Should You Mask 15% in Masked Language Modeling?

Alexander Wettig* **Tianyu Gao*** **Zexuan Zhong** **Danqi Chen**
Department of Computer Science, Princeton University
{awettig,tianyug,zzhong,danqic}@cs.princeton.edu

Abstract

Masked language models (MLMs) conventionally mask 15% of tokens due to the belief that more masking would leave insufficient context to learn good representations; this masking rate has been widely used, regardless of model sizes or masking strategies. In this work, we revisit this important choice of MLM pre-training. We first establish that 15% is not universally optimal, and larger models should adopt a higher masking rate. Specifically, we find that masking 40% outperforms 15% for BERT-large size models on GLUE and SQuAD. Interestingly, an extremely high masking rate of 80% can still preserve 95% fine-tuning performance and most of the accuracy in linguistic probing, challenging the conventional wisdom about the role of the masking rate. We then examine the interplay between masking rates and masking strategies and find that uniform masking requires a higher masking rate compared to sophisticated masking strategies such as span or PMI masking. Finally, we argue that increasing the masking rate has two distinct effects: it leads to more corruption, which makes the prediction task harder; it also enables more predictions, which benefits optimization. Using this framework, we revisit BERT's 80-10-10 corruption strategy. Together, our results contribute to a better understanding of MLM pre-training.[1]

1 Introduction

Pre-trained language models have transformed the landscape of natural language processing (Devlin et al., 2019; Liu et al., 2019; Raffel et al., 2020; Brown et al., 2020, *inter alia*). They are trained on vast quantities of text data and acquire rich and versatile language representations. Compared to autoregressive models, which always predict the next token in a sequence, masked language models (MLMs) like BERT (Devlin et al., 2019) predict a masked subset of input tokens based on the remaining context and are more effective on downstream tasks due to their bidirectional nature.

BERT chooses a 15% masking rate, based on the reasoning that models cannot learn good representations when too much text is masked, and the training is inefficient when too little is masked. Surprisingly, this important choice has been underexplored since 15% masking is used ubiquitously by BERT's successors (Liu et al., 2019; Joshi et al., 2020; Lan et al., 2020; He et al., 2021; Levine et al., 2021; Izsak et al., 2021), regardless of model sizes, masking strategies and optimization recipes.[2]

In this work, we aim to understand the impact of masking rates. We hypothesize that the optimal masking rate is not universally 15%, but should depend on other factors. First, we consider the impact of model sizes and establish that indeed larger models should adopt higher masking rates (§3). Specifically, we find that under an efficient pre-training recipe (Izsak et al., 2021), 40% outperforms 15% for BERT-large size models when fine-tuning on GLUE and SQuAD.

Interestingly, we observe that large models can still learn good representations even for very high masking rates: if we mask as much as 80% of input tokens and pre-trained models have a perplexity of more than 1000, the learned representations can still preserve more than 95% of fine-tuning performance on downstream tasks, compared to the default 15% masking (Table 1), and show considerable performance in linguistic probing (§4). This challenges common intuitions about masking rates and what models learn in MLM pre-training.

We then focus on the strategy of which tokens to mask as an additional factor to the optimal masking rate of MLMs (§5). We find that different masking rates should be used with different masking strategies, and the default uniform masking bene-

*The first two authors contributed equally.
[1]Our code and pre-trained models are publicly available at https://github.com/princeton-nlp/DinkyTrain.
[2]Some exceptions are discussed in §8.

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2985–3000

	Pre-training			Fine-tuning		
m	Example	PPL	MNLI	QNLI	SQuAD[3]
15%	We study high ▇▇ ing rates ▇ pre-training language models .	17.7	84.2	90.9	88.0
40%	We study high ▇▇ ▇▇ rates ▇ pre- ▇▇▇▇ ▇▇▇▇ models .	69.4	**84.5** ↑0.3	**91.6** ↑0.7	**89.8** ↑1.8
80%	We ▇▇▇ high ▇▇ ▇▇ ▇▇ ▇ ▇▇▇▇ ▇▇▇▇ models ▇	1141.4	80.8 ↓3.4	87.9 ↓3.0	86.2 ↓1.8
	Random initialization		61.5 ↓22.7	60.9 ↓30.0	10.8 ↓77.2

Table 1: Masked examples, validation perplexity (calculated in the same way as Devlin et al., 2019) of different masking rates on the one billion word benchmark (Chelba et al., 2013), and downstream task development performance (SQuAD: F1; accuracy for others). All the pre-trained models have a BERT-large architecture and are trained with the efficient pre-training recipe (§2.2). Full results are provided in Table 7.

fits more from higher masking rates than more sophisticated masking strategies such as span (Joshi et al., 2020; Raffel et al., 2020) and PMI masking (Levine et al., 2021); when all methods are considered at their optimal masking rate, uniform masking achieves competitive performance.

Finally, we propose to dissect the masking rate into two factors (§6): the *corruption rate*—how much of the context is corrupted (masked)—and the *prediction rate*—how much of the tokens the model predicts on. In MLMs, both are set to the masking rate. However, these two factors have opposing effects: higher prediction rates generate more training signals and benefit the optimization, while higher corruption rates make the prediction task more challenging by providing less context. To study the two factors independently, we design ablation experiments to disentangle corruption and prediction rates. Thus, we can verify that models benefit from higher prediction rates and suffer from more corruption. Using this framework, we also discuss BERT's practice of predicting on original or random tokens (the 80-10-10 rule), and we find that models usually perform worse under this corruption strategy (§7).

Together, our results demonstrate the overlooked impact of the masking rate in MLM pre-training and our analysis disentangles its opposing effects of corruption and prediction. We conclude by discussing the relation to work in other models and modalities (§8) and by highlighting several new avenues for efficient MLM in the future (§9).

2 Background

2.1 Masked Language Modeling

We focus on the widely popular masked language modeling (Devlin et al., 2019), a form of denoising-autoencoding, where a model is trained to restore a corrupted input sequence. Specifically, masked language models make independent predictions on the subset of masked tokens:

$$L(\mathcal{C}) = \mathop{\mathbb{E}}_{x \in \mathcal{C}} \mathop{\mathbb{E}}_{\substack{\mathcal{M} \subset x \\ |\mathcal{M}|=m|x|}} \left[\sum_{x_i \in \mathcal{M}} \log p(x_i | \tilde{x}) \right], \quad (1)$$

where one masks m (masking rate, typically 15%) percentage of tokens from the original sentence x and predicts on the masked token set $\mathcal{M}$ given the corrupted context $\tilde{x}$ (the masked version of x).

Different masking strategies have been proposed to sample $\mathcal{M}$: Devlin et al. (2019) randomly choose from the input tokens with a uniform distribution; Joshi et al. (2020) sample contiguous spans of text; Levine et al. (2021) sample words and spans with high pointwise mutual information (PMI). These advanced sampling strategies are adopted to prevent models from exploiting shallow local cues from uniform masking.

MLMs can encode bidirectional context while autoregressive language models can only "look at the past", and thus MLMs are shown to be more effective at learning contextualized representations for downstream use (Devlin et al., 2019). On the other hand, MLMs suffer a significant computational cost because it only learns from 15% of the tokens per sequence, whereas autoregressive LMs predict every token in a sequence. In this work, we focus on MLMs and study the effects of different masking rates on downstream performance.

2.2 Experiment Setup

We build most of our experiments on a recent efficient pre-training recipe—the 24hBERT recipe from Izsak et al. (2021)—by using which models can match BERT-base performance 6× faster (tested on 8×Titan-V). This efficient pre-training recipe allows us to run a large amount of experiments in an academic setup. Izsak et al. (2021)

[3]For our SQuAD v1.1 experiments, we continue training the models with 512-token sequences for 2,300 steps and report F1. See Appendix A for more details.

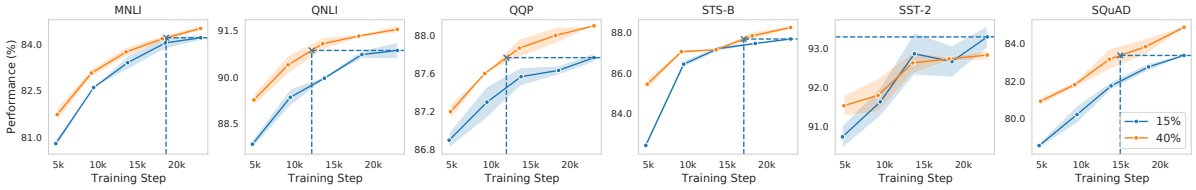

Figure 1: Downstream task development performance of `large` models trained with the efficient pre-training recipe, under masking rates of 15% and 40%. We highlight by the blue dotted line how long the 40% model takes to achieve the same performance as the 15% baseline; On QNLI and QQP, the 40% model achieved the same performance with almost half the training time.

make the pre-training faster by using a BERT-large architecture, a larger learning rate (2e-3), a larger batch size (4,096), a shorter sequence length (128)[4], and fewer training steps. We deviate from the 24hBERT with a few simple changes:

1. We adopt RoBERTa's BPE tokenizer (Sennrich et al., 2016; Liu et al., 2019) rather than BERT's tokenizer for it performs better in our preliminary experiments (see Appendix C).

2. Instead of adopting BERT's 80-10-10 token corruption strategy, we simply replace all the masked tokens with `[MASK]` by default. We find that the 80-10-10 corruption strategy does not perform better for most downstream tasks, as discussed in §7.

Following 24hBERT, we also do not perform next sentence prediction during pre-training, which was shown to hurt performance (Liu et al., 2019). We show hyperparameters for the efficient pre-training recipe and a comparison to other recipes (Devlin et al., 2019; Liu et al., 2019) in Appendix A. For models of different sizes, masking rates, and masking strategies, we follow the same recipe as our preliminary experiments show that it still performs the best.

We use fine-tuning downstream task performance as the measurement of how good the MLMs are, since fine-tuning is the predominant way to use pre-trained MLMs in downstream use. As evident from Table 1, pre-training metrics like perplexity do not correlate well with the downstream performance. We describe our downstream fine-tuning setting and hyperparameters in Appendix A.

[4]Izsak et al. (2021) only evaluate on GLUE tasks instead of SQuAD because of the short sequence length. We further train the model with 512 tokens for SQuAD in Table 1.

[5]For each task and each model size, *normalized performance* is calculated by $\frac{x-x_{15\%}}{\sigma}$ where $x_{15\%}$ is the performance of 15% masking rate and σ is the standard deviation across all masking rates. *Relative F1* is the F1 score subtracted by the 15% model F1.

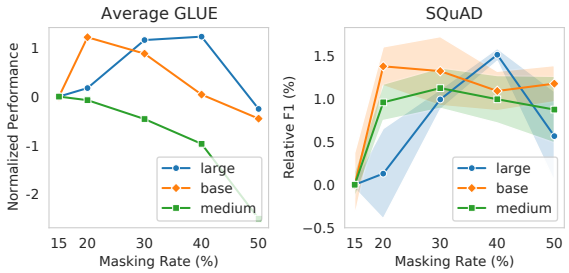

Figure 2: Impact of masking rates on different model sizes (`large`>`base`>`medium`).[5] We see that larger models favor larger optimal masking rates.

3 Larger Models Can Benefit From Higher Masking Rates

Devlin et al. (2019) choose the mysterious masking rate of 15%, for the belief that masking more leads to insufficient context to decode the tokens, and masking fewer makes the training inefficient, and this masking rate has been viewed as a constant across different model sizes. In this section, we train models of size `large` (354M parameters), `base` (124M parameters), and `medium` (51M parameters) for masking rates varying from 15% to 50%. The model configurations are listed in Appendix E.

Optimal masking depends on model sizes. The impact of masking rate across the model sizes is summarized by Figure 2, with detailed results given in Appendix E. We see that larger models possess higher optimal masking rates: on average, under the efficient pre-training recipe, `large` models take 40% as the optimal masking rate; `base` models take 20% and `medium` models take 15%. This shows that larger MLM models favor higher masking rates. We hypothesize that the additional capacity allows the large MLM to "handle" the more challenging task of predicting many tokens given less context.

Large models learn faster with 40% masking. We now compare the best performing masking rate 40% to the conventional 15% in more detail for our

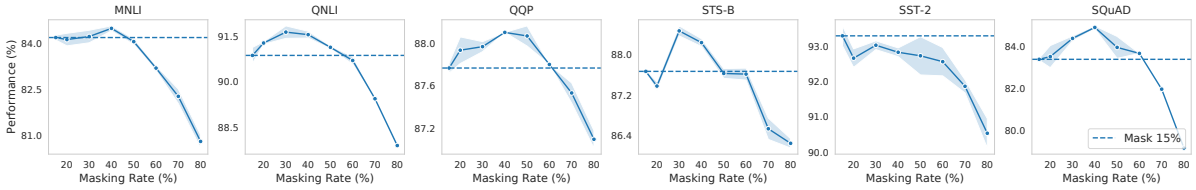

Figure 3: Impact of masking rates on `large` models with the efficient pre-training recipe. We see that on most tasks, higher masking rates outperform 15%. 40% is the optimal masking rate overall.

`large` model. First, we plot how the downstream task performance changes with different training steps in Figure 1. For most tasks, we see that 40% masking outperforms 15% consistently during the course of training, such that on QNLI and QQP, the 40% model can achieve the same performance as the 15% baseline with only half the training time. We also report the test results in Table 2, where again masking 40% outperforms 15% with our efficient pre-training recipe. However, the optimal masking rate can be task-dependent, as SST-2 performs better with 15% masking at the end of training. We acknowledge that the optimal masking rate may also depend on the training recipe. Since the efficient pre-training recipe uses a relatively small number of training steps, we explore training for over $4\times$ more steps, as well as training with a more expensive recipe from RoBERTa (Liu et al., 2019), and we find in Appendix D that using a 40% masking rate still performs well, achieving similar performance to the 15% masking rate. The experiments in the remaining sections of this paper are all based on large models.

4 MLMs in High-Masking Regimes

The success of masking 40% over 15% motivates us to explore what happens at even larger masking rates. Therefore, we pre-train additional `large` models with masking rates of up to 80%. We consider the question of what representations an MLM can learn with such limited input as the last masked sentence in Table 1, which is hard to decipher even for a human. While He et al. (2022) recently pioneered such high masking rates in the vision domain, and they reason that images are natural signals with heavy redundancy, while language is highly semantic and information-dense. To our knowledge, nobody has examined such high masking rates in masked language modeling before.

MLMs learn with extreme masking. We first confirm in Table 1 that the validation perplexity when pre-training with an 80% masking rate is extremely high (>1,000), which suggests that the MLM is unable to reconstruct corrupted inputs with independent token predictions. Therefore our setting differs from vision, where good reproductions are possible with high masking rates (He et al., 2022). Nevertheless, we find that MLMs can surprisingly still learn good representations: Figure 3 shows the performance of the models fine-tuned on a range of tasks, and we observe that pre-training with an 80% masking rate can retain 95% of fine-tuning performance, which is substantially better than fine-tuning from a random initialization, which is reported in Appendix B.

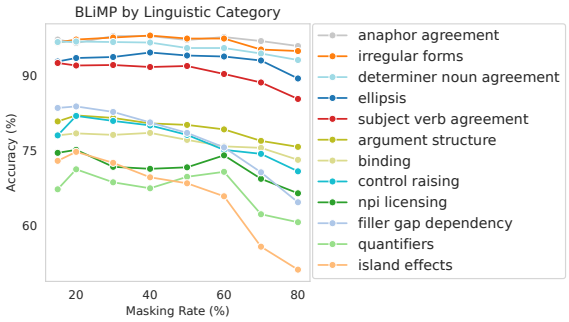

Figure 4: Evaluating our models on the BLiMP benchmark (Warstadt et al., 2020) using pseudo log-likelihood scoring (Salazar et al., 2020).

We hypothesize that MLMs at such high masking rates may be understood as a powerful skip-gram model (Mikolov et al., 2013), e.g., masking 80% of a 128 token sequence still learns skip-grams of length up to 26. Furthermore, when compared to the simple word2vec model, our Transformer models have access to positional information for each context token and prediction.

Analysis of linguistic probing. Besides downstream performance, we study the models' linguistic abilities by evaluating them on the BLiMP benchmark (Warstadt et al., 2020). We employ zero-shot pseudo log-likelihood scoring (Salazar et al., 2020), where a score is computed by masking each token individually, which is a greater distri-

	MNLI-m/mm	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	SQuAD
Masking 15%	84.2/83.4	90.9	70.8	73.5	**92.8**	88.8	**51.8**	87.3	88.0
★ Masking 40%	**84.7/84.0**	**91.3**	**70.9**	**75.5**	92.6	**89.8**	50.7	**87.6**	**89.8**

Table 2: The test results on the GLUE benchmark with `large` models, the efficient pre-training recipe (Izsak et al., 2021), and with 15% or 40% masking rates. For RTE, MRPC, and STS-B we fine-tune from the MNLI model following convention set by Phang et al. (2018). For SQuAD v1.1, we take the same setting as Table 1.

butional shift from higher masking rates. We show our results in Figure 4. We find that most linguistic phenomena are acquired evenly across masking rates from 15% to 60%, but they are still captured well by an MLM trained with 80% masking—which on average preserves 90% of the probing accuracy of the 15% model baseline. However, some categories such as filler gap dependencies and island effects show clear trends that performance deteriorates with higher masking rates—although it remains unclear to what extent such linguistic knowledge is required by downstream tasks in GLUE (Sinha et al., 2021). Overall, our results suggest that useful linguistic knowledge can be learned from a "patchy" training signal.

5 Masking Rates vs. Masking Strategies

Devlin et al. (2019); Liu et al. (2019) use uniform sampling for selecting which tokens to mask. Subsequent work showed that adopting more sophisticated masking strategies—such as span masking or PMI masking—can outperform uniform masking on a range of downstream tasks (Joshi et al., 2020; Levine et al., 2021). The argument for adopting advanced masking is that uniform masking enables models to exploit shallow local cues (Levine et al., 2021). An example is given by "`[MASK]` Kong": the model can easily predict "Hong" without using more context. However, all the previous studies used a constant 15% masking rate regardless of masking strategies, which raises the question of whether the conclusions still hold with a higher masking rate.

We experiment with multiple masking strategies as an additional factor for the optimal masking rate in `large` models. Figure 5 shows the results of uniform masking, T5-style span masking (Raffel et al., 2020)[6], and PMI masking (Levine et al., 2021) under masking rates from 15% to 40%. We see that (1) for all masking strategies, the optimal masking rates are higher than 15%; (2) the optimal masking rates for span masking and PMI masking are lower than that of uniform masking; (3) when all strategies adopt the optimal masking rates, the uniform masking achieves similar and even better results compared to the advanced strategies. We also remark that, when masking with 15%, simply increasing the masking rate can be a more effective way to increase performance on SQuAD than switching from uniform masking to another more advanced strategy. More fine-grained results with these masking strategies are included in Appendix E.

Interestingly, higher masking rates naturally increase the chance of masking neighbouring co-

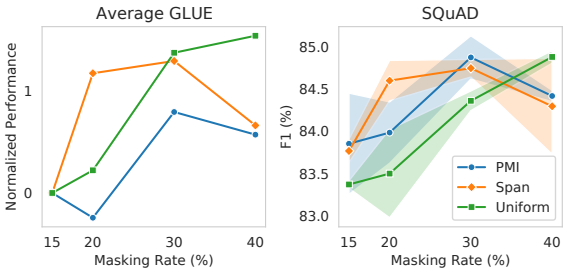

Figure 5: Performance of different masking strategies trained with different masking rates (efficient pre-training recipe, `large` models).

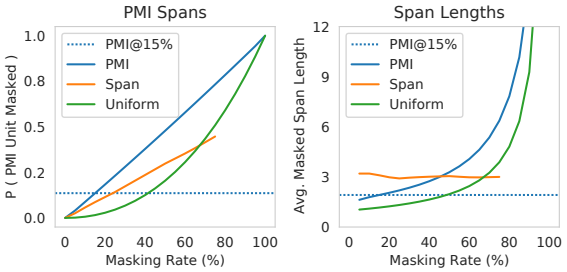

Figure 6: Higher masking rates increase the probability that an entire PMI span is masked (left) under different masking strategies. Uniform masking with a 40% rate masks as many PMI spans as regular PMI masking at 15%. Masks form longer spans for higher masking rates in uniform sampling, while the average length is fixed at 3 for T5-style span masking (which cannot be enforced for very high masking rates).

[6]Span maskings in Raffel et al. (2020) and Joshi et al. (2020) differ in sampling procedures and we follow Raffel et al. (2020) for implementation simplicity.

m_{corr}	m_{pred}	MNLI	QNLI	QQP	STS-B	SST-2
40%	40%	$84.5_{0.1}$	$91.6_{0.1}$	$88.1_{0.0}$	$88.2_{0.1}$	$92.8_{0.1}$
40%	20%	83.7↓	90.6↓	87.8↓	87.5↓	92.9↑
20%	20%	84.1↓	91.3↓	87.9↓	87.4↓	92.7↓
20%	40%	85.7↑	92.0↑	87.9↓	88.6↑	93.4↑
10%	40%	86.3↑	92.3↑	88.3↑	88.9↑	93.2↑
5%	40%	86.9↑	92.2↑	88.5↑	88.6↑	93.9↑

Table 3: Corruption vs. prediction. We take 40% masking as the baseline model (standard deviation reported), disentangle m_{corr} and m_{pred}, and manipulate each independently. The trend is clear: more prediction helps and more corruption hurts.

occuring tokens, similar to the effect of the advanced masking strategies. We consider the masked tokens over one epoch of training, and count the number of PMI n-grams (e.g., "Hong Kong") that were completely covered by different masking strategies. Figure 6 shows that raising the masking rate from 15% to 40% results in an 8-fold increase in the chance of masking a PMI n-gram under uniform masking and gives a value comparable to PMI masking at 15% masking rate. Similarly, higher masking rates also make the masked tokens form longer spans. However, at a given masking rate, uniform masking remains an easier task than span masking or PMI masking—it appears reasonable for uniform masking to admit a higher optimal masking rate for a given model capacity.

6 Understanding Masking As Corruption and Prediction

In this section, we analyze how masking rates affect the pre-training process of MLMs, through two distinct perspectives: task difficulty and optimization. We identify that the masking rate m determines two import aspects of the pre-training problem: the *corruption* rate m_{corr} and the *prediction* rate m_{pred}. m_{corr} is the proportion of tokens that are erased from the input sequence—typically by substituting `[MASK]`. m_{pred} is the proportion of tokens that the models predict, and each of those tokens contributes to the cross-entropy loss.

In Eq. (1), m_{corr} controls how much content is corrupted in $\tilde{x}$ compared to the original sentence x, and m_{pred} controls the number of predictions in the set $\mathcal{M}$. Usually, both the corruption and the prediction rates are tied to the masking rate, i.e., $m_{\text{corr}} = m_{\text{pred}} = m$, but they may impact representation quality differently.

m_{corr} controls task difficulty. Masked language modeling attempts to learn a conditional probability distribution over the vocabulary given the corrupted context $p(\cdot \mid \tilde{x})$ during pre-training. If a larger proportion of the input is corrupted, a token prediction is conditioned on fewer context tokens, making predictions harder and more uncertain.

m_{pred} affects optimization. Predicting more means the model learns from more training signals, so higher prediction rates boost the model performance. From another perspective, each prediction at each masked token leads to a loss gradient, which is averaged to optimize the weights of the model. Averaging across more predictions has a similar effect to increasing the batch size, which is proved to be beneficial for pre-training (Liu et al., 2019).

Experiments. In masked language modeling, both m_{corr} and m_{pred} are determined by the overall masking rate. To study how m_{corr} and m_{pred} affect the downstream performance independently, we design a simple ablation experiment to disentangle them:

1. If $m_{\text{pred}} < m_{\text{corr}}$, we mask m_{corr} of tokens and only make predictions on m_{pred} of the tokens. This can be implemented without additional cost. For example, with $m_{\text{corr}} = 40\%$ and $m_{\text{pred}} = 20\%$, we mask 40% and only predict on 20% tokens.

2. If $m_{\text{pred}} > m_{\text{corr}}$, we duplicate each sequence $\lceil \frac{m_{\text{pred}}}{m_{\text{corr}}} \rceil$ times and mask disjoint sets of m_{corr} of the tokens in different sequences. For example, with $m_{\text{corr}} = 20\%$ and $m_{\text{pred}} = 40\%$, for each sentence, we do twice 20% masking on different tokens and predict on all the masked tokens—this leads to a 20% corruption but a 40% prediction on each sequence. Note that this ablation takes $\lceil \frac{m_{\text{pred}}}{m_{\text{corr}}} \rceil$ times longer because we do multiple passes on every sequence, and is not efficient in practice.

Table 3 shows the ablation results with disentangled m_{corr} and m_{pred}. We see that (1) fixing the m_{corr} as 40%, lowering the m_{pred} from 40% to 20% results in a consistent drop on downstream tasks, showing that more predictions lead to better performance; (2) fixing the m_{pred} as 40%, lowering the m_{corr} leads to consistently better performance, suggesting that lower corruption rates make the pre-training task easier to learn and are better for pre-training. Though we see that the performance gain by lowering m_{corr} from 10% to 5% is much smaller than that by lowering m_{corr} from 40% to 20%, suggesting a diminishing marginal

	MNLI	QNLI	QQP	STS-B	SST-2
40% mask	$84.5_{0.1}$	$91.6_{0.1}$	$88.1_{0.0}$	$88.2_{0.1}$	$92.8_{0.1}$
+5% same	84.2↓	91.0↓	87.8↓	88.0↓	93.3↑
w/ 5% rand	84.5	91.3↓	87.9↓	87.7↓	92.6↓
w/ 80-10-10	84.3↓	91.2↓	87.9↓	87.8↓	93.0↑

Table 4: Impact of substituting masks with random/same tokens. "+5% same": do extra 5% same token predictions. "w/ 5% rand": use mask for 35% mask tokens and random tokens for 5% . "w/ 80-10-10": for the 40% masked tokens, 10% are same token predictions and 10% are random token corruptions.

return of reducing the corruption rate. (3) comparing $m_{\text{corr}} = 20\%, m_{\text{pred}} = 20\%$ and $m_{\text{corr}} = 40\%, m_{\text{pred}} = 40\%$, we see that the gain brought by more predictions transcends the drawback of more corruption, leading to better performance.

The ablation shows that when we tune the masking rate, we are tuning the corruption rate and the prediction rate together, which have antagonistic effects. The final outcome is decided by which rate weighs more—the model benefits from higher masking rates if the hindrance brought by high corruption is surpassed by the advantage from predicting more. Many factors may affect the balance between the two—for example, model sizes and masking strategies as we discussed in §3 and §5.

7 Revisiting BERT's Corruption Strategy

Devlin et al. (2019) suggest that it is beneficial to replace 10% of `[MASK]` tokens with the original token (*same token predictions*) and 10% with random tokens (*random token corruptions*). Since then, this 80-10-10 rule has been widely adopted in almost all the MLM pre-training work (Liu et al., 2019; Joshi et al., 2020; He et al., 2021). The motivation is that masking tokens create a mismatch between pre-training and downstream fine-tuning, and using original or random tokens as an alternative to `[MASK]` may mitigate the gap. With our corruption and prediction framework, we revisit the two kinds of mask replacements in the 80-10-10 rules and empirically verify whether they are beneficial to downstream performance.

Same token predictions. The loss from same token predictions is very small and should be regarded as an auxiliary regularization. Thus, same token predictions should neither count towards the corruption nor to the prediction—they do not corrupt the input and contribute little to learning.

Random token corruptions. Replacing with random tokens contribute to corruption and prediction rate, as the input is corrupted and the prediction task is non-trivial. In fact, we find that the loss is slightly higher on random tokens compared to `[MASK]`, as (1) the model needs to decide for all tokens whether the information at the input is from a corruption or not, and (2) predictions need to be invariant to large changes in the input embeddings.

Ablation experiments. We adopt the $m = 40\%$ model using only `[MASK]` replacements as the baseline, on top of which we add three models:

1. "+5% same": we mask 40% of tokens but predict on 45% of tokens. Adding same token predictions does not change m_{corr} or m_{pred}.

2. "w/ 5% random": we mask 35% of tokens and randomly replace another 5% of tokens, predicting on 40% in total.

3. "80-10-10": the original BERT recipe. Due to same token predictions, $m_{\text{corr}} = m_{\text{pred}} = 36\%$.

As shown in Table 4, we observe that same token predictions and random token corruptions deteriorate performance on most downstream tasks. The 80-10-10 rule performs worse than simply using all `[MASK]`—with the exception of SST-2, where same token predictions are beneficial. Overall, our results suggest that in the fine-tuning paradigm, the model can adapt to full, uncorrupted sentences, regardless of the use of alternative corruption strategies in pre-training. Therefore, we suggest to use only `[MASK]` for MLM pre-training. We also present an analysis based on information flow (Voita et al., 2019) in Appendix G.

8 Related Work

Masking rates and masking strategies. There exist a few works on studying the impact of masking rates, among which Liao et al. (2020) show that dynamically sampling the masking rate from 0% to 100% for each sequence can improve MLM's downstream performance as well as the ability as a generation model. On the other hand, masking strategies are heavily explored for both pre-training (Joshi et al., 2020; Raffel et al., 2020; Levine et al., 2021) and intermediate pre-training (Ye et al., 2021) without considering the effect of masking rates.

"Unrealistic" MLM training. A recent line of work shows that linguistically implausible MLM

objectives can achieve competitive or non-trivial downstream performance, e.g., training with shuffled word order (Sinha et al., 2021), with randomly generated sequences (Krishna et al., 2021), or predicting only the first character of masked tokens (Yamaguchi et al., 2021; Alajrami and Aletras, 2022). These studies echo our findings that even an "unrealistical" high masking rate can still lead to good downstream results.

Masking in other language models. Besides MLMs, there are other pre-training schemes, namely autoregressive language models (Radford et al., 2018; Brown et al., 2020) and sequence-to-sequence (seq2seq) language models (Raffel et al., 2020; Lewis et al., 2020). Similar to MLMs, seq2seq models corrupt text with a masking rate, but they predict with an autoregressive decoder and are fine-tuned in different ways; Song et al. (2019) also point out that masking rates control whether seq2seq models are closer to encoder-only MLMs (masking less) or decoder-only autoregressive LMs (masking more). Thus, we expect the masking rate studies in seq2seq models to draw a different conclusion from ours (Raffel et al., 2020; Tay et al., 2022b). Besides, Tay et al. (2022a) show that pre-training metrics are not correlated with downstream performance, echoing our findings that perplexity does not correlate with fine-tuning results.

ELECTRA (Clark et al., 2020) uses a smaller MLM to fill in 15% of the blanks and trains a model to distinguish whether a token was generated by the MLM or not. Despite the complicated training procedure, the main motivation of ELECTRA is to improve the training efficiency by predicting on 100% of tokens. Interestingly, we find that the corruption rate in ELECTRA becomes very low towards the end of training—the average corruption rate is roughly only 7%, but the replacements are "hard" negatives generated by the smaller MLM. We leave the study of its connection to corruption and prediction rates as future work.

Masking in other modalities. Recently, a number of works extend MLM training to images and videos and demonstrate strong pre-training results (He et al., 2022; Zhou et al., 2022; Feichtenhofer et al., 2022; Tong et al., 2022) . They adopt extremely high masking rates (e.g., 75% on images and 90% on videos) compared to their language counterparts, with the argument that images and videos are highly information redundant. Baevski et al. (2020) propose a similar style masked model in speech and adopt a masking rate of around 50%.

9 Conclusion & Discussion

In this work, we conduct a comprehensive study on the masking rates of MLMs. We discover that 15% is not universally optimal, and larger models should adopt a higher masking rate. We also find that masking strategies should be considered together with masking rates, and uniform masking needs a higher masking rate than more sophisticated masking strategies. We gain a better understanding of masking rates by disentangling them as corruption rates and prediction rates and analyze the 80-10-10 corruption strategy that are widely used in BERT models. Based on our findings, we discuss the implications of high masking rates and future directions of efficient MLM pre-training:

Implications on higher masking rates. A direct takeaway from our findings is that larger models may adopt higher masking rates for better sample efficiency. Figure 1 shows that a large model with 40% masking can achieve comparable results to a 15% baseline on several tasks with half the training time. Larger models also exhibit faster convergence for a given computational budget: Li et al. (2020) suggest it is more efficient to train larger models for fewer steps, as opposed to training smaller models for longer. This can be combined with higher masking rates for better sample efficiency.

Separating masked and unmasked tokens. The training efficiency can potentially benefit from encoding masked and unmasked tokens separately, where masked tokens use a much lighter-weight module. If a high masking rate is taken, this can significantly reduce the training cost due to the shorter input to the encoder. A similar approach has been explored by masked autoencoders in vision (He et al., 2022), where 75% of the input patches are masked and removed from the input of the heavy encoder to achieve a $4.1\times$ speedup. Recently, Liao et al. (2022) have applied these architectural improvements to natural language pre-training, and together with a high masking rate can accelerate MLM by a third of the pre-training budget.

Disentangling corruption and prediction. Models perform better when trained with lower corruption rates and higher prediction rates. However, in standard MLMs, those two factors are always tied

to the masking rate. Methods which can encode a sequence once and then efficiently predict many small sets of masks, for example by manipulating the attention, could substantially accelerate masked language modeling pre-training.

Limitations

(1) Our analysis of masking rates applies to a specific type of pre-training method, masked language modeling. We are also interested in studying masking rates in other pre-trained methods, e.g., seq2seq models and ELECTRA, and leave it for future work. (2) While we have shown how the optimal masking rate depends on model size and masking strategy, there may be additional factors, such as the vocabulary size, pre-training corpus or language family. In particular, our experiments focus on English, but languages with different structural and morphological features may have lower or even higher optimal masking rates, or rely more on advanced masking strategies. (3) We consider a well-established yet relatively small set of downstream tasks, which do not benchmark domain-specific knowledge or more advanced reasoning skills. (4) Due to the expensive nature of our pre-training experiments, we were not able to train multiple pre-trained models over multiple seeds. (5) Finally, our findings point out several promising directions but the paper primarily aims to study and understandthe impact of masking rates with respect to different factors. We leave exploring better architectures and methods for efficient pre-training to future work.

Ethical Considerations

Large language models can exhibit various kinds of stereotypes, as they capture societal biases encoded in the training data. These associations are not detected by standard GLUE or SQuAD evaluation. We do not expect that simple modifications of masking rates can make progress towards solving these problems. Language model pre-training is also computationally expensive, which comes at a significant environmental cost. Furthermore, it makes re-production and follow-up research difficult within an academic context. We reduce the computational requirements by following and promoting an efficient pre-training recipe and our findings point to future research for efficient MLM.

Acknowledgements

We thank Sadhika Malladi and the members of the Princeton NLP group for helpful discussion and valuable feedback. Alexander Wettig is supported by a Graduate Fellowship at Princeton University. This work is also supported by a Google Research Scholar Award.

References

Ahmed Alajrami and Nikolaos Aletras. 2022. How does the pre-training objective affect what large language models learn about linguistic properties? In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 131–147, Dublin, Ireland. Association for Computational Linguistics.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 12449–12460.

Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second PASCAL recognising textual entailment challenge.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *TAC*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

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

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment

challenge. In *the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional Transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *the Third International Workshop on Paraphrasing (IWP2005)*.

Christoph Feichtenhofer, Haoqi Fan, Yanghao Li, and Kaiming He. 2022. Masked autoencoders as spatiotemporal learners. *arXiv preprint arXiv:2205.09113*.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16000–16009.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations (ICLR)*.

Peter Izsak, Moshe Berchansky, and Omer Levy. 2021. How to train BERT with an academic budget. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 10644–10652.

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 of Computational Linguistics (TACL)*, 8:64–77.

Kundan Krishna, Jeffrey Bigham, and Zachary C. Lipton. 2021. Does pretraining for summarization require knowledge transfer? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3178–3189, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations (ICLR)*.

Yoav Levine, Barak Lenz, Opher Lieber, Omri Abend, Kevin Leyton-Brown, Moshe Tennenholtz, and Yoav Shoham. 2021. PMI-Masking: Principled masking of correlated spans. In *International Conference on Learning Representations (ICLR)*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Association for Computational Linguistics (ACL)*, pages 7871–7880.

Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joey Gonzalez. 2020. Train big, then compress: Rethinking model size for efficient training and inference of transformers. In *International Conference on Machine Learning (ICML)*, pages 5958–5968.

Baohao Liao, David Thulke, Sanjika Hewavitharana, Hermann Ney, and Christof Monz. 2022. Mask more and mask later: Efficient pre-training of masked language models by disentangling the [MASK] token. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1478–1492, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yi Liao, Xin Jiang, and Qun Liu. 2020. Probabilistically masked language model capable of autoregressive generation in arbitrary word order. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 263–274, Online. 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*.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

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.

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

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Technical report, OpenAI.

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. *The Journal of Machine Learning Research (JMLR)*, 21(140).

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 3505–3506.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Association for Computational Linguistics (ACL)*, pages 1715–1725.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning (ICML)*, pages 5926–5936.

Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. 2022a. Scale efficiently: Insights from pretraining and finetuning transformers. In *International Conference on Learning Representations*.

Yi Tay, Mostafa Dehghani, Vinh Q Tran, Xavier Garcia, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Neil Houlsby, and Donald Metzler. 2022b. Unifying language learning paradigms. *arXiv preprint arXiv:2205.05131*.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. 2022. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pretraining. In *Advances in Neural Information Processing Systems*.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. In *Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4396–4406.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations (ICLR)*.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association of Computational Linguistics (TACL)*, 7.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Atsuki Yamaguchi, George Chrysostomou, Katerina Margatina, and Nikolaos Aletras. 2021. Frustratingly simple pretraining alternatives to masked language modeling. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3116–3125, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Qinyuan Ye, Belinda Z. Li, Sinong Wang, Benjamin Bolte, Hao Ma, Wen-tau Yih, Xiang Ren, and Madian Khabsa. 2021. On the influence of masking policies in intermediate pre-training. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 7190–7202.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. 2022. Image BERT pre-training with online tokenizer. In *International Conference on Learning Representations (ICLR)*.

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.

A Experiment Setup

A.1 Pre-training

We implement our pre-training work based on `fairseq` (Ott et al., 2019). To further speed up pre-training, we integrate the `DeepSpeed` (Rasley et al., 2020) Transformer kernel for speedup.

We keep the other setting the same as the 24hBERT (Izsak et al., 2021), except that we use the RoBERTa tokenizer (Liu et al., 2019) and we do not adopt the 80-10-10 rule. We train our model on the English Wikipedia and BookCorpus (Zhu et al., 2015). We want to emphasize that using pre-layernorm (Shoeybi et al., 2019) is essential for the high learning rate in Izsak et al. (2021) to work. The hyperparameters for the efficient pre-training recipe are shown in Table 5. We train with 8 Nvidia GTX 2080 GPUs and use gradient accumulation to achieve the large batch sizes.

Hyperparameter	Efficient pre-training recipe
Peak learning rate	2e-3
Warmup proportion	6%
Batch size	4,096
Training steps	23,000
Sequence length	128
Architecture	`large`

Table 5: Our pre-training hyperparameter settings.

A.2 Downstream Task Evaluation

We fine-tune our model on the GLUE benchmark (Wang et al., 2019), including SST-2 (Socher et al., 2013), CoLA (Warstadt et al., 2019), MNLI (Williams et al., 2018), QNLI (Rajpurkar et al., 2016), RTE (Dagan et al., 2005; Bar Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009), MRPC (Dolan and Brockett, 2005), QQP[7] and STS-B (Cer et al., 2017), and the SQuAD v1.1 (Rajpurkar et al., 2016) dataset. For each dataset we run three random seeds and average the results. We apply grid search for the GLUE datasets, as shown in Table 6. For SQuAD, we use a learning rate of 1e-4, a batch size of 16, and train for 2 epochs. For both GLUE and SQuAD we use a linear scheduling for learning rates.

For all the results in the paper, we report accuracy for MNLI, QNLI, RTE, SST-2; we report F1 score for QQP, MRPC, and SQuAD; we report Matthew's correlation for CoLA and Spearman's correlation for STS-B.

[7] `https://www.quora.com/q/quoradata/`

Hyperparameter	MNLI, QNLI, QQP
Peak learning rate	{5e-5, 8e-5}
Batch size	32
Max epochs	{3, 5}
	RTE, SST-2, MRPC, CoLA, STS-B
Peak learning rate	{1e-5, 3e-5, 5e-5, 8e-5}
Batch size	{16, 32}
Max epochs	{3, 5, 10}

Table 6: Grid search hyperparameters for GLUE tasks.

For the SQuAD results in Table 1 and Table 2, we further train the models for 2300 steps (10% of the training) with a sequence length of 512, a learning rate of 5e-4, and a warmup rate of 10%. For other tables and figures, we present the SQuAD results without further pre-training, and the absolute numbers are lower because of the short pre-training sequence length. For some of the figures in the paper, we only show the results of MNLI, QNLI, QQP, STS-B, SST-2, and SQuAD due to limited space. Those tasks are selected because they have larger training set and the results are more reliable. We always show the development results in all our figures and tables except Table 2, where we report the test numbers for GLUE tasks.

B Different Masking Rates: Full Results

Table 7 shows the performance of 15%, 40% and 80% masked models on all GLUE tasks and SQuAD. We can see that 80% masking largely preserves the downstream performance and 40% outperforms 15% on most tasks.

C Tokenizer Comparison

Table 9 shows the performance of different tokenizers on downstream tasks. We see that on most tasks RoBERTa tokenizer is better than BERT tokenizer.

	MNLI-m/mm	QNLI	QQP	RTE
WordPieces	84.3/**84.9**	90.8	**88.2**	64.8
BPE	**84.5**/84.8	**91.6**	88.1	**67.0**
	SST-2	**MRPC**	**CoLA**	**STS-B**
WordPieces	92.5	75.5	56.6	**88.7**
BPE	**92.8**	**76.9**	**61.0**	88.2

Table 9: Comparison between BERT's uncased WordPieces tokenizer and RoBERTa's BPE tokenizer. Both models are `large` and trained with the efficient pre-training recipe with a 40% masking rate.

	MNLI-m/mm	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	SQuAD
Masking 15%	84.2/84.6	90.9	87.8	**67.3**	**93.3**	**77.0**	59.2	87.7	88.0
Masking 40%	**84.5/84.8**	**91.6**	**88.1**	67.0	92.8	76.9	**61.0**	**88.2**	**89.8**
Masking 80%	80.8/81.0	87.9	87.1	58.6	90.5	72.1	38.7	86.3	86.2
Random initialization†	61.5/61.2	60.9	70.7	49.6	80.0	45.4	11.9	17.5	10.8

Table 7: The development results on the GLUE benchmark with `large` models, the efficient pre-training recipe, and with 15%, 40%, or 80% masking rates. The SQuAD development results are attained with the same continuous training as in Table 1. Compared to the random initialization model, 80% masking rates clearly learn good representations for downstream tasks, despite having a very high perplexity. †: The random initialization models are trained with the same fine-tuning hyperparameters as pre-trained models, thus they could be undertrained.

	MNLI-m/mm	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B	SQuAD
Train longer with the efficient pre-training recipe									
Masking 15%	87.47/87.02	92.95	88.40	69.93	94.07	82.50	61.00	88.89	87.29
Masking 40%	86.63/86.83	93.13	88.40	68.87	94.67	79.50	61.23	89.60	87.16
Recipe from RoBERTa									
Masking 15%	87.40/87.23	93.04	88.43	67.53	94.13	80.80	59.80	90.05	90.72
Masking 40%	87.30/87.03	92.90	88.83	67.63	94.10	63.90	56.07	87.94	91.23

Table 8: Development results of 15% vs 40% masking with larger pre-training budget. We use the recipe from Table 3 in Liu et al. (2019), and the efficient pre-training recipe with more training steps. See Table 10 for hyperparameters.

D Longer Training

Hyperparameter	Train longer	RoBERTa
Peak learning rate	2e-3	7e-4
Warmup proportion	6%	6%
Batch size	4,096	2,048
Training steps	125,000	125,000
Sequence length	128	512

Table 10: Comparison between our longer pre-training recipes and a recipe from RoBERTa (Liu et al., 2019).

To see that how the different masking rates perform with longer training, we modify the efficient pre-training recipe for longer steps. We also experiment with a recipe used in the RoBERTa paper (Liu et al., 2019). Since the final RoBERTa models use more training data, we refer to the recipe used in RoBERTa's ablation in its Table 3. Table 10 shows the hyperparameters for the longer training, as well as a comparison to the RoBERTa's recipe. The major difference is that we train with much larger learning rate and only a sequence length of 128.

We train the models with 15% and 40% masking rates longer and evaluate them on downstream tasks. Figure 7 shows the results. We see that on most of the tasks, the trend that 40% is better than 15% still holds, though the 40% has a larger advantage when the training steps are limited.

We also train the model using a recipe from RoBERTa and present the results in Table 8. We see that (1) on most tasks 40% achieves comparable results compared to 15%; (2) our "train longer" results, which uses shorter sequences and larger learning rates, are comparable to the RoBERTa recipe results though with much shorter time.

E Results of Different Model Sizes and Masking Strategies

We show the configurations of different model sizes in Table 11. Figure 8 and Figure 9 show the results of the `base` model and the `medium` model, which serve as complementary materials for Figure 2.

Figure 10 shows the performance of uniform masking, T5-style span masking, and PMI masking on downstream tasks. This serves as a complementary material for Figure 5.

	`medium`	`base`	`large`
#Layers	8	12	24
#Attention heads	8	12	16
Hidden size	512	768	1024

Table 11: Configurations of different model sizes.

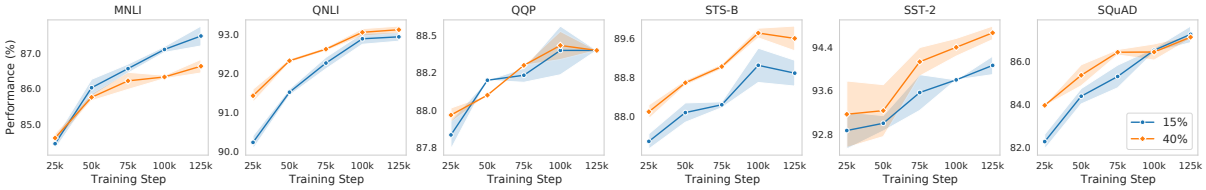

Figure 7: 15% vs 40% masking rates with `large` models and the efficient pre-training recipe but trained longer.

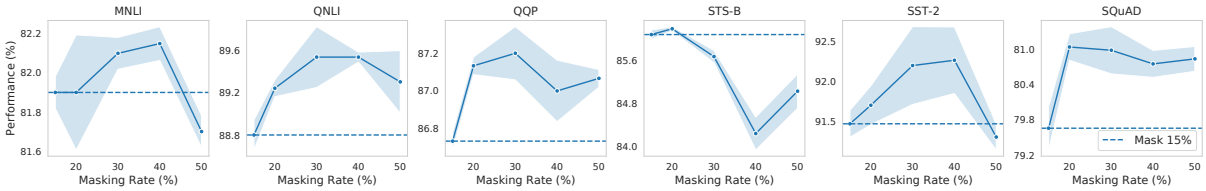

Figure 8: Results on selected downstream tasks with the `base` (124M parameter) model.

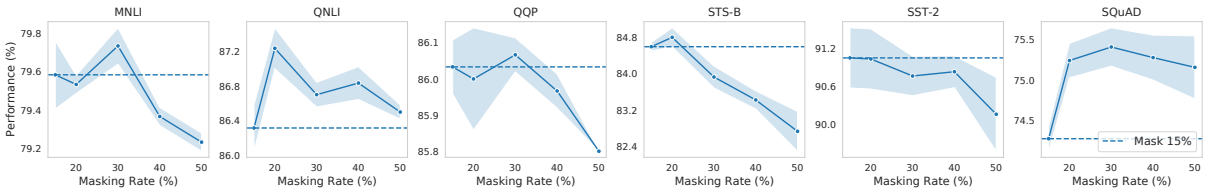

Figure 9: Results on selected downstream tasks with the `medium` (51M parameter) model.

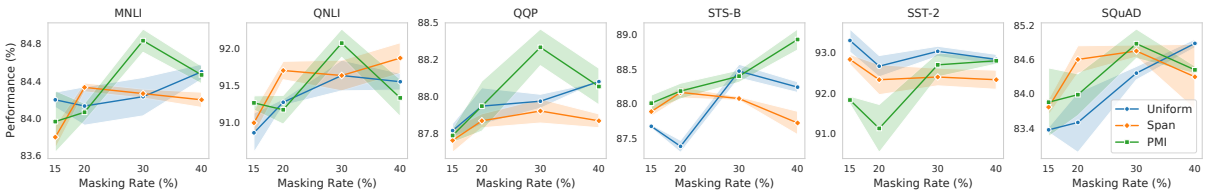

Figure 10: Comparison of different masking strategies on selected tasks. Models are trained with the efficient pre-training recipe, the `large` configuration, and several masking rates.

F Results on French MLM

To validate our conclusions in a new setting, we conduct experiments on MLM on a corpus in French. Similar to Izsak et al. (2021), we pre-train on 2020 French Wikipedia and fine-tuned on French XNLI. We report accuracy averaged over 4 seeds, and make the observation that 40% is better than 15%.

	XNLI-fr	
	valid	test
Masking 15%	78.3	77.3
Masking 40%	78.9	77.5

Table 12: We pre-train on 2020 French Wikipedia and fine-tuned on French XNLI. We report accuracy averaged over 4 seeds.

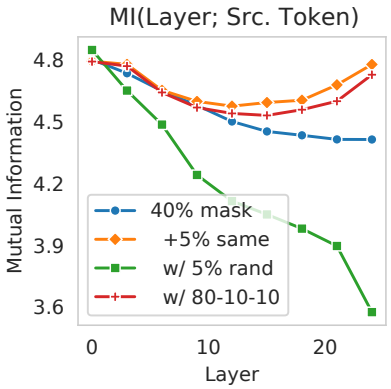

Figure 11: Mutual information between an input token and its intermediate representations for four different corruption strategies. See Table 4 for details on models.

G Information Flow Analysis

To visualize the effect of these corruption strategies (the 80-10-10 rule), we follow Voita et al. (2019)'s

analysis of measuring mutual information between an input token and its intermediate representations. Figure 11 shows that each model initially loses some information about the source token while acquiring information from the surrounding context. Using same token predictions during pre-training leads to a "reconstruction" stage in the last few layers, as observed by Voita et al. (2019), whereby information about the source token is restored from the context. However, this second stage is not present when same token predictions tokens are ablated: the [MASK]-only baseline propagates contextual features only—and no reconstruction occurs. This is more pronounced with random token corruption, where source information (that was less reliable during pre-training) is lost at a greater rate. One consequence is that information about the input tokens can be more easily extracted when pre-training with same token predictions. However, the reconstruction of the source tokens does not appear to be as important in the fine-tuning setting, as shown in our experiments in Table 4.