Better Together: Jointly Using Masked Latent Semantic Modeling and Masked Language Modeling for Sample Efficient Pre-training

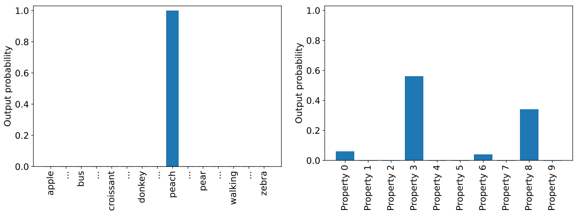
Gábor Berend
University of Szeged
2 Árpád tér, Szeged, Hungary
berendg@inf.u-szeged.hu

Abstract

In this paper, we demonstrate the benefits of jointly using Masked Latent Semantic Modeling (MLSM) and traditional Masked Language Modeling (MLM) as the pre-training objective of masked language models. The core idea behind MLSM is to modify the pre-training objective in a way which ensures that the language models predict a (latent) semantic distribution for the masked tokens – instead of outputting their exact identity as in MLM. Language models pre-trained with MLSM behave more favorable in terms of fine-tuneability towards downstream tasks, however, their performance lags behind MLM pre-trained language models in evaluations that investigate the linguistic capabilities. In an attempt to combine the strengths of the two different pre-training paradigms, we propose their joint use in a multi-task learning setting. Our evaluations that we performed using the BabyLM evaluation framework (Warstadt et al., 2023) demonstrate the synergistic effects of the joint use of the two different kinds of pre-training objectives.

1 Introduction

Albeit being effective and easy to implement in practice, the highly stochastic batch-based masked language modeling (MLM) objective frequently used for pre-training language models, such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), is not sample efficient and works in a rather unnatural way from a human cognitive perspective. This is caused by the fact that traditional MLM expects the neural models to recover the exact identity of the masked (sub)words within an input sequence. In an attempt to overcome the unnaturalness of MLM, (Berend, 2023) has recently proposed masked latent semantic modeling (MLSM), a sample efficient alternative to traditional masked language modeling.



(a) MLM objective

(b) MLSM objective

Figure 1: Comparisons of the probability distributions used in MLM (a) and MLSM (b) pre-training.

MLSM differs from MLM in that its objective is to recover the semantic distribution of masked (sub)tokens over an unsupervised inventory of latent semantic properties — as opposed to that of a one-hot distribution over the entire vocabulary of the language model. This kind of pre-training is arguably more plausible from a human cognitive perspective, i.e., traditional MLM acts as if there was a single proper substitute for a special `[MASK]` token (the one that got masked), whereas from a human perspective multiple viable tokens – tokens that share some common semantic properties – can substitute a masked token.

For instance, in the sentence *'She picked a delicious* `[MASK]`.', human subjects would agree that any word referring to an edible concept is a viable substitute for the last word of the sentence. In Figure 1, we illustrate the different kinds of outputs that the MLM (Figure 1a) and the MLSM (Figure 1b) objectives could produce for some masked token such as the one in the above example.

Even though (Berend, 2023) has demonstrated the improved sample efficiency of MLSM, language models pre-trained with it perform poorly in evaluations that test the linguistic capabilities of language models. In this paper, we extend the results from (Berend, 2023) in several important aspects. On the one hand, – instead of using a medium-sized BERT model – we pre-train base-sized DeBERTa (He et al., 2021) models, illustrating that the MLSM pre-training objective gener-

Proceedings of the 27th Conference on Computational Natural Language Learning:
Volume 2: The BabyLM Challenge, pages 298–307
December 6-7, 2023 ©2023 Association for Computational Linguistics

alizes across different model types and sizes. On the other hand, we investigate the added value of a multi-task learning setting during pre-training, in which the use of MLSM objective is coupled with traditional MLM. Our empirical results show vast improvements in the performance of the pre-trained language models using the joint objective. We release our source code[1] and pre-trained models that we created using the strict[2] and strict-small[3] datasets provided as part of the BabyLM shared task (Warstadt et al., 2023).

2 Methodology

In this section, we introduce the pre-training training objectives that we conducted experiments with.

2.1 Standard Masked Language Modeling

During MLM pre-training, we expect the masked language model to output a probability distribution over its entire vocabulary and the objective is to return one-hot distributions corresponding to the actually masked token, similar to what is illustrated in Figure 1a. The loss function for this kind of pre-training is the categorical cross entropy.

2.2 Knowledge distillation (KD)

During knowledge distillation (KD), we expect the language model to output such a probability distribution over its entire vocabulary that tries to mimic the output distribution of viable masked token substitutes, produced by another language model that is (partially) pre-trained using the standard MLM objective. This setting, hence, is basically a two phase pre-training, in which the first phase is a regular pre-training, followed by a knowledge distillation phase, during which we calculate the Kullback-Leibler divergence between the probability distribution outputted by the language model from the first phase and the model that is being trained.

In this two phase setting, we have the option to reinitalize the model weights, or to make a copy of the (partially) pre-trained model from the first phase, and start KD pre-training with the weights of the MLM pre-trained model in a transfer learning setting. As our preliminary experiments suggested that this latter form of continued pre-training is more beneficial, we opted for that variant of KD.

2.3 Masked Latent Semantic Modeling

We also utilize Masked Latent Semantic Modeling (Berend, 2023). MLSM is based on an efficient *unsupervised* method for determining the context-sensitive latent semantic distribution of *any* token. We use this as the target distribution that the language model needs to recover during a pre-training as illustrated in Figure 1b.

MLSM is similar to knowledge distillation in that it also relies on a (partially) pre-trained model, however, the mechanism in which it gets utilized differs rather substantially. The partially pre-trained model was not only used for providing the training signal, but also for initializing the weights of MLSM pre-trained models.

The MLSM approach is based on the observation that (sub)tokens with overlapping semantic content tend to have an overlapping set of non-zero coordinates in their sparse contextualized representation, which can be obtained by performing sparse coding on the hidden representations of transformer architectures (Berend, 2020). We incorporate this property of sparse token representations into pre-training, i.e., we devise such distributions of latent semantic properties of masked tokens that are based on the sparsity structure of the sparse representations during the second phase of pre-training.

Suppose that the language model from the first phase of pre-training produces hidden vectors $\boldsymbol{h}^{(l)} \in \mathbb{R}^d$ by its lth layer for a particular token within its context. We then construct a collection of hidden representations as $\boldsymbol{H}^{(l)} \in \mathbb{R}^{d \times n}$, and, as a preparatory step for the second phase of pre-training, we jointly optimize for a dictionary matrix $\boldsymbol{D} \in \mathbb{R}^{d \times k}$ and $\boldsymbol{\alpha}_{\boldsymbol{H}^{(l)}} \in \mathbb{R}^{d \times n}$, such that

$$\min_{\boldsymbol{D}, \boldsymbol{\alpha}_{\boldsymbol{H}^{(l)}}} \frac{1}{2} \|\boldsymbol{H}^{(l)} - \boldsymbol{D}\boldsymbol{\alpha}_{\boldsymbol{H}^{(l)}}\|_F^2 + \lambda \|\boldsymbol{\alpha}_{\boldsymbol{H}^{(l)}}\|_1,$$

where the norm of the columns vectors in $\boldsymbol{D}$ do not exceed 1, and the sparse linear coefficients in $\boldsymbol{\alpha}$ are non-negative, with the regularization coefficient λ controlling the sparsity level of $\boldsymbol{\alpha}$.

Once the dictionary matrix $\boldsymbol{D}$ is determined, we can obtain sparse contextualized representation for any token described by $\boldsymbol{h}^{(l)}$ via solving

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^k_{\geq 0}} \frac{1}{2} \|\boldsymbol{h}^{(l)} - \boldsymbol{D}\boldsymbol{\alpha}\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_1. \qquad (1)$$

As such, the determination of (1) can provide useful signal during the second phase of pre-training, i.e., by determining the sparse $\boldsymbol{\alpha}$ representation for

[1] https://github.com/SzegedAI/MLSM

[2] https://huggingface.co/SzegedAI/babylm-strict-mlsm

[3] https://huggingface.co/SzegedAI/babylm-strict-small-mlsm

the token which was assigned $\boldsymbol{h}^{(l)}$ by the language model from the first phase of pre-training, we can obtain its latent semantic profile via investigating its non-zero coefficients. Due to the non-negativity of $\boldsymbol{\alpha}$, it can be conveniently transformed into a probability distribution of semantic profiles via ℓ_1-normalization, each coordinate corresponding to a (latent) semantic property as illustrated in Figure 1b.

Similar to KD pre-training, MLSM also employs the Kullback-Leibler divergence as its objective for comparing the expected semantic distribution and the model output. A major difference between KD and MLSM though is that for the former, the domain of the target distribution is the entire vocabulary, whereas for MLSM, there are k many latent semantic properties to consider.

2.4 Joint training objectives

We relied on standard MLM on its own as one of our baseline approaches, as well as in conjunction with other pre-training objectives, in order to assess its added value as a joint self-supervised pre-training task. In the case MLM was used as an additional pre-training task, the losses of the different pre-training paradigms were added together and backpropagation was performed over the joint loss. When using MLM as an additional loss, we add the `+MLM` suffix to the pre-training approach that we augment it with. For instance, `KD+MLM` refers to such a pre-trained model that we obtained by relying on the joint objective of knowledge distillation and MLM.

3 Experiments and results

We performed our experimental evaluation based on the BabyLM Challenge environment (Warstadt et al., 2023), the goal of which is to provide a unified framework for pre-training language models based on moderate amounts of texts, inspired by children language acquisition (Saffran et al., 2001; Gilkerson et al., 2017; Dupoux, 2018). The size and the contents of the pre-training dataset released as part of the BabyLM Challenge is guided by the amount and types of texts children are typically exposed to by reaching preadolescence.

That is, the size of the pre-training corpus is limited in either 100 million (strict) or 10 million (strict-small) tokens, and the released text is mostly composed of transcribed speech. The concrete subcorpora of the challenge are the CHILDES (Macwhinney, 2000), dialogue portion of the British National Corpus (BNC), Children's Book Test (cbt; Hill et al., 2016), Children's Stories Text Corpus, Standardized Project Gutenberg Corpus (Gerlach and Font-Clos, 2020), OpenSubtitles (Lison and Tiedemann, 2016), QCRI Educational Domain Corpus (qed; (Abdelali et al., 2014)), Wikipedia, Simple Wikipedia and the Switchboard Dialog Act Corpus (Stolcke et al., 2000).

The evaluation framework contains a collection of supervised fine-tuning and zero-shot evaluations for assessing the utility and the linguistic capabilities of the pre-trained language models.

3.1 Training a tokenizer

As the goal of the BabyLM Challenge is to create an environment in which language models are not exposed to colossal amounts of pre-training text, all components of the trained language models conformed to the standardized pre-training data. To this end, we first trained a unigram tokenizer (Kudo, 2018) over the corresponding BabyLM strict/strict-small dataset, that comprised of roughly 100/10 million (whitespace separated) tokens. The vocabulary size we employed is 25000.

As increased vocabulary size can potentially yield better downstream performance (e.g., one of the potential reasons why RoBERTa (Liu et al., 2019) often performs better than BERT (Devlin et al., 2019) is due to its increased vocabulary size), we also attempted to train a unigram tokenizer with 50000 subtokens as well. Our preliminary results, however, showed vastly degraded performance for the increased vocabulary size.

For this reason, we continued our experiments with the tokenizers with 25000 cased entries, which was likely more beneficial compared to the one with twice the number of subtokens, as the training corpus itself was intentionally limited in its size, and the increased vocabulary was too large for the relatively small number of unique tokens in the pre-training corpora.

3.2 Pre-training

We used almost identical hyperparameters to (Berend, 2023). That is, we employed a batch size of 128 and a gradient accumulation over 8 batches, yielding an effective batch size of 1024. The learning rate for pre-training was set to $1\text{e}{-4}$ with linear scheduling.

We employed the kind of two-phase pre-training introduced earlier in Section 2, i.e., we first pre-

	KD	KD+MLM	MLM	MLSM	MLSM+MLM
anaphor agreement	0.801	0.893	0.801	0.476	0.718
argument structure	0.760	0.797	0.779	0.700	0.762
binding	0.655	0.645	0.660	0.680	0.654
control raising	0.763	0.771	0.766	0.706	0.770
determiner noun agreement	0.969	0.969	0.969	0.847	0.969
ellipsis	0.908	0.936	0.924	0.690	0.930
filler gap	0.826	0.850	0.850	0.714	0.850
hypernym	0.492	0.510	0.480	0.503	0.480
irregular forms	0.850	0.907	0.949	0.794	0.948
island effects	0.669	0.754	0.782	0.629	0.773
npi licensing	0.732	0.781	0.759	0.628	0.768
qa congruence easy	0.625	0.672	0.688	0.438	0.688
qa congruence tricky	0.358	0.394	0.467	0.442	0.424
quantifiers	0.733	0.752	0.768	0.484	0.754
subject aux inversion	0.929	0.949	0.951	0.808	0.951
subject verb agreement	0.893	0.904	0.893	0.764	0.903
turn taking	0.557	0.604	0.643	0.571	0.611
Average	0.736	0.770	0.772	0.640	0.762

(a) strict

KD+MLM	MLM	MLSM+MLM
0.880	0.829	0.831
0.765	0.739	0.737
0.684	0.661	0.676
0.737	0.728	0.757
0.948	0.933	0.939
0.830	0.819	0.827
0.781	0.768	0.777
0.477	0.495	0.479
0.910	0.896	0.902
0.630	0.650	0.685
0.719	0.712	0.743
0.734	0.688	0.703
0.370	0.364	0.333
0.706	0.728	0.733
0.863	0.827	0.830
0.852	0.816	0.825
0.525	0.521	0.521
0.730	0.716	0.724

(b) strict-small

Table 1: BLiMP results for the models pre-trained on the strict (a) and the strict-small (b) corpora.

trained a model using standard MLM, then used this model for initializing the second-phase model, the pre-training objective of which can potentially differ from MLM. We performed 20,000 and 80,000 update steps during the first and second phases, respectively.

As such, we had a total of 100,000 update steps, which together with the fact that we had an effective batch size of 1024, means that we considered approximately 100,000,000 sequences during pre-training. This resulted in 17 and 166 epochs when using the strict and the strict-small pre-training corpora, respectively. We performed pre-training on NVIDIA A6000 or V100 GPUs (depending on their availability). One pre-training took approximately 5 days to finish.

For the strict scenario, we report results when using the different pre-training paradigms on their own and in conjunction with MLM. As our experiments revealed a superior performance for the joint pre-training with MLM, we only consider those models that jointly use one of the pre-training paradigms and MLM during the second phase of pre-training for the strict-small case.

When applying MLSM, we set the number of latent semantic properties to one tenth of the size of the vocabulary, i.e., we had $k = 2500$. For the joint objectives (KL+MLM and MLSM+MLM), we weighted the two loss terms equally by simply adding the two loss terms together. Investigating different weighting of the MLM term could have been an interesting, but computationally demanding ablation experiment to conduct.

3.3 Quantitative evaluation

We next report our experimental results towards zero-shot (§3.3.1) and fine-tuning (§3.3.2) evaluation, using the BabyLM evaluation framework.[4]

3.3.1 Zero-shot results on BLiMP

The BabyLM framework uses the BLiMP dataset (Warstadt et al., 2020a) for assessing the linguistic capabilities of language models. BLiMP contains English sentence pairs that differ in their linguistic acceptability regarding a variety of grammatical concepts and the task is to select the correct sentence based on the pre-trained model.

To decide which sentence is linguistically more acceptable, the pseudo-log-likelihood (PLL; Salazar et al., 2020) scores of the sentences are calculated, and the sentence with the higher PLL is considered grammatically acceptable. The BabyLM evaluation framework focuses on 17 grammatical phenomena, the results of which are included in Table 1.

Table 1a reveals that the MLSM pre-trained model performs poorly on BLiMP. This is not surprising, as PLL is based on the predictions over the vocabulary of the model, however, MLSM totally neglect the kind of objective that is related to the vocabulary of the model, making the PLL values

[4] https://github.com/babylm/

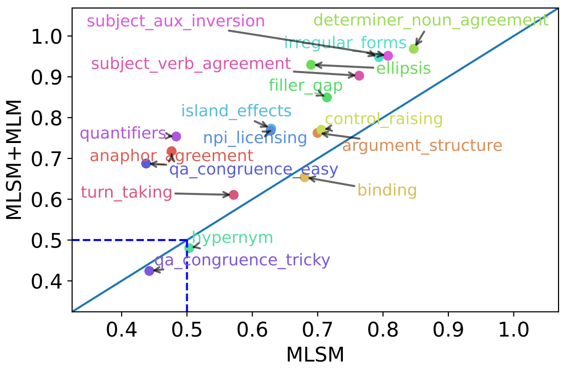


Figure 2: Pairwise comparison of BLiMP task performances between the MLSM (x-axis) and the MLSM+MLM (y-axis) pre-trained models.

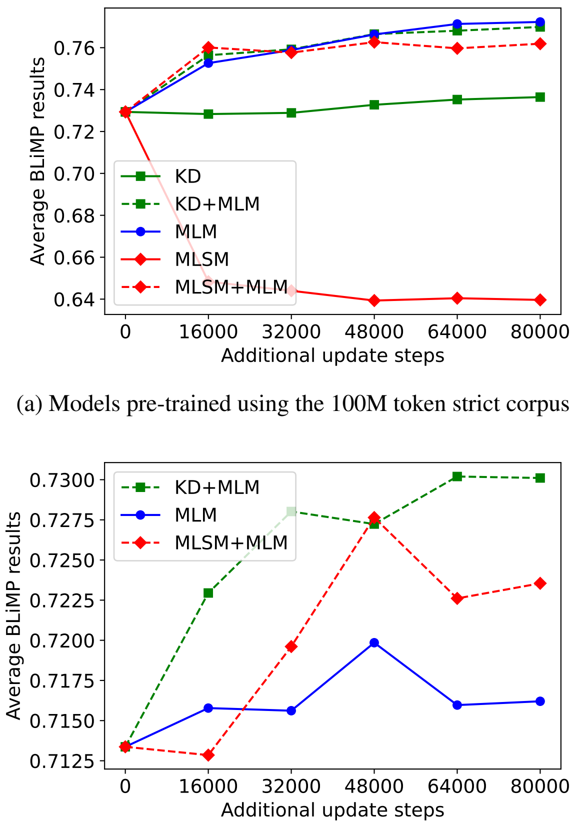
calculated by MLSM-only models less useful for approximating linguistic acceptability.

The model pre-trained with the joint MLSM objective (MLSM+MLM), however, performs 0.122 points better on average (0.640→0.762), nearly as good as the model pre-trained with MLM alone (0.772). The additional use of MLM also improves the BLiMP performance of knowledge distillation by 0.034 points on average (0.736→0.770).

Table 1b reveals that when using the reduced amount of strict-small pre-training corpus, the MLSM+MLM pre-trained model in fact outperforms the purely MLM pre-trained model variant.

We depict the added value of using the joint MLSM+MLM objective over the MLSM only objective when conducting pre-training on the 100 million token strict corpus in Figure 2. Each of the 17 sub-task is visualized by a point in the figure, with its x and y coordinates displaying the performance of the pre-trained model that was based on the MLSM and MLSM+MLM objectives. The dashed line indicates chance performance, and the diagonal line helps in identifying the added value of joint pre-training, i.e., the further away a point above the diagonal line is, the bigger positive impact the joint pre-training had towards the evaluation on the subtask represented by the given point.

During the second phase of pre-training, we evaluated intermediate checkpoints. Figure 3a and Figure 3b illustrates the average BLiMP performance of the models pre-trained with varying strategies and at different readiness levels for using the strict and the strict-small pre-training corpora, respectively. The x-axis indicates the number of additional update steps performed during the second phase of the pre-training.



(a) Models pre-trained using the 100M token strict corpus

(b) Models pre-trained using the 10M token strict-small corpus

Figure 3: Average BLiMP performances as a function of the number of update steps performed in the second phase of pre-training.

The MLSM curve in Figure 3a shows that the masked language modeling capabilities of an MLSM-only pre-trained model fade out quickly, as the average BLiMP performance drops drastically already at the first investigated checkpoint, i.e., at 16,000 additional MLSM update steps performed on a model that had gone through 20,000 steps of first phase MLM pre-training.

Figure 3a further reveals that there is a large performance gap between the MLSM and MLSM+MLM pre-trained models at every checkpoint, with the performance of MLSM+MLM being nearly as good or better than that of the purely MLM pre-trained model. As the size of the pre-training corpus gets reduced from 100 million to 10 million tokens, the average BLiMP performance of the alternatively pre-trained models becomes favorable compared to the MLM-only models as it is illustrated in Figure 3b.

3.3.2 Fine-tuning results

The BabyLM evaluation framework also includes supervised learning tasks from the GLUE (Wang

	KD	KD+MLM	MLM	MLSM	MLSM+MLM
BoolQ	**0.6943**	0.6885	0.6936	0.6857	0.6826
CoLA	0.4551	0.4687	**0.4962**	0.4758	0.4854
MNLI	0.7620	0.7669	0.7695	0.7558	**0.7704**
MNLI-mm	0.7641	0.7761	0.7779	0.7687	**0.7808**
MRPC	0.8263	0.8406	**0.8496**	0.8325	0.8339
MultiRC	0.5578	0.6114	0.6238	**0.6309**	0.5983
QNLI	0.8350	0.8409	**0.8447**	0.8427	0.8438
QQP	0.8366	0.8451	**0.8492**	0.8421	0.8428
RTE	0.5985	**0.6010**	**0.6010**	**0.6010**	0.5808
SST2	0.8907	0.8922	0.8927	**0.8952**	0.8907
WSC	0.6024	0.5964	0.5843	0.6024	**0.6054**

(a) strict

	KD+MLM	MLM	MLSM+MLM
BoolQ	**0.6843**	0.6729	0.6670
CoLA	0.3889	0.3794	**0.4171**
MNLI	0.7503	0.7426	**0.7542**
MNLI-mm	0.7506	0.7527	**0.7535**
MRPC	0.7645	**0.7766**	0.7653
MultiRC	0.580	**0.6076**	0.5676
QNLI	0.8205	**0.8261**	0.8237
QQP	0.8343	0.8346	**0.8351**
RTE	0.5404	**0.5556**	0.5202
SST2	0.8903	**0.8937**	0.8917
WSC	0.5813	0.5964	**0.6084**

(b) strict-small

Table 2: (Super)GLUE results for the models pre-trained on the strict (a) and the strict-small (b) corpora. Metrics are reported as accuracy, except for CoLA (Matthew Correlation Coefficient), MRPC (F1) and QQP (F1).

et al., 2019b) and SuperGLUE (Wang et al., 2019a) benchmarks and selected subtasks of MSGS (Mixed Signals Generalization Set; Warstadt et al., 2020b). The original datasets are filtered to those cases that include words that are present at least twice in the 10 million token strict-small training corpus. Unless stated otherwise, we report performance metrics in the form of accuracy.

We made no modifications in the hyperparameters of the official evaluation framework, apart from reducing the batch size from 64 to 32, which was necessary for avoiding out-of-memory error on the NVIDIA 2080Ti GPUs that accommodated our fine-tuning experiments. In order to account for the high variability in fine-tuning results, we repeated all experiments involving fine-tuning four times with different random seeds and report the average of the scores that we obtained. Due to the computational need of fine-tuning, we only evaluated the intermediate checkpoints at the 20%, 60% and 100% readiness levels, i.e., after 16000, 48000 and 80000 additional second phase pre-training steps.

(Super)GLUE Vocabulary-filtered versions of 11 different subtasks from (Super)GLUE are included in the BabyLM evaluation environment. The individual results obtained by the differently pre-trained DeBERTa models are listed in Table 2. Fine-tuning MLSM+MLM models again yielded better results compared to the MLSM models, however, the performance gap is not that pronounced as it was for BLiMP. The average fine-tuning performance of MLSM+MLM pre-trained model is on par with the one that got pre-trained with traditional MLM considering the models pre-trained over the 100 million corpus.

Figure 4 displays the fine-tuning performance of the intermediate model checkpoints of second phase pre-training. Figure 4a reveals that when using the 100 million token training corpus, the intermediate checkpoints of the MLSM+MLM and MLM models have similar fine-tuning performances averaged over the (Super)GLUE tasks, with a slight advantage towards MLSM+MLM.

For the smaller training corpus in Figure 4b, the advantage of MLSM+MLM pre-trained model is more notable, confirming that jointly using MLSM with MLM offers better sample efficiency.

MSGS MSGS (Warstadt et al., 2020b) is a sentence classification challenge set that contains training instances towards different linguistic categories and surface form features of sentences. Control tasks are 'regular' training and evaluation splits in the sense that there is no purposefully encoded spurious correlation in the training dataset that is not present in the test set. The challenge tasks, however, are designed with the intention of conflating two properties with each other in the training set in a way that the given relation do not hold for the test instances. This way, one can measure to what extent the model was able to learn and rely on the actual target contept to be learned as opposed to the deliberately included surface level spruious correlation in the training data.

Table 3 contains the results for the control tasks as well as for the challenging cases with the purposefully malignant training data in which a surface form characteristic goes along with the linguistic properties to be tested. The different kinds of test

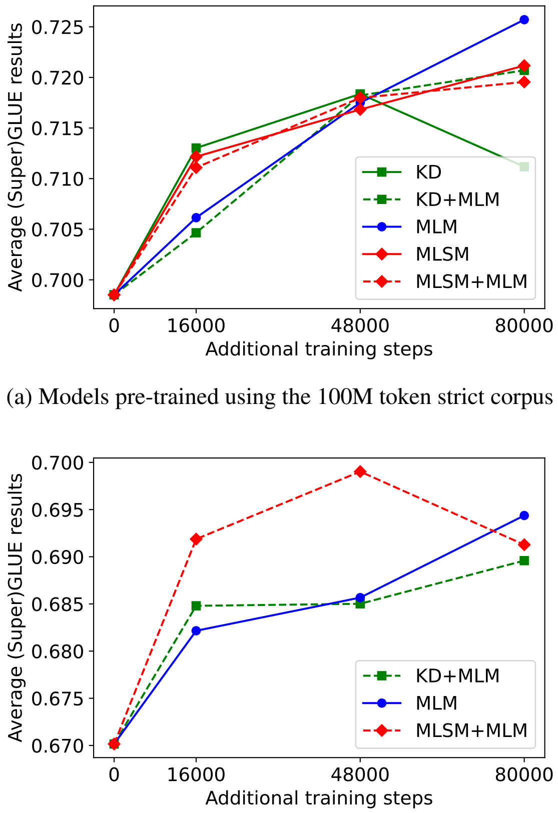
	KD	KD+MLM	MLM	MLSM	MLSM+MLM
CR (control)	0.7521	0.7609	0.7739	0.7842	0.7940
LC (control)	1.0	1.0	1.0	1.0	1.0
MV (control)	0.9999	0.9994	0.9997	0.9996	0.9995
RTP (control)	0.6905	0.8738	0.9117	0.9344	0.8785
SC (control)	0.7603	0.7786	0.7940	0.7130	0.7794
CR_LC	-0.4572	-0.6195	-0.6733	-0.6766	-0.5380
CR_RTP	-0.7686	-0.6571	-0.7805	-0.6051	-0.7613
MV_LC	-0.5329	-0.3928	-0.7954	-0.8370	-0.8558
MV_RTP	-0.0097	0.0729	-0.2217	-0.1047	-0.0385
SC_LC	-0.2849	-0.2673	-0.3011	-0.3087	-0.3223
SC_RP	-0.5758	-0.5601	-0.5039	-0.5346	-0.5173

(a) strict

	KD+MLM	MLM	MLSM+MLM
CR (control)	0.6311	0.6872	0.7351
LC (control)	1.0	1.0	1.0
MV (control)	0.9988	0.9956	0.9985
RTP (control)	0.8579	0.9857	0.8963
SC (control)	0.6657	0.6829	0.7845
CR_LC	-0.2261	-0.4080	-0.0729
CR_RTP	-0.6850	-0.8230	-0.6516
MV_LC	-0.9055	-0.9522	-0.9465
MV_RTP	-0.2882	-0.5484	-0.3947
SC_LC	-0.0300	-0.2715	-0.1664
SC_RP	-0.5290	-0.5681	-0.5275

(b) strict-small

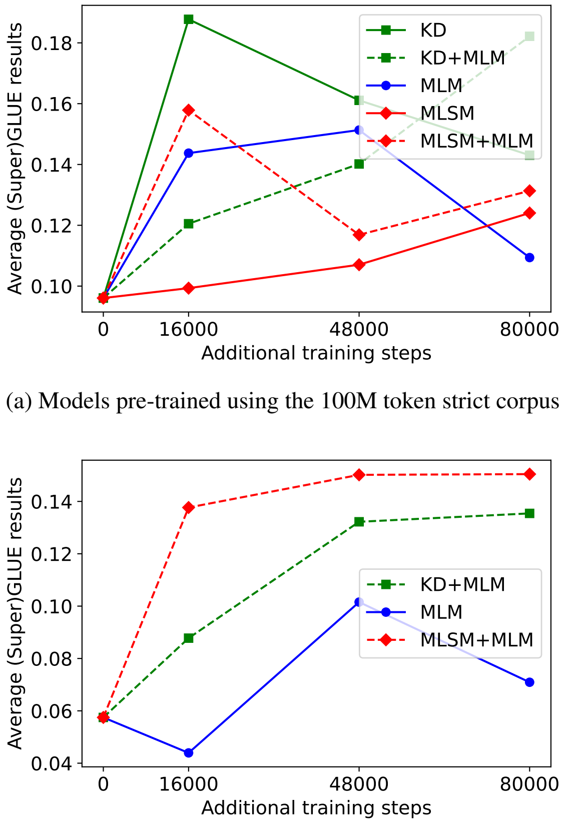
Table 3: MSGS results for the models pre-trained on the strict (a) and the strict-small (b) corpora.



(a) Models pre-trained using the 100M token strict corpus

(b) Models pre-trained using the 10M token strict-small corpus

Figure 4: Average SuperGLUE performances as a function of the number of update steps performed in the second phase of pre-training.



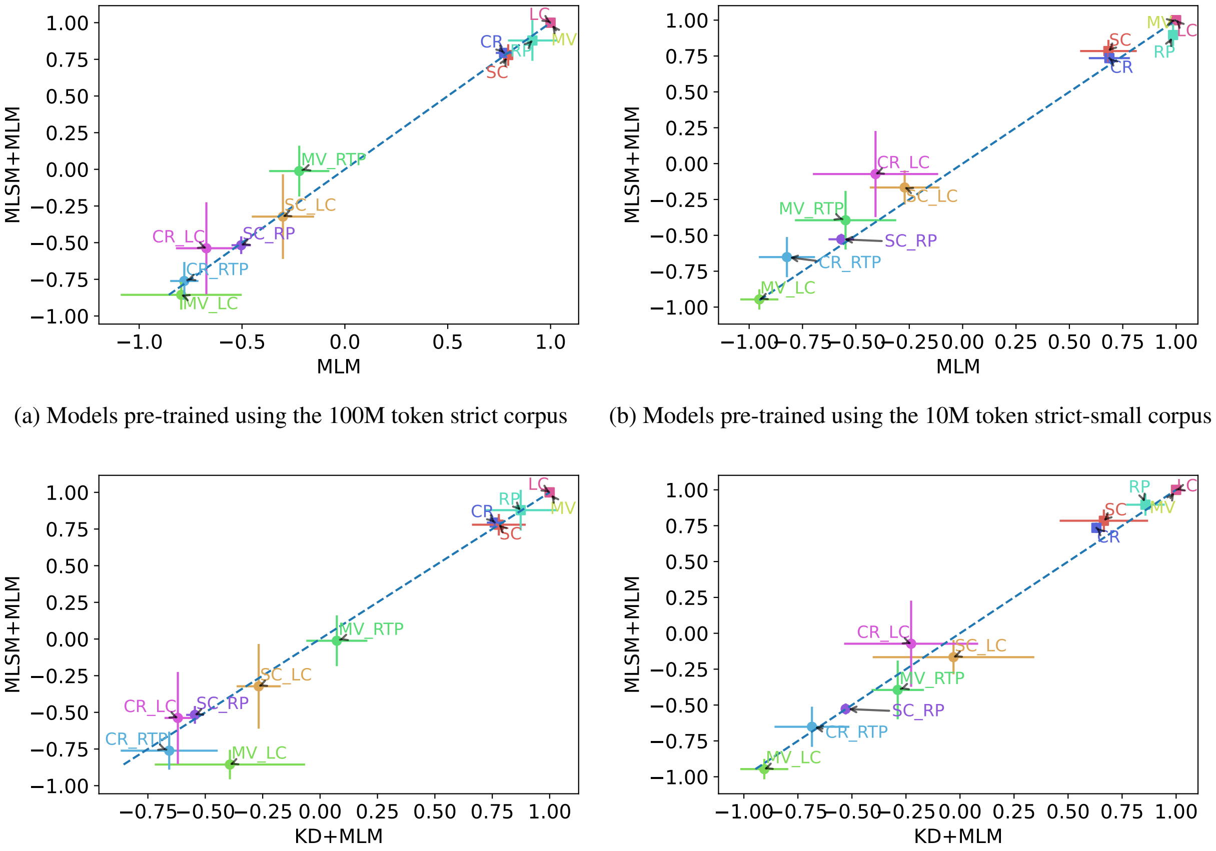
(a) Models pre-trained using the 100M token strict corpus

(b) Models pre-trained using the 10M token strict-small corpus

Figure 5: Average MSGS performances expressed in Matthew Correlation Coefficient as a function of the number of update steps performed in the second phase of pre-training.

cases are separated by an underscore. The five linguistic categories (and their combined challenge tasks) in the BabyLM evaluation framework are the control raising (CR), lexical content (LC), main verb (MV), relative token position (RTP) and SC (syntactic category) classes. The challenge sets are referenced as X_Y, where both X and Y corresponds to one of the above categories and they indicate the two categories that are purposefully conflated in the training, but not in the test set.

The performance of the differently pre-trained models on MSGS is similar to the previously re-



(a) Models pre-trained using the 100M token strict corpus

(b) Models pre-trained using the 10M token strict-small corpus

(c) Models pre-trained using the 100M token strict corpus

(d) Models pre-trained using the 10M token strict-small corpus

Figure 6: Pairwise performance comparison of best performing fully pre-trained models. MLSM+MLM performances are along the y-axis, the x-axis contains the performance of an alternatively pre-trained model. The fine-tuning performance on the unambiguous control tasks and the challenge tasks are denoted by squares and circles, respectively. For the task located above the main diagonal line, MLSM+MLM pre-trained models delivered better fine-tuning performance than the alternatively pre-trained model. The error bars correspond to the standard deviations of the Matthew Correlation Coefficient evaluation scores calculated over four experiments.

ported BLiMP and (Super)GLUE evaluations, i.e., MLSM+MLM pre-trained models perform well not only at the end of pre-training, but also across all the intermediate checkpoints as illustrated by Figure 5. The added value of MLSM+MLM pre-training is the most pronounced when the number of additional update steps is low. For the MSGS evaluation, we can see the largest average performance gain of MLSM+MLM when pre-training was conducted over the 10 million token strict-small training corpus (Figure 5b). The performance gains are already apparent (and actually the most pronounced) after performing only 16000 additional training steps.

Figure 6 contains scatter plots in which the MSGS fine-tuning performance of the best performing pre-trained models can be assessed on the individual tasks. The further a marker above the dashed diagonal line, the larger added value the use of the MLSM+MLM pre-trained model had over an alternatively pre-trained model for the given task. In case a point is located under the main diagonal, MLSM+MLM pre-trained model performed worse than a differently pre-trained model. The majority of the points are located above the diagonal line in each subplot, often by a large margin, confirming the additional benefits of jointly pre-training with masked latent semantic modeling and masked language modeling.

4 Conclusions

Even though MLSM is a cognitively more appealing pre-training objective than MLM, models exclusively pre-trained with MLSM fail at assigning reliable pseudo-log-likelihood scores to sequences (§3.3.1). To this end, we experimented with the coupled use of MLSM loss and the traditional MLM objective.

Our empirical results suggest that the joint use of masked latent semantic modeling and traditional masked language modeling can boost the performance of the pre-trained language models. This is especially the case for tasks that directly assess the linguistic capabilities of the pre-trained models that were obtained by relying on limited corpus size, i.e., the 10 million token strict-small dataset. Our ablation experiments also revealed that the advantages of MLSM pre-training are more pronounced during the earlier phase of pre-training.

Acknowledgments

The research received support from the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory. Additionally, we are grateful for the possibility to use ELKH Cloud (see Héder et al., 2022; https://science-cloud.hu/) which helped us in achieving the results published in this paper.

References

Ahmed Abdelali, Francisco Guzman, Hassan Sajjad, and Stephan Vogel. 2014. The AMARA corpus: Building parallel language resources for the educational domain. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1856–1862, Reykjavik, Iceland. European Language Resources Association (ELRA).

Gábor Berend. 2020. Sparsity makes sense: Word sense disambiguation using sparse contextualized word representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8498–8508, Online. Association for Computational Linguistics.

Gábor Berend. 2023. Masked latent semantic modeling: an efficient pre-training alternative to masked language modeling. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13949–13962, Toronto, Canada. Association for Computational Linguistics.

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.

Emmanuel Dupoux. 2018. Cognitive science in the era of artificial intelligence: A roadmap for reverse-engineering the infant language-learner. *Cognition*, 173:43–59.

Martin Gerlach and Francesc Font-Clos. 2020. A standardized project gutenberg corpus for statistical analysis of natural language and quantitative linguistics. *Entropy*, 22(1).

Jill Gilkerson, Jeffrey A. Richards, Steven F. Warren, Judith K. Montgomery, Charles R. Greenwood, D. Kimbrough Oller, John H. L. Hansen, and Terrance D. Paul. 2017. Mapping the early language environment using all-day recordings and automated analysis. *American Journal of Speech-Language Pathology*, 26(2):248–265.

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

Mihály Héder, Ernő Rigó, Dorottya Medgyesi, Róbert Lovas, Szabolcs Tenczer, Ferenc Török, Attila Farkas, Márk Emődi, József Kadlecsik, György Mező, Ádám Pintér, and Péter Kacsuk. 2022. The past, present and future of the ELKH cloud. *Információs Társadalom*, 22(2):128.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The goldilocks principle: Reading children's books with explicit memory representations. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

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.

Brian Macwhinney. 2000. The childes project: tools for analyzing talk. *Child Language Teaching and Therapy*, 8.

Jenny R. Saffran, Ann Senghas, and John C. Trueswell. 2001. The acquisition of language by children. *Proceedings of the National Academy of Sciences*, 98(23):12874–12875.

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.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–374.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In the Proceedings of ICLR.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Gotlieb Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Adina Williams, Bhargavi Paranjabe, Tal Linzen, and Ryan Cotterell. 2023. Findings of the 2023 BabyLM Challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the 2023 BabyLM Challenge*. Association for Computational Linguistics (ACL).

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

Alex Warstadt, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R. Bowman. 2020b. Learning which features matter: RoBERTa acquires a preference for linguistic generalizations (eventually). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 217–235, Online. Association for Computational Linguistics.