasks of blank infilling, following PET (Schick & Schütze, 2020b). Formally, given a labeled example $(\boldsymbol{x}, y)$, we map the input text $\boldsymbol{x}$ to a cloze question $c(\boldsymbol{x})$ via a pattern containing a single mask token [MASK]. The pattern should be similar to natural languages in the pre-training dataset. For example, the text in a sentiment classification task can be formulated as "[SENTENCE]. It's really [MASK]". The lable y is also mapped to an answer to the cloze, called the verbalizer $v(y)$. In the sentiment classification task, the label "positive" or "negative" is mapped to the word "good" or "bad" in the blank. The probability of the sentence being positive or negative is proportional to predicting "good" or "bad" for the blank. Therefore, the conditional probability of y given $\boldsymbol{x}$ is

$$p(y|\boldsymbol{x}) = \frac{p(v(y)|c(\boldsymbol{x}))}{\sum_{y' \in \mathcal{Y}} p(v(y')|c(\boldsymbol{x}))} \tag{5}$$

where $\mathcal{Y}$ is the label set. Then we can finetune GLM with the cross entropy loss.

GLM is especially suited to this setting for two reasons. Firstly, GLM can naturally handle filling the blank of unknown length. BERT-style models have to know the number of missing tokens via the number of [MASK] tokens or the positional encodings. Secondly, GLM breaks BERT's independence assumption between masked tokens and thus can capture more dependences.

For text generation tasks, we directly apply GLM as an autoregressive model. The given context constitutes the Part A of the input, with a [MASK] token at the end. Then GLM generates the text in Part B autoregressively. We can directly apply the pre-trained GLM for unconditional generation, or finetune GLM on downstream conditional generation tasks. The finetuning framework is illustrated in Figure 3.

2.5. Discussion and Analysis

2.5.1. Comparison with BERT

BERT is pre-trained with the autoencoding objective. The model needs to predict a subset of tokens in the original text replaced with the [MASK] token. Since the model predicts the masked tokens independently, BERT fails to capture the interdependence of masked tokens (Yang et al., 2019). Another disadvantage of BERT is that it cannot handle blank filling of multiple-token answers properly. To infer the probability of an answer of length l, BERT has to perform l consecutive predictions. To predict an answer from scratch, it is necessary to enumerate the set of all possible lengths, since BERT needs to change the number of [MASK] tokens according to the length of the answer.

2.5.2. Comparison with XLNet

Both GLM and XLNet (Yang et al., 2019) are pre-trained with autoregressive objectives. There are three differences when comparing GLM with XLNet. Firstly, XLNet uses the original position encoding before corruption. During inference, XLNet either needs to know the length of the answer or enumerate the set of possible lengths. Secondly, XLNet uses two-stream self-attention combined with target-aware representations, instead of right shift, to solve information leak with the transformer architecture. The two-stream attention increases the time cost of pre-training. Thirdly, XLNet decides whether a token should be predicted independently, while GLM first samples the lengths of masked spans.

2.5.3. Comparison with Encoder-Decoder Models

T5 (Raffel et al., 2020) proposes a blank-filling objective similar to that of GLM to pre-train the encoder-decoder transformer model. Instead, GLM uses a single transformer encoder model to learn both the bidirectional and unidirectional attention. With shared parameters for two types of attention, GLM can be more parameter-efficient than the encoder-decoder architecture. On the other hand, T5 (Raffel et al., 2020) uses independent positional encodings for tokens in the encoder and decoder, relying on sentinel tokens to distinguish different masked spans. In downstream tasks, at most one of these sentinel tokens is used, leading to a waste of model capacity and inconsistency between pre-training and finetuning.

2.5.4. Comparison with UniLM

UniLM (Dong et al., 2019) unifies different pre-training objectives under the autoencoding framework by changing the attention mask among bidirectional, unidirectional, and cross attention. Compared with autoregressive models, the model cannot fully capture the current token's dependence on previous tokens, due to the independence assumption of autoencoding models. Finetuning GLM on downstream generation tasks also relies on masked language modeling, which is less efficient than autoregressive models.

3. Experiments

In this section, we describe our experiments in two different settings. In the first setting, we pre-train GLM with a single BERT-style objective and compare it with BERT-like models on NLU tasks. We show that the autoregressive blank filling pre-training, combined with the new formulation of classification tasks, can outperform finetuning bidirectional encoders with linear classifiers. In the second setting, we pre-train GLM with both the BERT-style objective and the generation objective. We show that GLM can effectively share model parameters for different tasks.

3.1. Pre-training Data and Implementation

Following BERT (Devlin et al., 2019), we use the BooksCorpus (Zhu et al., 2015) and English Wikipedia as our pre-training data in the experiments. We use the same wordpiece tokenizer with 30k vocabulary as BERT. Each input text is a document sampled from the corpus, with a start-of-sequence token [SOS] prepended at the beginning and an end-of-sequence token [EOS] appended at the end. If the document is longer than the maximum sequence length of the transformer model, we randomly sample a continuous segment of the maximum length.

For the single-objective pre-training, we train a GLM $_{\text{Base}}$ and a GLM $_{\text{Large}}$ with the same architectures as BERT$_{\text{Base}}$ and BERT$_{\text{Large}}$, with 110M and 340M parameters respectively. For the multi-objective pre-training, we additionally train two GLM models with 410M (30 layers, hidden size 1024, and 16 attention heads) and 515M (30 layers, hidden size 1152, and 18 attention heads) parameters. The output matrix of the softmax prediction is tied with the input embedding matrix. The maximum sequence length is set to 512.

We use the AdamW optimizer (Loshchilov & Hutter, 2019) and set $\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1e{-}6$. We adopt a linear warm-up of the learning rate over the first 8000 steps and a cosine decay after that. The peak learning rate is set to $4e{-}4$ for GLM $_{\text{Base}}$ and $2e{-}4$ for GLM $_{\text{Large}}$. The weight decay is 0.1 and the dropout rate is 0.1. We train GLM on 64 Nvidia V100 GPUs for 200K steps with a batch size of 1024, which takes about 2.5 days.

For comparison with SOTA pre-trained language models, we also train a GLM $_{\text{Large}}$ model with RoBERTa (Liu et al., 2019) data, tokenization, and hyperparameters, denoted as GLM $_{\text{RoBERTa}}$. The stories dataset (Trinh & Le, 2019) used by RoBERTa is already unavailable. Therefore we replace the OpenWebText (38GB) (Gokaslan & Cohen, 2019) with OpenWebText2 (66GB). The whole dataset totals 158GB of uncompressed text, close in size to RoBERTa's 160GB dataset. Due to resource limit, we only pre-train the model for 250,000 steps, which are half of RoBERTa

Table 2. Results on the SuperGLUE dev set. Models with * are pre-trained for two times the number of steps of other methods.

Model	ReCoRD F1/Acc.	COPA Acc.	WSC Acc.	RTE Acc.	BoolQ Acc.	WiC Acc.	CB F1/Acc.	MultiRC F1a/EM	Avg
$\text{BERT}_{\text{Base}}$	65.4/64.9	66.0	65.4	70.0	74.9	**68.8**	70.9/76.8	68.4/21.5	66.1
GLM $_{\text{Base}}$	**73.5/72.8**	**71.0**	**72.1**	**71.2**	**77.0**	64.7	**89.5/85.7**	**72.1/26.1**	**70.7**
$\text{BERT}_{\text{Large}}$	76.3/75.6	69.0	64.4	73.6	80.1	**71.0**	94.8/92.9	71.9/24.1	72.0
$\text{UniLM}_{\text{Large}}$	80.0/79.1	72.0	65.4	76.5	80.5	69.7	91.0/91.1	77.2/38.2	74.1
GLM $_{\text{Large}}$	81.7/81.1	76.0	**81.7**	74.0	**82.1**	68.5	96.1/94.6	**77.1/36.3**	77.0
GLM $_{\text{Large}}$ (multi-task)	80.2/79.6	77.0	78.8	76.2	79.8	63.6	**97.3/96.4**	74.6/32.1	75.7
GLM $_{\text{410M}}$ (multi-task)	81.5/80.9	80.0	**81.7**	**79.4**	81.9	69.0	93.2/96.4	76.2/35.5	78.0
GLM $_{\text{515M}}$ (multi-task)	**82.3/81.7**	**85.0**	**81.7**	79.1	81.3	69.4	95.0/96.4	77.2/35.0	**78.8**
T5_{Base}	76.2/75.4	73.0	79.8	78.3	80.8	67.9	94.8/92.9	76.4/40.0	76.0
T5_{Large}	85.7/85.0	78.0	**84.6**	84.8	84.3	71.6	96.4/98.2	80.9/46.6	81.2
$\text{BART}_{\text{Large}}$ *	88.3/87.8	60.0	65.4	84.5	84.3	69.0	90.5/92.9	81.8/48.0	76.0
$\text{RoBERTa}_{\text{Large}}$ *	89.0/88.4	**90.0**	63.5	87.0	**86.1**	**72.6**	96.1/94.6	**84.4/52.9**	81.5
GLM $_{\text{RoBERTa}}$	**89.6/89.0**	82.0	83.7	**87.7**	84.7	71.2	**98.7/98.2**	82.4/50.1	**82.9**

and BART's training steps and close to T5 in number of trained tokens. Other hyperparameters are the same as those of $\text{RoBERTa}_{\text{Large}}$.

3.2. SuperGLUE

To evaluate our pre-trained GLM models, we conduct experiments on the SuperGLUE (Wang et al., 2019) benchmark. SuperGLUE consists of 8 challanging natural language understanding tasks, including question answering (Clark et al., 2019; Khashabi et al., 2018; Zhang et al., 2018), textual entailment (Dagan et al., 2005; Clark et al., 2019), coreference resolution (Levesque et al., 2012), word sense disambiguation (Pilehvar & Camacho-Collados, 2019), and causal reasoning (Roemmele et al., 2011). We adopt the same evaluation metrics as (Wang et al., 2019).

We reformulate each task as a blank filling task with human-crafted cloze questions, using the patterns constructed by PET (Schick & Schütze, 2020a). Then we finetune the pre-trained GLM models on each task as we described in Section 2.4. For finetuning, we also use the AdamW optimizer with peak learning rate $1e{-}5$, warm-up over the first 6% training steps and then a linear decay. For small datasets (COPA, WSC, CB, RTE), we finetune GLM for 20 epochs. For larger datasets, we reduce the number of training epochs (10 for WiC, BoolQ, MultiRC; 5 for ReCoRD) as the model converges earlier.

For fair comparison with GLM $_{\text{Base}}$ and GLM $_{\text{Large}}$, we choose $\text{BERT}_{\text{Base}}$ and $\text{BERT}_{\text{Large}}$ as our baselines, which are pre-trained on the same corpus and for a similar amount of time to our models. We report the performance of standard finetuning (i.e. classification on the [CLS] token representation). The performance of BERT with cloze questions is reported in Section 3.4. To compare with GLM $_{\text{RoBERTa}}$, we choose T5, $\text{BART}_{\text{Large}}$, and $\text{RoBERTa}_{\text{Large}}$ as our baselines. T5 has no direct match in amount of parameters for $\text{BERT}_{\text{Large}}$, so we present the results of both T5_{Base} (220M parameters) and T5_{Large} (770M parameters). All the other baselines are of similar size to $\text{BERT}_{\text{Large}}$.

The results are shown in Table 2. With the same amount of training data (BookCorpus + Wikipeida), GLM consistently outperforms BERT on most of the tasks, with either base or large architecture. The only exception is WiC, which is a word sense disambiguation task. On average, GLM $_{\text{Base}}$ scores 4.6% higher than $\text{BERT}_{\text{Base}}$, and GLM $_{\text{Large}}$ scores 5.0% higher than $\text{BERT}_{\text{Large}}$. It clearly demonstrates the advantage of our method in NLU tasks. In the setting of $\text{RoBERTa}_{\text{Large}}$, GLM $_{\text{RoBERTa}}$ can still achieve improvements over the baselines, but with a smaller margin. T5_{Large} performs the best among baselines, but of two times the size of $\text{BERT}_{\text{Large}}$. We also find that BART does not perform well on the challenging SuperGLUE benchmark. We conjecture this can be attributed to the low parameter efficiency of the encoder-decoder architecture and the denoising seq2seq objective.

3.3. Multi-Task Pre-training

Then we train a GLM $_{\text{Large}}$ and two slightly larger models with multi-task pre-training. In the multi-task pre-training setting, as described in Section 2.3, within one training batch, 50% of the time we sample the BERT-style spans and 50% of the time we sample the generation spans. We evaluate the multi-task model for NLU, seq2seq and zero-shot language modeling.

Table 3. Results on Gigaword abstractive summarization

Model	RG-1	RG-2	RG-L
MASS	37.7	18.5	34.9
$\text{UniLM}_{\text{Large}}$	38.5	19.5	35.8
$\text{GLM}_{\text{Large}}$	38.6	19.7	36.0
$\text{GLM}_{\text{Large}}$ (multi-task)	38.5	19.4	35.8
GLM_{410M} (multi-task)	**38.9**	**20.0**	**36.2**

3.3.1. SuperGLUE

For NLU tasks, we still evaluate models on the SuperGLUE benchmark. The results are also shown in Table 2. We can observe that $\text{GLM}_{\text{Large}}$ with multi-task pre-training performs slightly worse than $\text{GLM}_{\text{Large}}$, but still outperforms $\text{BERT}_{\text{Large}}$ and $\text{UniLM}_{\text{Large}}$. Increasing multi-task GLM's parameters to 410M ($1.25\times\text{BERT}_{\text{Large}}$) leads to performance better than $\text{GLM}_{\text{Large}}$. GLM with 515M parameters ($1.5\times\text{BERT}_{\text{Large}}$) can perform even better.

3.3.2. Seq2Seq

We use abstractive summarization, which aims to produce a concise and fluent summary conveying the key information in the input text, as the evaluation task for seq2seq. We use the Gigaword dataset (Rush et al., 2015) for model fine-tuning and evaluation. We fine-tune $\text{GLM}_{\text{LARGE}}$ on the training set for 10 epochs with AdamW optimizer. The learning rate has a peak value of $3e-5$, warm-up over the 6% training steps and a linear decay. We also use label smoothing with rate 0.1 (Pereyra et al., 2017). The maximum document length is 192 and the maximum summary length is 32. During decoding, we use beam search with beam size of 5 and remove repeated trigrams. We tweak the value of length penalty on the development set. The results are shown in Table 3. We can observe that $\text{GLM}_{\text{Large}}$ can achieve performance matching or better than seq2seq and unified pre-training models. $\text{GLM}_{\text{Large}}$ with multi-task pre-training performs slightly worse than $\text{GLM}_{\text{Large}}$. This indicates that the generative objective, which teaches the model to extend the given contexts, is not helpful to summarization, which aims to extract useful information from the context. Increasing multi-task GLM's parameters to 410M leads to the best performance on the task.

3.3.3. Language Modeling

Most language modeling datasets are based on Wikipedia documents, which our pre-training dataset already contains. For fair comparison with BERT, we cannot remove Wikipedia from the pre-training dataset. Therefore, we evaluate the perplexity of language modeling on the test set of our pre-training dataset, which contains 20MB texts and is denoted as BookWiki. We also evaluate on the LAMBADA dataset (Paperno et al., 2016), which tests the ability of systems to model long-range dependencies in text. The task is to predict the final word of a passage. As the baseline, we train a $\text{GPT}_{\text{Large}}$ model (Radford et al., 2018b; Brown et al., 2020) with the same data and tokenization as $\text{GLM}_{\text{Large}}$.

Table 4. Zero-shot language modeling results.

Model	Lambada (Accuracy)	BookWiki (Perplexity)
$\text{GLM}_{\text{Large}}$ (uni)	0.0	> 100
$\text{GLM}_{\text{Large}}$ (multi-task,uni)	47.4	15.1
– 2d positional encoding	45.8	15.1
GLM_{410M} (multi-task,uni)	49.5	14.5
GLM_{515M} (multi-task,uni)	**50.4**	**13.9**
$\text{GLM}_{\text{Large}}$ (bi)	10.6	> 100
$\text{GLM}_{\text{Large}}$ (multi-task,bi)	48.5	14.9
– 2d positional encoding	47.3	15.0
GLM_{410M} (multi-task,bi)	**53.5**	**14.3**
GLM_{515M} (multi-task,bi)	**54.9**	**13.7**
$\text{GPT}_{\text{Large}}$ (uni)	50.1	14.4

The results are shown in Table 4. All the models are evaluated in the zero-shot setting. Since GLM also learns the bidirectional attention, we also evaluate GLM under the setting in which the contexts are encoded with bidirectional attention, denoted with "bi" in the table. Without generative objective during pre-training, $\text{GLM}_{\text{Large}}$ cannot complete the language modeling tasks. With the same amount of parameters, $\text{GLM}_{\text{Large}}$ with multi-task pretraining performs worse than $\text{GPT}_{\text{Large}}$. This is expected since $\text{GLM}_{\text{Large}}$ also optimizes the BERT-style objective. Increasing GLM's parameters to 410M ($1.25\times$ $\text{GPT}_{\text{Large}}$) leads to performance close to $\text{GPT}_{\text{Large}}$. GLM with 515M parameters ($1.5\times$ $\text{GPT}_{\text{Large}}$) can further outperform $\text{GPT}_{\text{Large}}$. With the same amount of parameters, encoding the context with directional attention can improve the performance of language modeling. This is the advantage of GLM over unidirectional GPT.

Above all, we can conclude that GLM can effectively share model parameters across different tasks, achieving better performance than a standalone BERT or GPT with fewer than two times of its parameters.

3.4. Ablation Study

We perform an ablation study to understand the importance of the architecture improvements and the formulation of classification tasks as blank filling. On SuperGLUE, we evaluate GLM finetuned as sequence classifiers and BERT with cloze-style finetuning, and compare the performance to GLM with cloze-style GLM finetuning in Table 5. Com-

Table 5. Ablation study on the SuperGLUE dev set.

Model	ReCoRD F1/Acc.	COPA Acc.	WSC Acc.	RTE Acc.	BoolQ Acc.	WiC Acc.	CB F1/Acc.	MultiRC F1a/EM	Avg
$\text{BERT}_{\text{Base}}$ (cloze)	61.3/60.6	70.0	62.5	69.0	73.7	**66.8**	74.9/82.1	63.8/16.3	65.2
GLM $_{\text{Base}}$ (classifier)	35.2/34.4	58.0	63.5	61.4	74.9	58.3	**94.8/92.9**	46.3/ 5.4	58.8
GLM $_{\text{Base}}$	**73.5/72.8**	**71.0**	**72.1**	**71.2**	**77.0**	64.7	89.5/85.7	**72.1/26.1**	**70.7**
$\text{BERT}_{\text{Large}}$ (cloze)	70.0/69.4	**80.0**	76.0	72.6	78.1	**70.5**	93.5/91.1	70.0/23.1	73.2
GLM $_{\text{Large}}$ (classifier)	81.3/80.6	62.0	63.5	66.8	80.5	65.0	89.2/91.1	72.3/27.9	70.0
GLM $_{\text{Large}}$	**81.7/81.1**	76.0	**81.7**	**74.0**	**82.1**	68.5	**96.1/94.6**	**77.1/36.3**	**77.0**
– shuffle spans	82.0/81.4	61.0	79.8	54.5	65.8	56.3	90.5/92.9	76.7/37.6	68.5
+ sentinel tokens	81.8/81.3	69.0	78.8	77.3	81.2	68.0	93.7/94.6	77.5/37.7	76.0

pared to BERT with cloze-style finetuning, GLM benefits from the architecture improvement. Especially on ReCoRD and WSC, where the verbalizer consists of multiple tokens, GLM can consistently outperform BERT. This demonstrates GLM's advantage on handling variable-length blank. From the results, we can observe that the cloze formulation is critical for GLM's performance on NLU tasks. Especially for the base model, classifier finetuning leads to drop of performance by more than 10 points. For the large model, cloze-style finetuning can improve the performance by 7 points. The impact of the finetuning method varies with model sizes and datasets. With classifier finetuning, the base model fails to converge on the ReCoRD dataset, but the large model can achieve performance close to that of cloze-style finetuning. On WSC, classifier finetuning leads to the naive prediction of the majority class for both the base and the large models. Overall, the cloze-style finetuning can improve the performance of GLM on almost all the datasets.

We also compare GLM variants with different pretraining designs to understand their importance. The row 7 of Table 5 shows that removing the span shuffling (always predicting the masked spans from left to right) leads to a severe performance drop on SuperGLUE. The model in row 8 uses different sentinel tokens instead of the same [MASK] token to represent different masked spans. The model performs worse than the standard GLM, since it pays more modeling capacity to learn the sentinel tokens and transfers less to downstream tasks with only one blank. In Table 4, we show that removing the second dimension of 2d positional encoding hurts the performance on language modeling.

4. Related Work

Pre-trained Language Models In NLP, self-supervised learning has long been used to learn word vectors as inputs to neural networks (Mikolov et al., 2013; Pennington et al., 2014). Recently, pre-training large-scale language models with self-supervised learning on abundant web texts significantly improves the performance on downstream tasks. There are three types of pre-trained language models. The first type is the autoencoding model, which learns a bidirectional contextualized encoder for natural language understanding via denoising objectives. BERT (Devlin et al., 2019) pre-trains a large transformer model (Vaswani et al., 2017) via masked language modeling to obtain contextualized word representations. SpanBERT (Joshi et al., 2020) masks continuous spans of tokens for improved span representations. The second type is the autoregressive model, which learns an left-to-right language model for text generation. GPT (Radford et al., 2018a) shows that the representations learned by generative pre-training can also improve language understanding. XLNet (Yang et al., 2019) generalizes the autoregressive model with permutation language modeling to learn bidirectional attention for language understanding tasks. The third type is the encoder-decoder model pre-trained for seq2seq tasks. MASS (Song et al., 2019) maps an input text with continuous spans masked to the masked tokens. BART (Lewis et al., 2019) applies various transformations, including masking, deletion, and permutation, and recovers the original text with the decoder. PALM (Bi et al., 2020) is pre-trained for generating coherent text from given context and adds a BERT-based autoencoding objective to the encoder.

NLU as Generation Tasks Previously, pre-trained language models complete classification tasks for NLU with linear classifiers on the learned representations. GPT-2 (Radford et al., 2018b) shows that generative language models can learn to complete understanding tasks such as question answering and reading comprehension by directly predicting the correct answers, even without any explicit supervision. GPT-3 (Brown et al., 2020) further proves that language models can achieve strong performance on NLU tasks in the few-shot learning setting by adding a few labeled examples in the context. However, generative models requires much more parameters to work due to the limit of unidirectional attention. T5 (Raffel et al., 2020) formulates most language tasks in the text-to-text framework, but requires more parameters to outperform BRET-based models such as RoBERTa (Liu et al., 2019).

Recently, PET (Schick & Schütze, 2020b;a) proposes to reformulate input examples as cloze questions with patterns similar to the pre-training corpus in the few-shot setting. It has been shown that combined with gradient-based fine-tuning on ALBERT, PET can achieve better performance in the few-shot setting than GPT-3, while requiring only 0.1% of its parameters. Athiwaratkun et al. (2020); Paolini et al. (2021) propose augmented natural language for structured prediction tasks such as sequence tagging and relation extraction. Donahue et al. (2020) and Shen et al. (2020) also study blank infilling language models. Different from their work, we pre-train language models by blank infilling and evaluate its performance in downstream NLU and generation tasks.

5. Conclusions

GLM is a general pre-training framework for natural language understanding, generation and seq2seq. We show that the NLU tasks can be formulated as conditional generation tasks, and therefore solvable by autoregressive models. GLM unifies the pre-training objectives for different tasks as autoregressive blank filling, with mixed attention mask and the novel 2D position encodings. Empirically we show that GLM outperforms previous methods for NLU tasks and can effectively share parameters for different tasks. In the future, we hope to scale GLM to larger transformer models and more pre-training data, and examine its performance in more settings such as knowledge probing and few-shot learning.

References

Athiwaratkun, B., dos Santos, C., Krone, J., and Xiang, B. Augmented natural language for generative sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 375–385, 2020.

Bao, H., Dong, L., Wei, F., Wang, W., Yang, N., Liu, X., Wang, Y., Gao, J., Piao, S., Zhou, M., and Hon, H. Unilmv2: Pseudo-masked language models for unified language model pre-training. In *ICML 2020*, volume 119, pp. 642–652, 2020.

Bi, B., Li, C., Wu, C., Yan, M., Wang, W., Huang, S., Huang, F., and Si, L. PALM: Pre-training an Autoencoding&Autoregressive Language Model for Context-conditioned Generation. In *EMNLP 2020*, pp. 8681–8691, 2020.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language Models are Few-Shot Learners. In *NeurIPS 2020*, 2020.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, 2019.

Dagan, I., Glickman, O., and Magnini, B. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pp. 177–190. Springer, 2005.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL 2019*, pp. 4171–4186, 2019.

Donahue, C., Lee, M., and Liang, P. Enabling language models to fill in the blanks. *arXiv preprint arXiv:2005.05339*, 2020.

Dong, L., Yang, N., Wang, W., Wei, F., Liu, X., Wang, Y., Gao, J., Zhou, M., and Hon, H. Unified language model pre-training for natural language understanding and generation. In *NeurIPS 2019*, pp. 13042–13054, 2019.

Gokaslan, A. and Cohen, V. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

Joshi, M., Chen, D., Liu, Y., Weld, D. S., Zettlemoyer, L., and Levy, O. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Trans. Assoc. Comput. Linguistics*, 8:64–77, 2020.

Khashabi, D., Chaturvedi, S., Roth, M., Upadhyay, S., and Roth, D. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 252–262, 2018.

Levesque, H., Davis, E., and Morgenstern, L. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*. Citeseer, 2012.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. BART: Denoising Sequence-to-Sequence Pre-training for

Natural Language Generation, Translation, and Comprehension. In *ACL 2020*, pp. 7871–7880, 2019.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

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

Mackenzie, J., Benham, R., Petri, M., Trippas, J. R., Culpepper, J. S., and Moffat, A. CC-News-En: A Large English News Corpus. In *CIKM 2020*, pp. 3077–3084, 2020.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. Distributed representations of words and phrases and their compositionality. In *NIPS 2013*, pp. 3111–3119, 2013.

Paolini, G., Athiwaratkun, B., Krone, J., Ma, J., Achille, A., Anubhai, R., Santos, C. N. d., Xiang, B., and Soatto, S. Structured prediction as translation between augmented natural languages. *arXiv preprint arXiv:2101.05779*, 2021.

Paperno, D., Kruszewski, G., Lazaridou, A., Pham, Q. N., Bernardi, R., Pezzelle, S., Baroni, M., Boleda, G., and Fernández, R. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *ACL 2016*, 2016.

Pennington, J., Socher, R., and Manning, C. Glove: Global Vectors for Word Representation. In *EMNLP 2014*, pp. 1532–1543, 2014.

Pereyra, G., Tucker, G., Chorowski, J., Kaiser, L., and Hinton, G. E. Regularizing neural networks by penalizing confident output distributions. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, 2017.

Pilehvar, M. T. and Camacho-Collados, J. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 1267–1273, 2019.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving Language Understanding by Generative Pre-Training. 2018a.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2018b.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

Roemmele, M., Bejan, C. A., and Gordon, A. S. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, pp. 90–95, 2011.

Rush, A. M., Chopra, S., and Weston, J. A neural attention model for abstractive sentence summarization. In *EMNLP 2015*, pp. 379–389, 2015.

Schick, T. and Schütze, H. It's not just size that matters: Small language models are also few-shot learners. *CoRR*, abs/2009.07118, 2020a.

Schick, T. and Schütze, H. Exploiting cloze questions for few-shot text classification and natural language inference. *CoRR*, abs/2001.07676, 2020b.

Shen, T., Quach, V., Barzilay, R., and Jaakkola, T. Blank language models. *arXiv preprint arXiv:2002.03079*, 2020.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., and Catanzaro, B. Megatron-lm: Training multi-billion parameter language models using model parallelism. *CoRR*, abs/1909.08053, 2019.

Song, K., Tan, X., Qin, T., Lu, J., and Liu, T.-Y. MASS: Masked Sequence to Sequence Pre-training for Language Generation. In *ICML 2019*, volume 97, pp. 5926–5936, 2019.

Trinh, T. H. and Le, Q. V. A Simple Method for Commonsense Reasoning. *arXiv:1806.02847 [cs]*, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *NIPS 2017*, pp. 5999–6009, 2017.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *NeurIPS 2019*, pp. 3261–3275, 2019.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *NeurIPS 2019*, pp. 5754–5764, 2019.

Zhang, S., Liu, X., Liu, J., Gao, J., Duh, K., and Van Durme, B. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*, 2018.

Zhu, Y., Kiros, R., Zemel, R. S., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV 2015*, pp. 19–27, 2015.

A. Pretraining Setting

A.1. Dataset

To train GLM $_{\text{Base}}$ and GLM $_{\text{Large}}$, we use BookCorpus (Zhu et al., 2015) + Wikipedia used by BERT (Devlin et al., 2019).

To train GLM $_{\text{RoBERTa}}$, we follow the pre-training datasets of RoBERTa (Liu et al., 2019), which consist of BookCorups (Zhu et al., 2015) + Wikipedia (16GB), CC-News (the English portion of the CommonCrawl News dataset[2] 76GB), OpenWebText (web content extracted from URLs shared on Reddit with at least three upvotes(Gokaslan & Cohen, 2019), 38GB) and Stories (subset of CommonCrawl data filtered to match the story-like style of Winograd schemas (Trinh & Le, 2019), 31GB). The Stories dataset is already unavailable[3]. Therefore, we remove the Stories dataset and replace OpenWebText with OpenWebText2[4] (66GB). The CC-News dataset is not publicly available and we use the CC-News-en published by (Mackenzie et al., 2020). All the datasets used total 158GB of uncompressed texts, close in size to RoBERTa's 160GB datasets.

A.2. Hyperparameters

To train GLM $_{\text{RoBERTa}}$, we follow most of the hyperparameters of RoBERTa. The main difference includes: (1) Due to resource limit, we only pre-train GLM $_{\text{RoBERTa}}$ for 250,000 steps, which are half of RoBERTa and BART's training steps, and close to T5 in number of trained tokens. (2) We use cosine decay instead of linear decay for learning rate scheduling (3) We additionally apply gradient clipping with value 1.0.

The hyperparameters for GLM $_{\text{Base}}$ and GLM $_{\text{Large}}$ are similar to those used by GLM $_{\text{RoBERTa}}$. For trade-off of training speed and fair comparison with BERT (batch size 256 and 1,000,000 training steps), we use batch size of 1024 and 200,000 training steps for GLM $_{\text{Large}}$. Since GLM $_{\text{Base}}$ is smaller, we reduce the number of training steps to 120,000 to speed up pre-training.

The hyperparameters for all the pre-training settings are summarized in Table 6.

[2]https://commoncrawl.org/2016/10/news-dataset-available

[3]https://github.com/tensorflow/models/tree/archive/research/lm_commonsense#1-download-data-files

[4]https://openwebtext2.readthedocs.io/en/latest

Table 6. Hyperparameters for pretraining

Hyperparameters	GLM $_{\text{Base}}$	GLM $_{\text{Large}}$	GLM $_{\text{RoBERTa}}$
Number of Layers	12	24	24
Hidden size	768	1024	1024
FFN inner hidden size	3072	4096	4096
Attention heads	12	16	16
Attention head size	64	64	64
Dropout	0.1	0.1	0.1
Attention Dropout	0.1	0.1	0.1
Warmup Steps	6k	8k	30K
Peak Learning Rate	4e-4	2e-4	4e-4
Batch Size	1024	1024	8192
Weight Decay	0.1	0.1	0.01
Max Steps	120k	200k	250k
Learning Rate Decay	Cosine	Cosine	Cosine
Adam ϵ	1e-6	1e-6	1e-6
Adam β_1	0.9	0.9	0.9
Adam β_2	0.98	0.98	0.98
Gradient Clipping	1.0	1.0	1.0

B. Downstream Tasks

B.1. SuperGLUE

The SuperGLUE benchmark consists of 8 NLU tasks. We formulate them as blank infilling tasks, following (Schick & Schütze, 2020a). Table 7 shows the cloze questions and verbalizers we used in our experiments. For 3 tasks (ReCoRD, COPA, and WSC), the answer may consist of multiple tokens, and for the other 5 tasks, the answer is always a single token.

When finetuning GLM on the SuperGLUE tasks, we construct the input using the cloze questions in Table 7 and replace the blank with a [MASK] token. Then we compute the score of generating each answer candidate. For the 5 single-token tasks, the score is defined to be the logit of the verbalizer token. For the 3 multi-token tasks, we use the sum of the log-probabilities of the verbalizer tokens. Thanks to the autoregressive blank infilling mechanism we proposed, we can obtain all the log-probabilities in one pass. Then we compute the cross entropy loss using the groundtruth label and update the model parameters.

For the baseline classifiers, we follow the standard practice to concatenate the input parts of each task (such as the premise and hypothesis for textual entailment, or the passage, question and answer for ReCORD and MultiRC) and add a classification layer on top of the [CLS] token representation. We also implemented cloze-style finetuning for the other pre-trained models, but the performance was usually similar to the standard classifier, as we shown in the ablation study. Models with blank-infilling objectives, such as T5 and our GLM, benifits more from converting the NLU tasks into cloze questions. Thus for T5 and GLM, we report the performance after such convertion in our main results. For $\text{BERT}_{\text{Large}}$, $\text{RoBERTa}_{\text{Large}}$, and $\text{GLM}_{\text{RoBERTa}}$, we search the best hyperparameters on the dev set. The ranges of the hyperparameters for the grid search are: learning rate in $\{5e-6, 1e-5, 2e-5\}$ and batch size in $16, 32$.

B.2. Seq2Seq

We use the abstractive summarization dataset Gigaword (Rush et al., 2015) for model fine-tuning and evaluation. We fine-tune GLM $_{\text{LARGE}}$ on the training set for 10 epochs with AdamW optimizer. The learning rate has a peak value of $3e-5$, warm-up over the 6% training steps and a linear decay. We also use label smoothing with rate 0.1 (Pereyra et al., 2017). The maximum document length is 192 and the maximum summary length is 32. During decoding, we use beam search with beam size of 5 and remove repeated trigrams. We tweak the value of length penalty on the development set. The evaluation metrics are the F1 scores of Rouge-1, Rouge-2, and Rouge-L on the test set.

B.3. Language Modeling

We evaluate the model's ability of language modeling with perplexity on BookWiki and accuracy on the LAMBDA dataset (Paperno et al., 2016).

Perplexity is an evaluation criterion that has been well studied for language modeling. Perplexity is the exponentiation

Dataset	Task	Cloze Question	Verbalizers
ReCoRD	Question answering	[passage p] [cloze question q]	Anwer candidates
COPA	Causal reasoning	"[choice c_1]" or "[choice c_2]"? [premise p], so ___.	c_1 / c_2
WSC	Coreference resolution	[sentence s] The pronoun '$*p*$' refers to ___.	Noun n
RTE	Texual entailment	"[hypothesis h]"? | ___, "[premise p]"	"yes" (entailment), "no" (not entailment)
BoolQ	Question anwering	[passage p]. Question: q? Answer: ___.	"yes" / "no"
WiC	Word sense disambiguation	"[sentence s_1]" / "[sentence s_2]" Similar sense of [word w]? ___.	"yes" / "no"
CB	Texual entailment	"[hypothesis h]"? | ___, "[premise p]"	"yes" (entailment), "no" (contradiction), "maybe" (neutral)
MultiRC	Question answering	[passage p]. Question: q? Is it [answer a]? ___.	"yes" / "no"

Table 7. Cloze questions and verbalizers for the 8 SuperGLUE tasks used in our experiments.

of the average cross entropy of a corpus.

$$\text{PPL} = \exp(-\frac{1}{T}\sum_{t=1}^{T} p(x_t|\boldsymbol{x}_{<t})) \tag{6}$$

where $\boldsymbol{x}_{<t} = [x_0, \cdots, x_{t-1}]$. Since transformers can only operate on a window of fixed input size w, we cannot fully calculate $p(x_t|\boldsymbol{x}_{<t})$ and can only calculate $p(x_t|\boldsymbol{x}_{t-w:t-1})$. Even calculating this value for each token is prohibitively expensive, since we need to conduct T evaluations of w-size contexts. To improve evaluation efficiency, we adopt *overlapping evaluation*, where we advance the sliding windows by some overlap o each time and only compute the cross entropy loss for the last o tokens of the window. In our experiments we set $o = 256$ for all the models.

LAMBDA is a cloze-style dataset to test the ability of long-range dependency modeling. Each example is a passage consisting of 4-5 sentences with the last word missing and the model is required to predict the last word of the passage. Since we use WordPiece tokenization, a word can be split into several subword units. We use teacher forcing and consider the prediction correct only when all the predicted tokens are correct.

C. Text Generation Samples

We show texts generated by GLM $_{\text{Large}}$ (multi-task) given unseen contexts randomly sampled from the test set. We use top-k random sampling with $k = 40$ for generation and set maximum sequence length to 512. Some of the texts are cut short.

Example 1 ***Context***: *The Wyoming State Legislature is the legislative branch of the U.S. State of Wyoming .*

GLM: *It consists of a state senate and a house of representatives. As of 2019, state senators are Steve Kish, Joe L. Williams, and Linda S. Bollens. Members of the Wyoming State Legislature are elected from single-member districts representing the majority of the state. The current state senate members are: In recent years, there have been four changes to the senate. The most recent is the creation of a six-seat district that includes all or part of the following: In the 2009 elections, the state senate members were elected to six-year terms. The current state house members are: The Wyoming Constitution assigns certain powers to the governor. Most notably, the governor is president of the senate and governor. However, if the governor desires to appoint a member to the Wyoming state senate, a law authorizes the governor to do so. The governor of Wyoming holds no legislative power but has the power to veto lawmakers, which is not limited to the veto of laws. Under the wyoming state constitution, the governor can veto the actions of the other members of the wyoming house of representatives. The governor can also appoint members of the wyoming senate. In addition, the governor can appoint members of the Wyoming house of representatives. Wyoming's constitution provides that the governor can appoint a member of the wyoming state senate to the wyoming supreme court, and the chairman of the wyoming senate.*

Example 2 ***Context***: *Jonathan Terry is a television and film actor.*

GLM: *He was a voice actor for the "X-Men" cartoon series. He was also a voice actor for "the Simpsons" as well as "the marvelous misadventures of superman. He also has voiced characters in "the legend of korra" He has appeared on several television series, including "the simpsons", "the x-files", "heroes" and "the simpsons movie" as the character captain billy higgledy-pig, and the web series "krusty mysteries" as the character Colonel Trungus. He wrote for "the Guardian" newspaper. Jonathan Terry was born in London. Terry grew up in Surrey, England and attended* *the University of Sussex in the United Kingdom, graduating with a degree in english literature. He was a guest lecturer at King's College London, and then took two years of acting courses at the brit school of acting to prepare for his future career in the entertainment industry. Terry first appeared in the TV series "the Simpsons" as the character captain Billy Higgledy-pig, but his character was only a one-time recurring character in the series' first six seasons. He later appeared as a regular for the show's final six seasons, and has been a frequent guest in the show since. He appeared in the first few episodes of "" as the character major Jack Ryan. He has also appeared as part of the supporting cast of several episodes of "the secret life of pets". He has also worked on "the simpsons" TV show since "the simpsons movie", most notably playing the roles of Captain Skeletor and the ghost of the same name. He plays characters in several films, including "", "", "" and "". He has appeared in music videos for the killers in 1993, the pretenders in 1995, and in the TV shows "the royal" and "the bill".*

Example 3 ***Context***: *Corona was a station along the port Washington branch of the long island rail road in the Corona section of queens, New York City. It was one of two stations built by the flushing railroad in Corona, this one having been at Grand Avenue (later called National Avenue, now National Street) and 45th Avenue.*

GLM: *It had a brick entrance building next to the tracks, and one platform with 2 side platforms. The station burned down and was rebuilt in 1908. The station closed on september 15, 1927, with the train service transferred from Grand Avenue to 45th Avenue. There were plans to build a subway extension to Corona, but it was never built. In the 1980s, two stations were constructed on the line, Corona Road and Corona Park. The new Corona station opened in 1988, and the original Corona station was demolished.*

Example 4 ***Context***: *Robert Lee Smith (born July 5, 1938) is a former American football cornerback in the national football league. He played for the Los Angeles Rams (1962–1965) and the Detroit Lions (1965–1966). He was also a kick and punt returner.*

GLM: *In his four-year NFL career, he played in 33 games and started 14, registering 62 career interceptions. He completed his NFL career with five interceptions. Smith was born in La Canada Flintridge, Michigan, in 1938. He attended Bishop O' Dowd high school in Flintridge. Smith' s father was a pharmacist. He graduated from Michigan State University in 1958 with a degree in business administration. Smith played for the Michigan Wolverines football team from 1959 to 1963. In his two years as a reserve cornerback, he led the conference in interceptions with five. In 1962, he set the Wolverines' all-time interception record with 13, and was second overall in the 1962 season's Heisman Trophy voting. Smith also won the Dick Butkus award as the nation's outstanding linebacker. In 1961, the "Los Angeles Times" wrote that Smith "is an outstanding pass rusher, with an average of almost 100 yards per punt return." Smith was inducted into the university of Michigan athletic hall of honor in 1989 and the national football foundation hall of fame in 1991. He was elected to the Michigan sports hall of fame in 1995. Smith earned the honor because of his accomplishments prior to his NFL career. He was one of four Michigan players honored as first-overall selections in the 1964 NFL draft. The others were Joe Namath, Bill Nelsen, and Jerry Kramer. In 1966, the NFL gave players $300,000 a season to play football. After his rookie season, he was not selected to play in the 1966 pro bowl. On January 13, 1966, the Rams traded smith to the Detroit Lions for Paul Hornung, and later that year he was traded to the Lions for Ray "the Lion" Jones in exchange for Linebacker Jim "the Hawk" Johnson. On September 10, 1968, he was traded back to Los Angeles for a second round pick in the 1970 draft. He was also traded to the St. Louis Cardinals for a second round pick in the 1970 draft. On June 2, 1970 he was cut by the Cardinals. On November 15, 1970, the Los Angeles Rams acquired Smith from the Lions in exchange for Linebacker Tony Harris. The Rams waived Smith during the September 1, 1972 offseason. Smith's number at Michigan State was # 7 in 1969*