On the Influence of Masking Policies in Intermediate Pre-training

Qinyuan Ye[1†] Belinda Z. Li[2‡] Sinong Wang[3] Benjamin Bolte[3]
Hao Ma[3] Wen-tau Yih[3] Xiang Ren[1] Madian Khabsa[3]
[1]University of Southern California [2]MIT CSAIL [3]Facebook AI
{qinyuany,xiangren}@usc.edu bzl@mit.edu
{sinongwang,bbolte,scottyih,haom,mkhabsa}@facebook.com

Abstract

Current NLP models are predominantly trained through a two-stage "pre-train then fine-tune" pipeline. Prior work has shown that inserting an *intermediate* pre-training stage, using heuristic masking policies for masked language modeling (MLM), can significantly improve final performance. However, it is still unclear (1) *in what cases* such intermediate pre-training is helpful, (2) whether hand-crafted heuristic objectives are *optimal* for a given task, and (3) whether a masking policy designed for one task is *generalizable* beyond that task. In this paper, we perform a large-scale empirical study to investigate the effect of various masking policies in intermediate pre-training with nine selected tasks across three categories. Crucially, we introduce methods to *automate* the discovery of optimal masking policies via direct supervision or meta-learning. We conclude that the success of intermediate pre-training is dependent on appropriate pre-train corpus, selection of output format (*i.e.*, masked spans or full sentence), and clear understanding of the role that MLM plays for the downstream task. In addition, we find our learned masking policies outperform the heuristic of masking named entities on TriviaQA, and policies learned from one task can positively transfer to other tasks in certain cases, inviting future research in this direction.

1 Introduction

Large, neural language models (LMs) pre-trained with masked language modeling (Devlin et al., 2019; Raffel et al., 2020) have achieved impressive results over a variety of NLP tasks. Studies show that an additional intermediate pre-training stage between general pre-training and task-specific fine-tuning further improves downstream performance (Fig. 1). For example, intermediate pre-training by masking and recovering named entities or dates, known as salient span masking (SSM, Guu et al. 2020), significantly improves a model's performance of answering factoid questions in a closed-book setting (Roberts et al., 2020). However, there is a lack of systematic study on how intermediate pre-training works, whether heuristic masking policies like SSM are near-optimal, or whether they generalize to different NLP tasks. Additionally, it is unclear that for tasks other than closed-book QA, whether intermediate pre-training is helpful, or what masking strategy should be adopted.

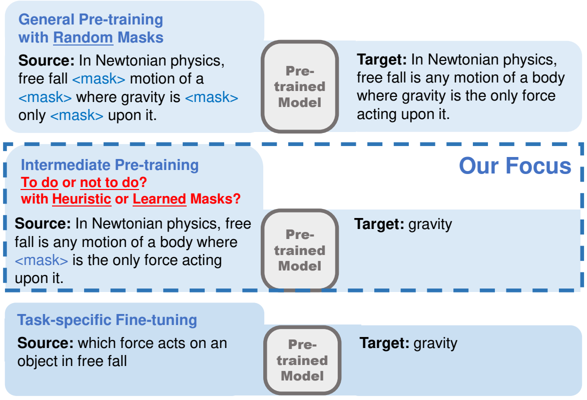

General Pre-training
with Random Masks
Source: In Newtonian physics, free fall <mask> motion of a <mask> where gravity is <mask> only <mask> upon it.
Pre-trained Model
Target: In Newtonian physics, free fall is any motion of a body where gravity is the only force acting upon it.
Intermediate Pre-training
To do or not to do?
with Heuristic or Learned Masks?
Our Focus
Source: In Newtonian physics, free fall is any motion of a body where <mask> is the only force acting upon it.
Pre-trained Model
Target: gravity
Task-specific Fine-tuning
Source: which force acts on an object in free fall
Pre-trained Model
Target: gravity

Figure 1: **Analysis Setup.** We investigate the influence brought by different masking policies during intermediate pre-training, a stage between general pre-training and task-specific fine-tuning. We apply three types of policies (heuristic, supervised, meta-learned) on three categories of tasks (closed-book QA, knowledge-intensive language tasks, multiple-choice QA).

In this paper, we offer a large-scale, systematic study on the effects and transferability of masking strategies during intermediate pre-training, while we carefully control all other aspects (§3). We first begin our analysis with a focus on three **heuristic masking policies** (§4.1). We fine-tune the models resulting from intermediate pre-training on nine selected tasks covering three categories (closed-book QA, knowledge-intensive language tasks, and

[†] Work partially done while interning at Facebook AI.
[‡] Work partially done while working at Facebook AI.

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 7190–7202
November 7–11, 2021. ©2021 Association for Computational Linguistics

multi-choice QA). Our results suggest that successful intermediate pre-training is dependent on the selection of appropriate corpus. Moreover, heuristic-based approaches are effective only when we have a precise understanding of the role masked language modeling (MLM) plays in downstream task. For example, MLM serves as a sort of memorization step (Petroni et al., 2019), whereby learning to unmask spans in context is analogous to memorizing facts about the span. In the absence of such understanding, heuristic policies may be sub-optimal.

This motivates us to explore whether automating the discovery of optimal masking policies is possible. We design methods to **learn a masking policy** with supervised learning (§4.2) or meta-learning (§4.3), and compare downstream task performance using the same protocol in our previous analysis. Notably, we observe that masking policies learned with supervised learning and meta-learning outperforms the SSM policy for TriviaQA (Joshi et al., 2017), and these policies learned from TriviaQA also help improve performance on Web Questions (Berant et al., 2013). We also discuss the pros and cons of learned masking policies, such as downstream task learning efficiency, risks of over-fitting and learning instability.

Finally, in hopes to better understand the heuristic and learned masking policies, we provide **quantitative analysis** on the masks produced by these policies. We visualize the distribution of part-of-speech tags among masked tokens, and their relation to token frequency in the corpus (§5.3). We find that the masking policies learned from TriviaQA tend to mask more proper nouns and tend to mask less frequent words when compared to SSM.

Overall, our empirical analysis provides useful suggestions for NLP researchers who aim to improve downstream task performance using intermediate pre-training and heuristic masking strategies. In addition, our experiments reveal that infusing task-specific knowledge into LMs with learned masking policies is a promising way to improve downstream task performance, and invite future research in this direction.

2 Preliminary: Masked Language Modeling

In this section, we revisit MLM objective with the notation that we will use throughout the paper. MLM is a predominant pre-training objective for large-scale transformers in NLP (Devlin et al., 2019). MLM and its variants can be characterized with two key components: a **masking policy** $g(.;\phi)$, parameterized by ϕ, which decides the collection of tokens to be masked, and a **language model** $f(.;\theta)$, parameterized by θ.

Formally, given a sequence of tokens $\mathbf{x} = [x_1, x_2, ..., x_m]$, $g(\mathbf{x};\phi)$ generates a sequence of binary decisions $\mathbf{d} = [d_1, d_2, ..., d_m]$, where $d_i = 1$ indicates the token x_i will be masked. The source sequence for pre-training, $\mathbf{x}^{(\text{src})}$, is formulated by replacing the selected tokens with a special `<mask>` token, i.e., $\mathbf{x}^{(\text{src})} = [x_1^{(\text{src})}, x_2^{(\text{src})}, ..., x_m^{(\text{src})}]$, where $x_i^{(\text{src})} = x_i$ if $d_i = 0$ and $x_i^{(\text{src})} =$ `<mask>` if $d_i = 1$. We denote this operation as $\mathbf{x}^{(\text{src})} = \mathbf{x} \oplus \mathbf{d}$. The target sequence $\mathbf{x}^{(\text{tar})}$ can be either the full original sequence $\mathbf{x}$ (BART, Lewis et al. 2020), or the sequence of masked tokens (T5, Raffel et al. 2020).

3 Analysis Setup

In this section we introduce the analysis pipeline (§3.1) and downstream datasets we use (§3.2). We defer the details of learned masking policies to §4.

3.1 Experiment Procedure

Our goal is to analyze the influence in downstream task performance brought by different masking policies $g(.;\phi)$ during intermediate pre-training. Towards this goal, we ensure that the only variable is the masking policy, while all other aspects are controlled, so that the downstream performance reveal the influence we aim to study. We first initialize with a BART-base model (Lewis et al., 2020); then for each masking policy, we conduct experiments following a two-stage pipeline:

Stage 1. Intermediate Pre-training. We perform intermediate pre-training with a given masking policy $g(.;\phi)$. All intermediate pre-training is done with input sequence length of 128, batch size of 2048, learning rate of 0.0001, up to a total number of $100,000$ updates, using Wikipedia snapshot from December 20, 2018[1].

Stage 2. Task-specific Fine-tuning. We fine-tune each resulting checkpoint from Stage 1 on related downstream tasks, and evaluate their performance. We follow the same routine of hyperparameter search for each checkpoint. We then run the

[1] The snapshot is available at https://archive.org/download/enwiki-20181220. Wikipedia is licensed under CC BY-SA 3.0.

fine-tuning experiments with the best hyperparameter setting and three different random seeds. See Appendix B for details.

3.2 Downstream Tasks and Datasets

We focus our study on nine downstream tasks across three categories. We introduce their details and explain the rationale behind our selection in the following.

Closed-book QA. Closed-book QA is a task that requires a language model to directly answer questions without access to external knowledge (Roberts et al., 2020). This paradigm assumes that the model memorizes large amounts of knowledge from its pre-training data, which gets "packed" into its parameters, and can subsequently be "retrieved" to answer questions. Notably, Roberts et al. (2020) reported 9%+ improvement in exact match on TriviaQA when intermediate pre-training with salient span masking (*i.e.*, masking and recovering named entities or dates) is performed on a T5-11B model. This observation inspired our work. Our study considers three datasets for closed-book QA: Natural Questions (**NQ**, Kwiatkowski et al. 2019), WebQuestions (**WQ**) and TriviaQA (**TQA**).

Knowledge-Intensive Tasks from KILT. Extending from closed-book QA, we select three tasks from the KILT benchmark (Petroni et al., 2020) that also aims to test a model's implicit knowledge capacity, while having different task formats and goals. Aidayago2 (**AY2**, Hoffart et al. 2011) is an entity linking task that requires the model to assign a Wikipedia page to an entity mention in the text. The output is the unique name of the Wikipedia page in text format. Zero-shot relation extraction (**ZSRE**, Levy et al. 2017) is a slot filling task that aims to predict the object when given the subject and the relation. The relations in the train/dev/test splits are non-overlapping. Wizard of Wikipedia (**WoW**, Dinan et al. 2019) is a dataset of dialogue histories relevant to knowledge in Wikipedia. The model is required to act like a chatbot and generate the response given previous dialogue history.

Knowledge-Intensive Multiple-choice QA. We select three multiple-choice QA datasets, in which the questions can be answered with commonsense/background knowledge without any context, but the dataset provides additional context paragraphs to explicitly state the background knowledge used. We use WIQA (Tandon et al., 2019) which focuses on procedural text, QuaRTz (Tafjord et al., 2019) which focuses on qualitative relationship, and ROPES (Lin et al., 2019) which focuses on causes and effects. We reformat these tasks into sequence-to-sequence format, following UnifiedQA (Khashabi et al., 2020).

To summarize, all tasks above can be treated as sequence-to-sequence tasks, where each example is a source-target pair $(\mathbf{s}, \mathbf{t})$, accompanied with a context paragraph $\mathbf{c}$ provided by the dataset. Details for dataset splitting are in Appendix D.1.

4 Compared Masking Policies

We experiment with three categories of masking policies: heuristic policies, where g is a fixed heuristic function (§4.1); supervised policies, where g is a model whose weights are learned from *direct supervision* on downstream tasks (§4.2); and meta-learned policies, where g is a model whose weights are learned through *meta-learning* on downstream tasks (§4.3).

4.1 Heuristic Policy

We experiment with the following three heuristic masking policies: (1) BART's original denoising objective (**+Orig**); (2) Masking and recovering 15% randomly selected tokens (**+Rand**)[2]; (3) Salient span masking, *i.e.*, masking and recovering one named entity (Roberts et al., 2020; Guu et al., 2020) (**+SSM**).

4.2 Supervised Policy

When students prepare for closed-book exams, they are likely to review and memorize what they perceive as most important in the text book. Such perception is learned from their prior experience of taking closed-book exams. Following this intuition, Ye et al. (2020) proposed to learn a masking policy for closed-book QA tasks to help the model focus on *likely answers* during intermediate pre-training. The masking policy is trained with (answer, context) examples, and the policy is an extractive model that extracts the answer span from the context. For example, if the context $\mathbf{x}$ is [Charles, Schulz, was, the, creator, of, Snoopy] and the answer is "Charles Schulz", the label for the answer start index will be [1,0,0,0,0,0,0]; for end index it will be [0,1,0,0,0,0,0]. In the following, we briefly recap the method with our notations.

[2] 15% is borrowed from BERT and T5. "+Orig" and "+Rand" are different in that $\mathbf{x}^{(\text{tar})} = \mathbf{x}$ for "+Orig", while $\mathbf{x}^{(\text{tar})}$ contains the masked 15% tokens for "+Rand".

Model. Given context paragraph tokens $\mathbf{x} = [x_1, x_2, ..., x_m]$, we first use an embedding matrix $\mathbf{E}$ to embed each token: $[\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n]$. Then, we use a 2-layer bi-directional LSTM model to compute the hidden representation at each position.[3]

$$[\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n] = \text{Bi-LSTM}([\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n]) \quad (1)$$

Finally, we use two learned vectors $(\mathbf{w}_{st}, b_{st})$ and $(\mathbf{w}_{ed}, b_{ed})$ to compute the logits for each position being the start or end position of the potential answer/target span. For example, the logit of position j being a start/end position is computed as follows.

$$y_{j,st} = \mathbf{w}_{st}\mathbf{h}_j + b_{st}, \quad y_{j,ed} = \mathbf{w}_{ed}\mathbf{h}_j + b_{ed}. \quad (2)$$

Policy Inference. When deploying the policy to intermediate pre-training, we select the potential answer spans by ranking the sum of start and end logits of each potential spans, in accordance to the inference step in machine reading comprehension models. That is, we rank the spans (i, j) according to $y_{i,st} + y_{j,ed}$. We consider two variants when deploying the policy: (a) masking the top 1 span or (b) sampling 1 span from the top 5 spans.

Applicability and Limitation. Supervised policy is designed for closed-book QA, and one limitation of this method is that the target span $\mathbf{t}$ must appear *as is* in the context paragraph $\mathbf{c}$. Within all other knowledge intensive tasks, only ZSRE satisfies this constraint. To sum up, we apply supervised policy method to TQA, NQ, ZSRE.

4.3 Meta-learned Policy

Conceptually, what the learned masking policy captures is closely related to the concept of "learning to learn" (Schmidhuber, 1987; Thrun and Pratt, 1998). At a high level, the masking policy should provide the model with the desired initialization for the downstream task, such that the model can better learn the downstream task in only a few fine-tuning updates. Therefore, we construct a meta-learning approach, which we describe below.

Overview. We formulate each $(\mathbf{c}, \mathbf{s}, \mathbf{t})$ example as a small "task". For each task, the goal is to improve the performance of generating target sequence $\mathbf{t}$ given input $\mathbf{s}$, immediately after learning from the context $\mathbf{c}$. This is similar to taking quizzes, where a student first learns from a passage $\mathbf{c}$ and then is immediately tested on it by trying to answer $\mathbf{t}$ given $\mathbf{s}$. Studying from $\mathbf{c}$ strategically with an optimal masking policy will result in better performance (*i.e.*, smaller loss in generating $\mathbf{t}$).

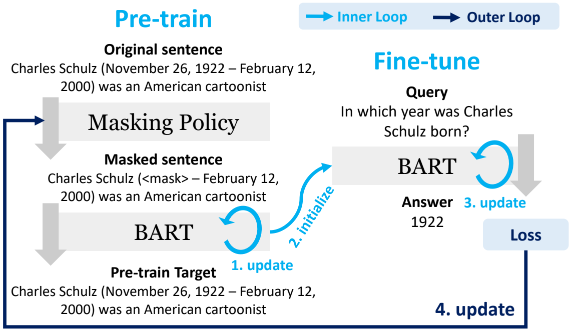

Pre-train
Inner Loop
Outer Loop
Original sentence
Charles Schulz (November 26, 1922 – February 12, 2000) was an American cartoonist
Masking Policy
Masked sentence
Charles Schulz (<mask> – February 12, 2000) was an American cartoonist
BART
1. update
Pre-train Target
Charles Schulz (November 26, 1922 – February 12, 2000) was an American cartoonist
2. initialize
Fine-tune
Query
In which year was Charles Schulz born?
BART
Answer
1922
3. update
Loss
4. update

Figure 2: **Update masking policy by learning from one context-query-answer example. (1) Inner Loop:** (a) A context paragraph c is first masked with current policy $g(.;\phi)$, and the language model is trained to recover masked tokens for one step; (b) the language model is trained on (q, a) pair for one step. **(2) Outer Loop:** We use the validation loss on the same (q, a) pair to update the masking policy, by directly taking the gradient of loss L w.r.t policy parameters ϕ.

Following work in gradient-based meta-learning (Finn et al., 2017; Grefenstette et al., 2019), we set up an *inner* and *outer* loop. We briefly sketch the procedure in Fig. 2. In the inner loop, we focus on the current $(\mathbf{c}, \mathbf{s}, \mathbf{t})$ examples by applying the current masking policy $g(.;\phi)$ and performing pre-train/fine-tune updates to $f(.;\theta)$. In the outer loop, we update the policy $g(.;\phi)$ with the signal at the end of inner loop training. We denote $\phi^{(p)}$ as the masking policy parameters after p outer loop optimization steps, and $\theta^{(p,q)}$ as the LM parameters after p outer loop optimization steps and q inner loop optimization steps.

Inner Loop. In one inner-loop curriculum, we first take the context as a pre-training sentence, *i.e.*, $\mathbf{x} = \mathbf{c}$, and use the current masking policy $g(.;\phi^{(p)})$ to determine the masks $\mathbf{d}$ and the implied perturbed input $\mathbf{x}^{(\text{src})}$, *i.e.*, $\mathbf{d} = g(\mathbf{x};\phi^{(p)})$, $\mathbf{x}^{(\text{src})} = \mathbf{x} \oplus \mathbf{d}$.

We pre-train $\theta^{(p,0)}$ for one step to recover $\mathbf{x}$ from $\mathbf{x}^{(\text{src})}$:

$$\theta^{(p,1)} = \theta^{(p,0)} - \alpha_0 \nabla_{\theta^{(p,0)}} \mathcal{L}(f(\mathbf{x}^{(\text{src})};\theta^{(p,0)}), \mathbf{x}), \quad (3)$$

where α_0 is the learning rate, and $\mathcal{L}(.,.)$ is the cross entropy loss for recovering $\mathbf{x}$ using disturbed input

[3] Though the masking policy can theoretically take any form, we opt for a lightweight architecture (2-layer Bi-LSTM) as we need to apply it to millions of pre-training instances.

$\mathbf{x}^{(\mathrm{src})}$ and model parameters $\theta^{(p,0)}$.

Next, we take $\theta^{(p,1)}$ as initialization and fine-tune it for one step on the downstream objective of predicting $\mathbf{t}$ given $\mathbf{s}$:

$$\theta^{(p,2)} = \theta^{(p,1)} - \alpha_1 \nabla_{\theta^{(p,1)}} \mathcal{L}(f(\mathbf{s}; \theta^{(p,1)}), \mathbf{t}). \quad (4)$$

Outer Loop. In outer loop, we update the masking policy $g(.;\phi^{(p)})$. We aim to answer query $\mathbf{s}$ correctly after the inner-loop curriculum. We define the meta-loss $\mathcal{L}'$ as the decrease in losses after the one fine-tuning update, *i.e.*,

$$\mathcal{L}' = \mathcal{L}(f(\mathbf{s}; \theta^{(p,2)}), \mathbf{t}) - \mathcal{L}(f(\mathbf{s}; \theta^{(p,1)}), \mathbf{t}). \quad (5)$$

$\mathcal{L}'$ characterizes how fast the model has adapted itself to answer $(\mathbf{s}, \mathbf{t})$ within one step of optimization. Since all computations in Eq. (3-5) are continuous[4], we optimize ϕ by directly taking gradients from $\mathcal{L}'$,

$$\phi^{(p+1)} = \phi^{(p)} - \alpha_2 \nabla_{\phi^{(p)}} \mathcal{L}'. \quad (6)$$

Controlling Masking Budget. Higher order optimization is known to be unstable (Antoniou et al., 2019). In early stages of the study, we found the policy to be flipping between masking none or all of the tokens. To stabilize, we add a softened L2 loss to control the portion of mask/not-mask decisions output by $g(.;\phi)$. Denoting $l(\mathbf{x})$ as the input sequence length, $l(\mathbf{d})$ as the number of mask decisions; we define a budget γ and a tolerance factor ϵ, and compute the regularization term $\mathcal{L}_{reg}$,

$$\mathcal{L}_{reg}(\mathbf{x}, \mathbf{d}) = \begin{cases} 0, & |\gamma l(\mathbf{x}) - l(\mathbf{d})| \le \epsilon l(\mathbf{x}) \\ (\gamma l(\mathbf{x}) - l(\mathbf{d}))^2, & \text{otherwise} \end{cases} \quad (7)$$

For example, when $\gamma = 15\%$, $\epsilon = 5\%$ and the input sequence $\mathbf{x}$ contains 100 tokens, the policy will not be penalized if it's masking 15 ± 5 of all tokens in the sequence. We modify the optimization step in Eq. (6) as follows, where β is a co-efficient balancing the regularization intensity.

$$\phi^{(p+1)} = \phi^{(p)} - \alpha_2 \nabla_{\phi^{(p)}} (\mathcal{L}' + \beta \mathcal{L}_{reg}(\mathbf{x}, \mathbf{d})) \quad (8)$$

Post-processing. When we deploy a learned policy to pre-training, we are no longer constrained by differentiability. Based on useful techniques in previous work, we apply post-processing to predicted masking decisions $\mathbf{d}$. (1) Whole-word masking and text infilling (Liu et al., 2019; Lewis et al., 2020): whenever one subword x_i within a whole word is masked ($d_i = 1$), we expand the mask and always mask the whole word. When consecutive tokens are masked, we replace the sequence of `<mask>` in the input sequence with exactly one `<mask>` token. (2) Additional budget control: Even with our budget regularization loss (Eq. 7), we find some input sequences get too many masks ($> 50\%$). This creates extremely challenging pre-train examples that may prevent the model from learning useful information. For these sentences we randomly "unmask" tokens to keep the portion of masks below 30%.

For space concerns, we leave pseudo-code and other implementation details in Appendix A.2.

5 Results and Discussion

Following our analysis setup (§3), we present the results for closed-book QA in Table 1, knowledge-intensive language tasks (KILT) in Table 2 and multiple-choice QA in Table 3. In the following, we aim to understand the influence brought by different masking policies through these results. We also introduce several ad-hoc experiments to verify our hypotheses raised in our analysis.

5.1 Comparison of Heuristic Policies

Continue pre-training with the original objective is helpful in general. Prior work has shown that intermediate pre-training on *encoder* models (*i.e.*, RoBERTa, Liu et al. 2019) with in-domain corpora helps to improve downstream *classification* tasks performance (Gururangan et al., 2020). Our experiments help to examine whether similar conclusion holds for *text-to-text* models and *tasks beyond classification*. From our results, we found intermediate pre-training with Wikipedia and BART's original objective (+Orig) improves performance of two closed-book QA tasks (TQA and WQ), one entity linking task (AY2), and two multiple-choice QA tasks (WIQA and QuaRTz); maintains performance on NQ and ZSRE; leads to worse performance on ROPES. Overall, intermediate pre-training leads to improved performance; this may be due to the common observation that language models tend to improve with further pre-training even after validation perplexity have plateaued, or that Wikipedia as a general knowledge-intensive corpus, is more closely related to our downstream tasks, compared to the

[4] We use Gumbel Softmax (Jang et al., 2017) to discretize the output of $g(.;\phi^{(p)})$ and formulate masking decision $\mathbf{d}$, and we use embedding mixture for $\mathbf{x}^{(\mathrm{src})} = \mathbf{x} \oplus \mathbf{d}$

	TQA	WQ	NQ
BART-Base	$21.82_{\pm.15}$	$26.23_{\pm.05}$	$23.72_{\pm.25}$
+Orig	$22.91_{\pm.16}$	$27.17_{\pm.56}$	$23.85_{\pm.37}$
+Rand	$22.93_{\pm.14}$	$27.25_{\pm.68}$	$24.64_{\pm.44}$
+SSM	$23.62_{\pm.29}$	$28.17_{\pm.04}$	$24.80_{\pm.06}$
+Supervised-NQ(Top1)	$23.48_{\pm.10}$	$27.43_{\pm.38}$	$24.58_{\pm.10}$
+Supervised-NQ(Top5)	$23.73_{\pm.21}$	$28.15_{\pm.05}$	$24.86_{\pm.28}$
+Supervised-TQA(Top1)	$24.71_{\pm.21}$	$27.84_{\pm.03}$	$24.58_{\pm.19}$
+Supervised-TQA(Top5)	$24.43_{\pm.09}$	$28.35_{\pm.73}$	$24.66_{\pm.22}$
+Meta-learned-NQ	$23.50_{\pm.28}$	$27.07_{\pm.20}$	$24.83_{\pm.18}$
+Meta-learned-TQA	$23.88_{\pm.04}$	$27.49_{\pm.17}$	$24.85_{\pm.21}$
BART-Large	$24.28_{\pm.51}$	$28.82_{\pm.33}$	$24.72_{\pm.16}$
+Orig	$24.34_{\pm.35}$	$28.28_{\pm.35}$	$24.91_{\pm.68}$
+SSM	$26.29_{\pm.43}$	$29.79_{\pm.47}$	$25.34_{\pm.23}$
+Supervised-TQA(Top1)	$27.18_{\pm.34}$	$29.71_{\pm.74}$	$24.28_{\pm.28}$

Table 1: **Performance of Closed-book QA Tasks.** We report average and standard deviation of exact match over three runs with different random seeds. Dark blue highlights the best performing model. Light blue highlights models that are not significantly worse than the best performing model (p>0.1 in paired t-test).

	AY2	ZSRE	WoW
Metric	EM	EM	F1
BART-Base	$81.07_{\pm.15}$	$1.89_{\pm.15}$	$15.14_{\pm.22}$
+Orig	$81.38_{\pm.06}$	$1.67_{\pm.15}$	$15.20_{\pm.13}$
+Rand	$81.67_{\pm.13}$	$2.29_{\pm.19}$	$14.69_{\pm.21}$
+SSM	$81.74_{\pm.19}$	$3.52_{\pm.03}$	$14.68_{\pm.16}$
+Supervised-ZSRE(Top1)	$81.57_{\pm.03}$	$2.84_{\pm.15}$	$14.58_{\pm.01}$
+Supervised-ZSRE(Top5)	$81.90_{\pm.22}$	$2.90_{\pm.03}$	$14.50_{\pm.38}$
+Meta-learned-ZSRE	$81.31_{\pm.22}$	$1.99_{\pm.21}$	$15.07_{\pm.09}$
+Meta-learned-WoW	$80.90_{\pm.23}$	$1.64_{\pm.05}$	$15.32_{\pm.05}$

Table 2: **Performance of KILT Tasks.**

mixture of corpus[5] used to pre-train BART.

A closer look at ROPES, the exception. We notice that the context paragraphs in ROPES are from science textbooks *and* Wikipedia. We hypothesize that intermediate pre-training on *only* Wikipedia may cause catastrophic forgetting of some scientific knowledge obtained during BART's general pre-training. To verify this, we randomly mask 15% tokens in ROPES context paragraphs and computed MLM loss. The BART-Base checkpoint achieves 1.97 in NLL Loss, while +Orig achieves 2.02. This supports our hypothesis that intermediate pre-training on a smaller corpus (e.g., Wikipedia) may make the model forget knowledge in general pre-training (e.g., scientific textbooks). We conclude that it is important to pay attention to the corpus from which the downstream dataset is created.

Select the heuristic masking policy that resemble the downstream task most. So far, we limit the focus to the +Orig objective. Now we further add +Rand and +SSM into the comparison. From the results in Table 1, we first confirm that salient span masking (SSM) is indeed very beneficial for closed-book QA (Roberts et al., 2020). In addition, SSM helps improve performance for two entity-centric knowledge intensive tasks (AY2 and ZSRE, see Table 2) and two multiple-choice QA tasks (ROPES and QuaRTz, see Table 3). Note that ROPES focus on causal relationships between entities and QuaRTz focus on qualitative relations (involving numbers); both can be considered entity-centric. We conclude that using heuristic masking policies that resemble the downstream tasks, or masking information known to be important for the downstream task, tend to improve downstream performance. When it's difficult to design a heuristic that satisfy these needs, using random masking may be helpful. In this case, we recommend to decide whether to generate full sequence (+Orig) or only masked tokens (+Rand) based on the task output length. If the downstream tasks requires generating long sentences, generating full sequence is more helpful. This is supported by the observation that +Orig is better than +Rand for WoW. On the other hand, if the target sequences in the downstream dataset are shorter, generating masked tokens is more helpful, as shown by experiments on NQ, AY2, ZSRE and ROPES.

	ROPES	WIQA	QuaRTz
BART-Base	$46.60_{\pm0.48}$	$71.18_{\pm1.12}$	$62.80_{\pm1.16}$
+Orig	$43.68_{\pm0.67}$	$73.06_{\pm0.72}$	$63.35_{\pm0.52}$
+Rand	$44.59_{\pm1.15}$	$70.55_{\pm0.42}$	$63.31_{\pm1.74}$
+SSM	$50.51_{\pm1.15}$	$69.31_{\pm0.77}$	$64.41_{\pm1.04}$
+Meta-learned-ROPES	$53.71_{\pm2.33}$	$73.05_{\pm0.98}$	$62.93_{\pm1.28}$
+Meta-learned-WIQA	$48.30_{\pm0.69}$	$72.38_{\pm0.37}$	$63.14_{\pm1.26}$
+Meta-learned-QuaRTz	$49.01_{\pm1.92}$	$72.65_{\pm0.53}$	$63.69_{\pm0.48}$

Table 3: **Performance of Multiple-choice QA Tasks.** We report accuracy for each task.

5.2 How Do Learned Policies Perform?

We have introduced two ways to automate the discovery of better masking policies, with supervised learning (§4.2) and meta-learning (§4.3). We now extend our analysis to these learned policies.

Successful Cases. We observe that learned policies are most successful on TriviaQA, with both the supervised policy and the meta-learned policy outperforming SSM. We attribute its success to the following reasons: (1) (context, source, target) examples are abundant, so the masking policy has

[5] Similar to RoBERTa, BART uses the combination of BookCorpus (Zhu et al., 2015), CC-News (Nagel, 2016), OpenWebText (Gokaslan and Cohen, 2019), and Stories (Trinh and Le, 2018) as pre-training corpus.

Training Data Used	0.1%	1%
(a) BART-Base	3.69%	5.54%
(b) +SSM	5.56%	7.31%
(c) +Supervised-TQA(Top1)	6.49%	8.40%
(b) +Meta-learned-TQA	4.50%	6.44%

Table 4: **Performance of TriviaQA in low-resource settings.** Exact match is reported. Supervised policy outperforms other masking policies in low-resource setting, consistent with the full-dataset setting.

sufficient supervision. TriviaQA dataset is accompanied with large-scale context paragraphs created with distant supervision, so the scale of $(\mathbf{c}, \mathbf{s}, \mathbf{t})$ examples is larger than other datasets. (2) The heuristic masking policy does not "perfectly" resemble the downstream task, and it still has room for improvement. SSM masks one random named entity in the context. However, the answer to trivia questions are not necessarily named entities, and one named entity may be more important than another. Therefore the learned policies can better capture the characteristics of TriviaQA than SSM. Apart from TriviaQA, meta-learned policies outperforms +Orig on NQ, ZSRE and ROPES, demonstrating the effectiveness of the method. This also opens up a promising direction for downstream tasks whose heuristic masking policy is not intuitive (*e.g.*, dialogue response generation, multiple-choice QA).

Improved learning efficiency. We additionally consider a low-resource setting for TriviaQA, where we use 0.1% and 1% of its training set for fine-tuning. We present the results in Table 4. We observe that the supervised policy has better sample efficiency than SSM. We also observe that intermediate pre-training by generating full sequence (a/d) is worse than generating spans (b/c), supporting our previous conclusion that the choice of target sequences should be based on the downstream task output format (span or sentence).

Overfitting on ZSRE. ZSRE dataset has a unique setting: it is a slot filling task similar to close-book QA; however it adds additional challenge as the relations in train/dev/test splits are non-overlapping. We hypothesize that this train/test discrepancy leads to unsatisfactory behavior of learned ZSRE policies, and we conduct a set of controlled experiment to validate this hypothesis. Concretely, we use 90% of its original train set as the new train set, use the 10% remaining training examples as a "matched" dev set, and the original dev set as a "mismatched" dev set. In our experiments, SSM achieves $20.02\%_{\pm.16\%}$ EM on match-dev, and $3.21\%_{\pm.15\%}$ on mismatch-dev. Supervised-ZSRE(Top5) achieves $20.37\%_{\pm.04\%}$ on match-dev (outperforms SSM, $p<0.05$), and $2.94\%_{\pm.11\%}$ on mismatch-dev. These experiments show that our supervised policy is learning useful information, but has overfitted to the training data and becomes less robust to distribution shift during inference. In comparison, SSM is agnostic to train-test discrepancy and thus achieves the strongest performance.

Generalization of learned policies. We observe several cases where a policy learned from one dataset positively transfer to another downstream tasks. That includes Supervised-TQA(Top5) bringing improvement to WQ, +Supervised-NQ(Top5) bringing improvement to TQA, and +Supervised-ZSRE(Top5) bringing improvement to AY2, compared to random masking baselines. This is reasonable since all these tasks are entity-centric and are similar in nature. For tasks with significantly different formats and goals, *e.g.*, ZSRE and WoW, policies learned on one does not benefit the other. Here we only exhibit the evidence supporting that learned masking policies can positively transfer, and we leave the question of "when and why does it work" as future work.

Remarks. Supervised/meta-learned masking policies are our initial attempt towards the idea of "learning to mask". While being successful and exhibiting the evidence for positive transfer in certain cases, we recognize the potential risks of overfitting, or suffering from high instability in meta-learning. We hope future work can investigate these issues and design novel methods to learn better masking policies.

5.3 Quantitative Analysis: What are Masked? How are policies different?

In this section we aim to understand how masking policies are different from each other in terms of their masking decisions. We analyze the relation between masking decisions, part-of-speech tags and token frequency. Specifically, we take 1% of the pre-train corpus and compare the masking decisions made by each policy to facilitate our analysis.

Relation to Part-of-speech Tags. In Fig. 3, we plot the stacked bar chart of part-of-speech tags to visualize their distribution. Each bar represent the portion of masks having the part-of-speech tag, amongst all masks produced by this policy. Notably, most *supervised* policies learns to focus more on proper nouns, and less on common nouns.

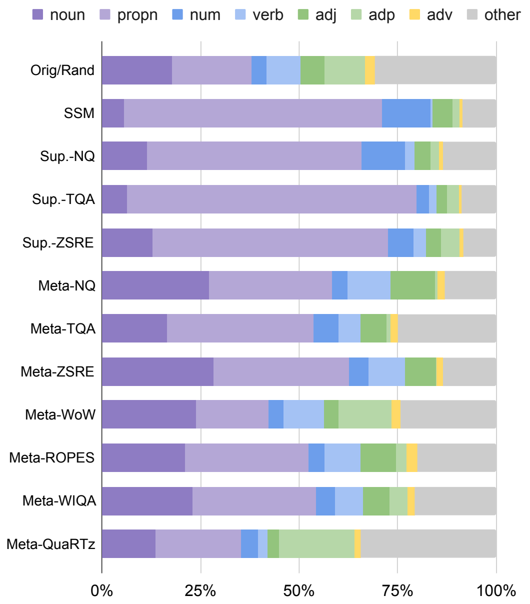
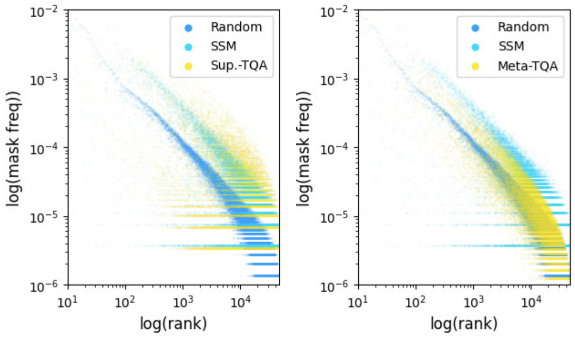


Figure 3: Part-of-speech tag distribution for masked produced by different masking policies.

This is consistent with the goal of the entity-centric downstream tasks. Comparing Supervised-TQA and SSM, Supervised-TQA focuses less on nouns, numbers and adjectives, and it focuses even more on proper nouns. This suggests that Supervised-TQA better characterizes the property of TQA, and thus outperforms SSM by learning to mask task-specific information. Due to the differences in learning procedures, the *meta-learned* policies has distributions different from supervised policies. Still, meta-learned policies for NQ and TQA masks more proper nouns compared to random masking, similar to their supervised counterparts.

Relation to Token Frequency. In Fig. 4, we plot the relation between mask frequency and token frequency for masking policies learned from TQA, along with random masking and SSM for reference. Mask frequency is computed as the number of occurrences that a token was masked divided by the number of all masked tokens. For random masking, the datapoints approximate a Zipfian distribution (Zipf, 1999), with some noise due to random sampling of words. Secondly, for SSM, most datapoints fall on a curve above the random masking line, while a small portion of tokens are less likely to be masked, formulating line segments in the bottom area. These observations indicate that SSM tend to mask less frequent tokens, but its behavior is not fully explained away with token frequency. The two learned policies, Supervised-TQA (Fig. 4(a)) and Meta-TQA (Fig. 4(b)) are in general similar to SSM, while the curve for Supervised-TQA is more scattered, indicating a weaker preference for Zipfian behavior.



(a) Supervised-TriviaQA (b) Meta-learned-TriviaQA

Figure 4: Relation between token frequency rank in the corpus and the token's mask frequency. Best viewed in color.

6 Related Work

Implicit Knowledge in Pre-trained Language Models. Petroni et al. (2019) discovered that pre-trained language models can implicitly store relational knowledge in their parameters, and such knowledge can be accessed with cloze-style queries. Roberts et al. (2020) introduced the task of closed-book QA, which breaks the convention of retriever-reader strategy for open-domain QA, and requires the model to directly generate answers with its implicit knowledge. Closed-book QA performance is boosted significantly when salient span masking (Guu et al., 2020) is used. Guu et al. (2020) maintained that SSM helps the model to "focus on problems that require world knowledge".

Self-supervised Pre-training. Pre-trained language models has shown its capability on a wide variety of NLP tasks. Current self-supervised objectives are mostly *heuristic*, including masked language modeling (Devlin et al., 2019), span boundary representation learning (Joshi et al., 2020), corrupted sentence reconstruction (Lewis et al., 2020), etc. Raffel et al. (2020) systematically studied the self-supervised objectives used in previous literature. Related to our goal of exploring pre-training objectives, ELECTRA (Clark et al., 2020) propose a replaced token prediction task which improves pre-training efficiency. Chen et al. (2020) propose to reduce the variance of gradients in SGD and expedite model pre-training. Levine et al. (2020)

propose to mask n-grams according to Pointwise Mutual Information (PMI). These works typically consider the efficiency of an objective when pre-training *from scratch* and without preconceived focus on a given problem; while we focus on encoding knowledge or adapting the model during intermediate pre-training with a given task in mind.

Domain/Task-specific Pre-training. Gururangan et al. (2020) experiment on four domains (biomedical, computer science, news, reviews) and eight different datasets, where they discover that pre-training with in-domain corpus leads to better downstream performance. Kang et al. (2020) propose to learn a mask generator via reinforcement learning. Closely related to us, Gu et al. (2020) propose task-guided pre-training by learning to predict importance score for each token in pre-train corpus. Vu et al. (2020); Pruksachatkun et al. (2020) studies knowledge transfer from intermediate-task fine-tuning, while we focus on a different problem setting of intermediate pre-training with generic corpus (*e.g.*, Wikipedia). We believe both settings have practical utility in real-world applications.

7 Conclusion

In this paper, we study the influence brought by different masking policies used during intermediate pre-training, and offer two methods as our initial attempts towards automating the discovery of optimal masking policy. From extensive experiments with heuristic and learned masking policies across three categories of tasks, we have identified several successful cases of intermediate pre-training, offered in-depth analysis and insights for the masking policies we used, discussed the risks of learned masking policies, and summarized several suggestions for researchers who wish to adopt intermediate pre-training in their applications.

We also acknowledge that, despite our additional efforts and experiments, several observations still cannot be explained away. We invite future research into this challenging and under-explored problem, to expand on our methods, and to search the space of pre-training objectives beyond masked language modeling. Furthermore, we hope our work encourages researchers to consider the type of downstream applications they wish to deploy their LMs in, before investing resources into large-scale pre-training.

Acknowledgments

We would like to thank anonymous reviewers for their constructive feedback. We would like to thank Sebastian Riedel, Edward Grefenstette, Douwe Kiela, Patrick Lewis, Xian Li, Sewon Min, and USC INK Lab members for the insightful discussions. Qinyuan Ye and Xiang Ren are supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via Contract No. 2019-19051600007; the DARPA MCS program under Contract No. N660011924033; the Defense Advanced Research Projects Agency with award W911NF-19-20271; NSF IIS 2048211.

References

Antreas Antoniou, Harrison Edwards, and Amos Storkey. 2019. How to train your MAML. In *International Conference on Learning Representations*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Liang Chen, Tianyuan Zhang, Di He, Guolin Ke, Liwei Wang, and Tie-Yan Liu. 2020. Variance-reduced language pretraining via a mask proposal network. *arXiv preprint arXiv:2008.05333*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, M. Auli, and J. Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. *ArXiv*, abs/1811.01241.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135, International Convention Centre, Sydney, Australia. PMLR.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. *URl: https://skylion007. github. io/OpenWebTextCorpus*.

Edward Grefenstette, Brandon Amos, Denis Yarats, Phu Mon Htut, Artem Molchanov, Franziska Meier, Douwe Kiela, Kyunghyun Cho, and Soumith Chintala. 2019. Generalized inner loop meta-learning. *arXiv preprint arXiv:1910.01727*.

Yuxian Gu, Zhengyan Zhang, Xiaozhi Wang, Zhiyuan Liu, and Maosong Sun. 2020. Train no evil: Selective masking for task-guided pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6966–6974, Online. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-augmented language model pre-training. In *International Conference on Machine Learning (ICML)*.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 782–792, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. *ArXiv*, abs/1611.01144.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Minki Kang, Moonsu Han, and Sung Ju Hwang. 2020. Neural mask generator: Learning to generate adaptive word maskings for language model adaptation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6102–6120, Online. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Yoav Levine, Barak Lenz, Opher Lieber, Omri Abend, Kevin Leyton-Brown, Moshe Tennenholtz, and Yoav Shoham. 2020. PMI-masking: Principled masking of correlated spans. *ArXiv*, abs/2010.01825.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. 2019. Reasoning over paragraph effects in situations. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 58–62, Hong Kong, China. Association for Computational Linguistics.

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 preprint arXiv:1907.11692*.

Sebastian Nagel. 2016. Cc-news. *URL: http://web. archive. org/save/http://commoncrawl. org/2016/10/newsdatasetavailable*.

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 *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vassilis Plachouras, Tim Rocktäschel, et al. 2020. Kilt: a benchmark for knowledge intensive language tasks. *arXiv preprint arXiv:2009.02252*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5231–5247, Online. Association for Computational Linguistics.

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. *Journal of Machine Learning Research*, 21(140):1–67.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Jürgen Schmidhuber. 1987. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. Ph.D. thesis, Technische Universität München.

Oyvind Tafjord, Matt Gardner, Kevin Lin, and Peter Clark. 2019. QuaRTz: An open-domain dataset of qualitative relationship questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5941–5946, Hong Kong, China. Association for Computational Linguistics.

Niket Tandon, Bhavana Dalvi, Keisuke Sakaguchi, Peter Clark, and Antoine Bosselut. 2019. WIQA: A dataset for "what if..." reasoning over procedural text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6076–6085, Hong Kong, China. Association for Computational Linguistics.

Sebastian Thrun and Lorien Pratt. 1998. Learning to learn: Introduction and overview. In *Learning to learn*, pages 3–17. Springer.

Trieu H Trinh and Quoc V Le. 2018. A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847*.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across NLP tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7882–7926, Online. Association for Computational Linguistics.

Qinyuan Ye, Belinda Z. Li, Sinong Wang, Benjamin Bolte, Hao Ma, X. Ren, Wen tau Yih, and Madian Khabsa. 2020. Studying strategically: Learning to mask for closed-book qa. *ArXiv*, abs/2012.15856.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

George Kingsley Zipf. 1999. *The psycho-biology of language: An introduction to dynamic philology*, volume 21. Psychology Press.

A Additional Training Details

A.1 Supervised Policy

Training Details. The embedding matrix $\mathbf{E}$ is initialized with the weights in BART-base model. We optimize cross entropy loss between the logits outputted by the model and the gold annotations. For each source of supervision stated above, we train the policy for 30 epochs with learning rate of 1e-5 and batch size of 512, and select the best checkpoint according to validation loss.

A.2 Meta-learned Policy

Design choices. We use a 1D convolution layer with two additional linear layers as our policy network $g(.;\phi)$. The linear layers output two logits for each token in input sequence $\mathbf{x}$. The two logits for each tokens go through Gumbel Softmax (Jang et al., 2017) to decide whether it should be masked ($d_i = 1$) or not ($d_i = 0$). We've also experimented with Bi-LSTM as the encoder, but find meta-learning with LSTMs to be extremely unstable.

Intuitive Example. The PTLM f is given a piece of context $\mathbf{c}$ "Charles Schulz (November 26, 1922 – February 12, 2000) was an American cartoonist" and is expected to take an upcoming "closed-book exam" based on this piece of context. In the pre-train step, the current policy g predicts masks (e.g., Charles Schulz (<mask> – February 12, 2000) was an American cartoonist) and take one step of optimization, implicitly encoding this piece of knowledge into its parameters. After this, the PTLM "transit to closed-book exam mode" by fine-tuning on $(\mathbf{s}, \mathbf{t})$ for one step. Finally the language model "takes the closed-book exam" and the loss for generating $\mathbf{t}$ given $\mathbf{s}$ as input can be interpreted as the supervision for the masking decisions (i.e., whether masking "November 26, 1922" is helpful).

Pseudo-code. We provide pseudo-code for our method in Algorithm 1.

B Hyperparameters

For downstream task fine-tuning, we first select the learning rate from {5e-6, 1e-5, 2e-5, 3e-5} and then fix learning rate to select batch size from {32, 64, 128, 256}. See Table 5 for more details.

Parameter Name	Value
Max Epoch	100
Validation Interval	2 or 5
Warmup Updates	500
Learning Rate	{5e-6, 1e-5, 2e-5, 3e-5}
Batch Size	{32, 64, 128, 256}
Label Smoothing	0.1
Dropout	0.1
Weight Decay	0.01
Clip Norm	0.1
Generation Beam Size	4
Generation Min/Max Length	1/20
Generation Length Penalty	1.0

Table 5: Hyperparameters for Downstream Task Fine-tuning.

C Discussion on NQ

In Table 1 we observe that performances on NQ are close for all BART-base models; therefore it is hard to rank all compared methods. We argue that multiple factors leads to this phenomenon, including dataset characteristics and evaluation protocol. Specifically, NQ may not be an ideal testbed for our study due to three reasons.

Firstly, intermediate pre-training in general might not be as beneficial for this particular task. For instance, Roberts et al. (2020) reports only 2% EM gain on NQ using T5-11B. In our experiments, we use significantly smaller pre-trained models (BART-base/large), so the effect brought by intermediate pre-training will be even smaller. In our case we believe the effect is hidden in the variance brought by random seeds.

Secondly, performance on NQ may not represent the real implicit knowledge capacity of a LM. For reference, we observe a 20% dev set EM when fine-tuning a randomly initialized BART-base model on NQ. The general pre-training stage brings merely 4-5% EM improvement, and therefore the improvement brought by intermediate pre-training can be marginal.

And finally, evaluation based on exact match may substantially underestimate the model capability, as suggested in (Roberts et al., 2020).

D Reproducibility

D.1 Dataset Details

We obtain closed-book QA datasets from https://github.com/facebookresearch/DPR/blob/master/data/download_data.py, knowledge-intensive language tasks from https://github.com/facebookresearch/KILT/blob/master/scripts/donwload_all_kilt_data.py. We

Algorithm 1 Meta-learning Policy $g(.;\phi)$

```
Input: Dataset $\mathcal{S} = \{(\mathbf{c}, \mathbf{s}, \mathbf{t})\}$
Output: Masking Policy $g(.;\phi)$, A Pre-trained Language Model $f(.;\theta)$
for $p = 1..T$ do
    $\{(\mathbf{c}, \mathbf{s}, \mathbf{t})\} = \text{SampleBatch}(\mathcal{S})$
    $\mathbf{xc}; \mathbf{d} = g(\mathbf{x}; \phi^{(p)}); \mathbf{x}' = \mathbf{x} \oplus \mathbf{d}$    // Apply current masking policy $g(.;\phi)$
    $\theta^{(p,1)} = \theta^{(p,0)} - \alpha_0 \nabla_{\theta^{(p,0)}} \mathcal{L}(f(\mathbf{x}'; \theta^{(p,0)}), \mathbf{x})$    // Inner Loop Update 1: Pre-train on $(\mathbf{x}, \mathbf{x}')$
    $\theta^{(p,2)} = \theta^{(p,1)} - \alpha_1 \nabla_{\theta^{(p,1)}} \mathcal{L}(f(\mathbf{s}; \theta^{(p,1)}), \mathbf{t})$    // Inner Loop Update 2: Fine-tune on $(\mathbf{s}, \mathbf{t})$
    $\mathcal{L}' = \mathcal{L}(f(\mathbf{s}; \theta^{(p,2)}), \mathbf{t}) - \mathcal{L}(f(\mathbf{s}; \theta^{(p,1)}), \mathbf{t})$    // Compute loss measuring how fast the model adapts to $(\mathbf{s}, \mathbf{t})$
    $\mathcal{L}_{reg} = \delta[|\gamma l(\mathbf{x}) - l(\mathbf{d})| > \epsilon l(\mathbf{x})](\gamma l(\mathbf{x}) - l(\mathbf{d}))^2$    // Compute regularization loss to control masking budget
    $\phi^{(p+1)} = \phi^{(p)} - \alpha_2 \nabla_{\phi^{(p)}} (\mathcal{L}' + \beta \mathcal{L}_{reg}(\mathbf{x}, \mathbf{d}))$    // Outer Loop Update for $\phi$
    $\theta^{(p+1,0)} = \theta^{(p,1)}$    // Maintain pre-train progress at timestamp $p$
```

Category	Dataset	#Train	#Dev	#Test
Closed-book QA	Natural Questions (Kwiatkowski et al., 2019)	79,168	8,757	3,610
	WebQuestions (Berant et al., 2013)	3,417	361	2,032
	TriviaQA (Joshi et al., 2017)	78,785	8,837	11,313
Knowledge-Intensive Tasks (KILT)	Aidayago2 (Hoffart et al., 2011)	18,395	4,784	4,463
	Zero-shot Relation Extraction (Levy et al., 2017)	147,909	3,724	4,966
	Wizard of Wikipedia (Dinan et al., 2019)	94,577	3,058	2,944
Knowledge-Intensive Multiple-choice QA	ROPES (Lin et al., 2019)	10,924	844	844
	WIQA (Tandon et al., 2019)	29,808	6,894	3,003
	QuaRTz (Tafjord et al., 2019)	2,696	384	784

Table 6: Details of Datasets Used in This Study.

obtain ROPES, WIQA and QuaRTz from huggingface datasets (https://huggingface.co/datasets). For more details, see Table 6. KILT hosts the test set evaluation on its leaderboard and the test set annotations are not publicly available; therefore we report performance on dev set in Table 2. The test set annotations for ROPES is not publicly available, so we take 50% of original dev set as the new dev set, and the other 50% as the new test set.

D.2 Training Details

Implementation. All our experiments are implemented with `fairseq` (Ott et al., 2019). For higher-order optimization in the meta-learning approach optimization, we use `higher` library (Grefenstette et al., 2019). Our code will be released upon acceptance.

Infrastructure and Runtime. Intermediate pre-training experiments are done with NVIDIA Quadro GP100 or NVIDIA Tesla V100 GPUs, based on availability. For BART-Base, we use 32 GPUs in parallel; For BART-Large, we use 64 GPUs in parallel. Pre-train job takes less than 24 hours for BART-Base models and less than 48 hours for BART-Large models. The checkpoints from intermediate pre-training will be released upon acceptance. Fine-tuning jobs are all done with one single GPU, with either NVIDIA Quadro GP100, NVIDIA Quadro RTX 8000, NVIDIA Quadro RTX 6000, NVIDIA GeForce RTX 1080 Ti, or NVIDIA GeForce RTX 2080 Ti, based on availability. The list the estimated maximum training time in the following: NQ (4h), WQ (2h), TQA (40h), AY2 (4h), ZSRE (2h), ROPES (1h), WIQA (1h), QuaRTz (1h).

Number of Parameters. BART-Base model contains 140 million parameters, BART-Large model contains 406 million parameters. Supervised policies contain 43 million parameters (where the word embeddings take 39 millions parameters). Meta-learned policies contain 40 million parameters (where the word embeddings take 39 millions parameters).