Improving Pre-trained Language Model Sensitivity via Mask Specific losses: A case study on Biomedical NER

Anonymous ACL submission

Abstract

Adapting language models (LMs) to novel domains is often achieved through fine-tuning a pre-trained LM (PLM) on domain-specific data. Fine-tuning introduces new knowledge into an LM, enabling it to comprehend and efficiently perform a target domain task. Fine-tuning can however be inadvertently insensitive if it ignores the wide array of disparities (e.g word meaning) between source and target domains. For instance, words such as *chronic* and *pressure* can often be treated lightly in social conversations, however, clinically, these words are usually a cause of concern. To address insensitive fine-tuning, we propose Mask Specific Language Modeling (MSLM), an approach that efficiently acquires target domain knowledge by appropriately weighting the importance of domain-specific terms (DS-terms) during fine-tuning. MSLM jointly masks DS-terms and generic words, then learns mask-specific losses by ensuring LMs incur larger penalties for inaccurately predicting DS-terms compared to generic words. Results of our analysis show that MSLM improves LMs sensitivity and detection of DS-terms. We empirically show that an optimal masking rate is not only dependant on the LM, but also on the dataset and the length of sequences too. Our proposed masking strategy outperforms advanced masking strategies such as span- and PMI-based masking.

1 Introduction

Fine-tuning is the prevailing practice in adapting an LM to a new domain. A plethora of research works ranging from task-generalization (Claudino et al., 2018; Peters et al., 2019; Peng et al., 2019), to few-shot learning (Gao et al., 2020; McCann et al., 2018) to in-context tuning (Chen et al., 2021) all unanimously credit fine-tuning for the state-of-the-art results across a diverse set of NLP tasks. Despite all of its remarkable strides, fine-tuning has been fairly criticised for its instability and brittleness by a few pockets of NLP researchers (Mosbach et al., 2020; Lee et al., 2019; Dodge et al., 2020). Lee et al. (2019); Dodge et al. (2020) attributed fine-tuning's instability to catastrophic forgetting and small sized datasets, and most recently Mosbach et al. (2020) exposed the optimization challenges encountered during fine-tuning LMs.

Social Conversation	
Dan:	Hi Gary, how was your week?
Gary:	It has ended well but I had a lot of **pressure** throughout the week to meet a deadline. I felt like I would get **attacked** by colleagues.
Clinical Conversation	
Dan:	Hi Gary, how was your week?
Gary:	It has ended well but my **pressure** was high throughout the week. I felt like I would get an **attack**.

Table 1: Comparing the sensitivity of two words in two different conversations (Social and Clinical setting). The brighter the colored boxes wrapping the words, the more concerning for the respective conversation.

It is notable that, across all prior critics, the focus and attention has been strongly directed towards the performance of these LMs, and very limited attention has been paid towards the sensitivity and domain-specific knowledge these LMs pickup during fine-tuning. There is usually such a wide array of disparities between the source (used for pre-training) and the target (used for fine-tuning) domains. Some of these may include but not limited to, word meaning (Navigli, 2009; Zhou and Bollegala, 2021), word intensity (strength or potency of a word in given domain) (Yin et al., 2020; Baek, 2022) and abbreviation disambiguation (Wu et al., 2015). If these disparities are not properly catered for, fine-tuning can easily become an underwhelming adaptation process and insensitive to specialised target domains. For instance, words such as *chronic*, *pressure* and *attack* will often be treated

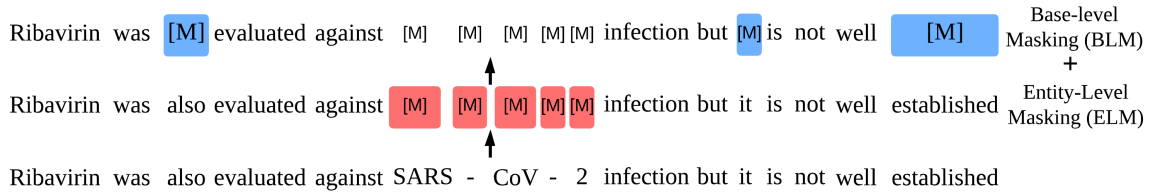

Figure 1: Joint ELM-BLM masking of tokens in an input sequence.

lightly in social conversations, however, clinically these words are usually a cause of concern. For example, we notice that, whereas the words "*pressure*" and "*attack*" are mentioned by the respondent in both the social and clinical contexts in Table 1, they definitely require more attention in the clinical conversation, hence the questioner "Dan" ought to be more sensitive to the respondent "Gary".

In this work, we address the domain sensitive fine-tuning (DSFT) discussed in the previous paragraph. We use insensitivity in our context to imply the below par awareness of DS-terms, rather than language-insensitivity as it pertains to human feelings. We investigate the hypothesis: **"The awareness of or sensitivity of PLMs towards DS-terms can be appropriately elevated without hurting their downstream performance"**.

In order to strategically increase an LM's awareness of DS-terms, we revisit the language understanding and generation phenomenon of Mask Language Modeling (MLM) (Devlin et al., 2019). We modify MLMs to up-weight the significance of masked DS-term tokens such that the attention towards them is relatively larger than that towards masked non DS-term tokens. In doing so, we introduce the notion of **"mask-specific loss"**, which we compute using appropriately assigned weights that are computed using a strategy similar to the one Mosbach et al., 2020 used to address class imbalance. We further introduce entity recognition and entity classification objectives to collectively contribute towards a cross entropy loss with an aim to enhance the ability of a model to detect mentions. We refer to this approach as Mask-Specific Language Modeling (MSLM).

Using the biomedical domain as our test bed, we evaluate how well MSLM can perform when tasked to extract clinical entities from a host of datasets within the Biomedical Language Understanding & Reasoning Benchmark (BLURB) (Gu et al., 2021). To study the effectiveness of MSLM, we do not simply compare the perplexity of our sensitive models to the vanilla models, instead, we proceed to check confidence scores with which the two sets of models predict DS-terms. We assess the impact of our proposed masking strategy by varying the masking rate and lengths of input sequences and monitoring their influence on the LMs prediction results. In addition, we study how this masking strategy compares to other advanced strategies such as PMI (Pointwise Mutual Information) (Levine et al., 2020) and Span (Joshi et al., 2020). Our experiments demonstrate (a) a performance improvement in extraction of exact mentions of named entities, (b) the influence the masking rate and sequence lengths has on prediction performance, and (c) the superiority of the proposed masking strategy over other advanced masking strategies.

2 Mask-Specific Language Modeling

In designing our approach, we draw lessons from two prior tested and proven phenomena: (1) MLMs are effective in learning representations for subtokens, words (Devlin et al., 2019), phrases (Sun et al., 2019) and spans (Levine et al., 2020; Joshi et al., 2020); and (2) high prediction rates (proportion of tokens to be predicted) substantively affect optimization, i.e. they increase training signals, which subsequently boost performance (Wettig et al., 2022). We refer to these two phenomena respectively as the *MLM-effect* and the *High-prediction-effect* in the remainder of this paper.

2.1 Masking

Randomly replacing a percentage of tokens in a sentence with [MASK] tokens (Base level Masking (**BLM**; Devlin et al., 2019)) intuitively enables LMs to learn the bi-directional context that often surrounds words in written language text.

Because certain spans of words are best understood when all of their constituted words are written together to denote a named entity such as a person, an organisation and a location, replacing named entity spans with [MASK] tokens (Entity level Masking (**ELM**; Sun et al., 2019; Abaho et al., 2022)) has also proven to be effective in learning contextualised representations for these

entities.

We leverage benefits of the two above strategies and propose a new masking strategy, "Joint ELM-BLM" shown in Figure 1. On its own, ELM would help enrich an LM with contextual knowledge necessary in discriminating our targeted DS-terms, however, when exploiting the *MLM-effect*, we consider it necessary to avoid tightly coupling the LM's weights onto these DS-terms. We therefore utilise BLM to preserve a PLM's inherent domain and generic knowledge. More so, we avoid the assumption that 15% masking rate is optimal (Devlin et al., 2019) and instead explore several rates to find an optimal one. In our experimental setup, we ensure that BLM- and ELM-masked sets are disjoint sets of tokens.

Besides datasets with annotations of DS-terms (clinical entities), we assume access to a Biomedical PLM denoted as Enc_{PLM}. This LM can be used for encoding each input sequence s of n tokens to obtain $\mathbf{H}$, a matrix of n vectors as shown in (1).

$$\mathbf{H} = \text{Enc}_{\text{PLM}}(x_1, \ldots, [\text{MASK}]_i, \ldots, x_n) \quad (1)$$

2.1.1 Mask specific losses

The main goal in our approach is to strategically increase a PLM's sensitivity towards DS-terms while simultaneously retaining sufficient knowledge of generic terms. The first attempt in achieving this is masking DS-terms along with generic terms as discussed in §2.1.

To further achieve our goal, we introduce the idea of mask specific losses, which essentially aims to impose larger penalties on the model for inaccuracies in predicting corrupted (masked) DS-terms compared to the corrupted generic terms.

Typically, instance-specific losses are computed by re-scaling weights for each possible class in the label space (Wang et al., 2017; Cui et al., 2019), however, in this case, rather than classes, we have ELM- and BLM-masked tokens as well as unmasked tokens. To compute the weights assigned to the tokens in our masked input, we firstly obtain the number of ELM- and BLM-masked tokens within the training dataset and denote them as N_{ELM} and N_{BLM} respectively. A mask specific weight is computed for each of the mask types (ELM & BLM), as the difference between 1 and the the corresponding mask type probability (i.e. the mask type's distribution out of the total mask types distribution), given by (2). The final mask specific weight is obtained as the softmax over the mask specific weights from previous step as given by (5).

$$w_x = 1 - \frac{\text{N}_\text{x}}{\sum_{x \in \{\text{BLM,ELM}\}} \text{N}_\text{x}} \quad (2)$$

$$w_{\text{BLM}} = \begin{cases} 0.5 & \text{if } w_{\text{BLM}} > 0.5 \\ w_{\text{BLM}} & \end{cases}$$
$$w_{\text{ELM}} = \begin{cases} 0.5 & \text{if } w_{\text{ELM}} < 0.5 \\ w_{\text{ELM}} & \end{cases} \quad (3)$$

$$w = ([w_{\text{BLM}}, w_{\text{ELM}}]) \quad (4)$$

$$\boldsymbol{w} = \text{softmax}(w) \quad (5)$$

In order to elevate the sensitivity towards DS-terms but equally avoid overfitting onto them, we introduce a sensitivity threshold, which is used to encourage the ELM-masked tokens related weight (w_{ELM}) and also to carefully suppress the BLM-masked tokens related weight (w_{BLM}). Because of the sporadic nature of the mentions of DS-terms within the dataset, the distribution of ELM-masked tokens will typically be lower than that of BLM-masked tokens, in other words not every input sequence will have a mention of DS-term/s, while every input sequence will have tokens that are subject to BLM. We therefore set the sensitivity threshold to 0.5 to force a balance in their probability distribution (i.e. implying that BLM and ELM are equally likely to occur for a given input sequence). We then ensure that w_{BLM} never rises above this threshold and similarly, w_{ELM} should never fall below that threshold as shown in (5).

The normalized weight vector $\boldsymbol{w}$ is used to compute the MSLM loss (L_{MSLM}) during the prediction of the masked tokens x_i as given by (6).

$$L_{\text{MSLM}} = -\sum w_i^{(x)} \log P(x_i|s) \quad (6)$$

Here, $w_i^{(x)} \in \boldsymbol{w}$ is a mask-specific weight for a masked token x_i that lies within the sequence s.

2.2 Entity detection and Classification

Because the biomedical domain is richly endowed with classification schemes that are used in categorizing clinical entities (Jackson et al., 2018; Gu et al., 2021), we maximize the *High-prediction-effect* by formulating an entity recognition and classification task. The idea behind this is, the more predictions a model has to make (both in predicting masked-out tokens as well as classifying unmasked

	#Sents Train | Val | Test	#Classes	AvgSentLen	#Ments Train | Val | Test	AvgMents	AvgMentsLen
BC2GM	12632 | 2531 | 5065	2	25.17	15197 | 3061 | 632	1.20	2.4
NCBI-disease	5432 | 923 | 942	2	25.24	5134 | 787 | 960	0.95	2.2
BC5CDR-chem	4812 | 4602 | 4582	2	25.75	5385 | 5203 | 5347	1.12	1.3

Table 2: Dataset statistics. #Sents is the number of sentences and #Ments is the number of DS-term mentions, AvgSentLen is the average length of sentences, AvgMents is the average number of DS-terms mentioned per sentence obtained as (# of train Ent_Ments)/(# of train sents). Full table with all datasets in 8 in the Appendix.

entities), the more signals it would get through computing gradients during optimization. The entity recognition task is defined below.

Task formulation: Given a sentence $s = \{x_i\}_{i=1}^{n}$ of n tokens, where each x_i is tagged with a BIO label (Sang and Veenstra, 1999), we build a model that can accurately extract entities $\{e_i^{(s)}\}_{i=1}^{N}$ mentioned in s. We obtain a probability distribution across all BIO labels as given by (7).

$$\hat{y}_i = \text{softmax}(f(\boldsymbol{h_i} \circ \boldsymbol{W}^{(ed)})) \tag{7}$$

Here, f is a nonlinear function, $\circ$ denotes the vector concatenation and $\boldsymbol{W}^{(ed)} \in \mathbb{R}^{1\times k}$ is a trainable weight vector, $h_i \in \mathbf{H}$. In addition to L_{MSLM}, we compute an entity detection loss given by (8).

$$L_{\text{ED}} = -\sum_{i=1}^{n}\sum_{j\in\text{BIO}} y_{i,j} \log \hat{y}_{i,j} \tag{8}$$

Entity Linking/Classification loss: Given a detected entity, we obtain an entity span representation in (9), and compute probability distribution across all entity types E in (10),

$$\boldsymbol{e_m} = \text{meanpool}(h_i, \ldots, h_M) \tag{9}$$

where the entity m has 1 to M tokens.

$$\hat{y}_m^l = \text{softmax}(f(\boldsymbol{e_m} \circ \boldsymbol{W}^{(ec)})) \tag{10}$$

where f is a non-linear function and $\boldsymbol{W}^{(ec)} \in \mathbb{R}^{1\times d}$ is a trainable weight vector. The classification loss is given by (11).

$$L_{\text{EL}} = -\sum_{l\in E} y_m^l \log \hat{y}_m^l \tag{11}$$

Model loss: We optimize the joint loss of all three cross-entropy losses as given in (12).

$$L = L_{\text{MSLM}} + L_{\text{ED}} + L_{\text{EL}} \tag{12}$$

3 Experiments

To evaluate MSLM, we initialize multiple biomedical LMs which were pre-trained on massive collections of publicly available scientific literature in PubMed. Compared LMs include **BioBERT** (Lee et al., 2020), **SciBERT** (Beltagy et al., 2019), **PubMedBERT** (Gu et al., 2021) and **BioELECTRA** (raj Kanakarajan et al., 2021).

Datasets: To facilitate our investigation, we use Named Entity Recognition (NER) datasets within the BLURB benchmark (Gu et al., 2021). These include **NCBI-disease** containing 6892 disease mentions linked to 790 distinct disease concepts, **BC5CDR-Disease** & **BC5CDR-Chemical** containing mentions of diseases and chemicals in 1,500 PubMed articles, **BC2GM** containing 20,000 sentences with gene mentions, **JNLPBA** containing 2,000 PubMed abstracts with mentions of molecular biology-related entities such as DNA and **EBM-NLP** containing 5,000 PubMed clinical trial abstracts with mentions of the PICO elements (We specifically use the version with denoised outcome annotations as used by Abaho et al. (2019, 2021)).

Metrics: We use an exact match (EM) score metric to measure the sensitivity towards DS-terms. EM counts a prediction of an entire entity as 1 if and only if it completely matches the correct answer, both in terms of the precise boundary of the DS-term mention as well as the term's classification. Furthermore, we measure macro-F1 score for NER performance (Hajic et al., 2009) and perplexity of the models to monitor how well the models adapt to and comprehend the domain datasets.

Setup: Two important factors in our setup include, (1) we establish ELM rate with respect to the total number of DS-terms mentioned in an input sequence rather than all input sequence tokens. For example, if the number of DS-terms in a sequence s is denoted as DS_s and $\text{DS}_s = 4$, an ELM of 25% implies $0.25\times 4 = 2$, hence 2 out of the 4 DS-terms

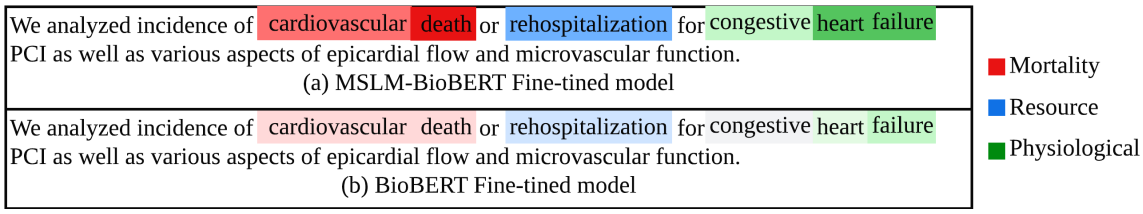

Figure 2: Visualization of the confidence score with which different DS-terms belonging to different classes (within the EBM-NLP dataset) are predicted. The color intensity increases with the confidence score.

are randomly masked. Whenever this computation returns a decimal value, we round off the value upward to the nearest integer (e.g if $\mathrm{DS}_s = 3$, and ELM=25%, $0.25 \times 3 = 1.5$, which will be rounded off to 2), (2) Since ELM consumes a portion of the masking budget as explained above, we halve the conventional 15% rate to get a BLM rate of 7.5%. Furthermore, high masking rates are not favourable for moderately-sized (ca. 125M parameters) LMs (Wettig et al., 2022) such as the ones we use in this paper. This constraint is used in our initial set of experiments, however, later on, we explore how varying both BLM and ELM rates would affect model performance especially because the average length of sentences and DS-term mentions varies across different datasets listed in Table 2.

Implementation details: The infrastructure used in our experiments includes, PyTorch 2.0 for developing MSLM and two GPU machines, a 48G NVIDIA RTX A6000 and a 28G N-series (NC6s_v3) Azure Virtual Machine. The two GPUs are not used to concurrently run the same experiment but to run different experiments in parallel. Results reported are based on testing performance. Dataset statistics are included in Table 2.

		Vanilla	MSLM ELM=1,BLM=0.075
BC2GM	BioBERT	88.4	**90.3**$_{\pm 0.5}$
	PubMedBERT	86.8	**89.8**$_{\pm 0.4}$
	BioELECTRA	87.6	89.1$_{\pm 0.2}$
	SciBERT	85.7	**87.1**$_{\pm 0.4}$
NCBI-disease	BioBERT	89.1	**90.1**$_{\pm 0.1}$
	PubMedBERT	**89.9**	**89.9**$_{\pm 0.2}$
	BioELECTRA	88.5	**88.9**$_{\pm 0.2}$
	SciBERT	88.4	**89.9**$_{\pm 0.1}$
BC5DCR-chem	BioBERT	93.3	**94.0**$_{\pm 0.2}$
	PubMedBERT	94.0	**94.4**$_{\pm 0.2}$
	BioELECTRA	90.8	**94.0**$_{\pm 0.2}$
	SciBERT	90.7	**93.7**$_{\pm 0.2}$
EBM-NLP	BioBERT	64.3	**75.4**$_{\pm 0.4}$
	PubMedBERT	65.5	**76.2**$_{\pm 0.3}$
	BioELECTRA	63.7	**73.2**$_{\pm 0.3}$
	SciBERT	69.7	**73.4** $_{\pm 0.2}$

Table 3: Exact match (EM) scores obtained when MSLM (ELM=100%, BLM=7.5%) is initialized with various biomedical PLMs. Average scores across 5 runs and their standard deviation are reported for the MSLM models which are compared against Vanilla versions of the LMs. Best results are in bold and full results are provided in Table 7 in Appendix.

3.1 Sensitivity towards DS-terms

To investigate the sensitivity of MSLM-fine-tuned models, we evaluate two metrics: (a) the confidence in the models predictions, and (b) the EM score of the predictions. With the former, we visualize the softmax probabilities (which we also refer to as confidence scores) with which model predicts DS-terms using the heatmap in Figure 2. For demonstration purposes, we use the EBM-NLP dataset since it has multiple classes in comparison to the other datasets. As observed in Figure 2, despite both sets of models predicting the correct classes for the 3 DS-terms, cardiovascular death (Mortality outcome), rehospitalization (Resource-use outcome) and congestive heart failure (Physiological outcome), the confidence score with which the model predicts classes for the DS-terms is visibly higher for MSLM-BioBERT models.

Table 3 reports EM scores, which are indicative of the model performance in detecting full or exact mentions of DS-terms. We notice that, MSLM improves the performance (+3.2 percentage points on average) with which LMs detect full mentions. Most notably, we observe significant performance increases in the EM scores for the EBM-NLP dataset (+8.5 percentage points average across models) in comparison to the other datasets, which we attribute to (1) the relatively higher number of Average DS-term mentions per sentence within the dataset, and (2) the relatively bigger training set size as seen in Table 8. With the exception of NCBI-dataset (with PubMedBERT model), we observe that MSLM achieves consistent performance improvements when detecting full mentions of DS-terms.

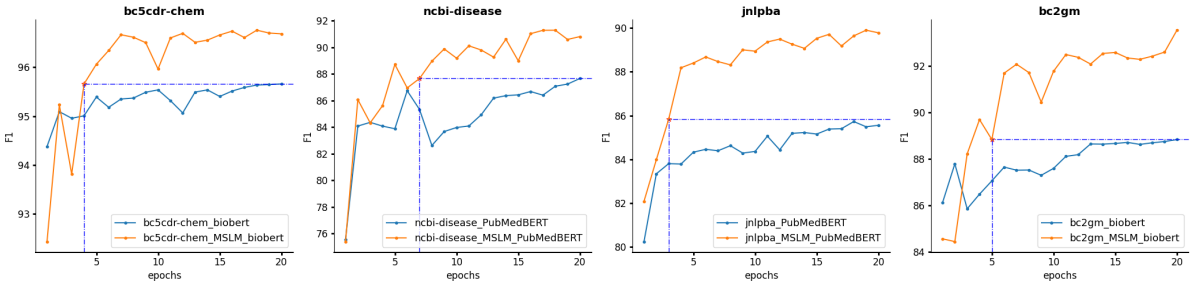

Figure 3: Downstream NER F1 performance of the vanilla and the MSLM-fine-tuned models (i.e. DSFT models). ELM and BLM rates used in §3.1 are maintained. More plots in Appendix F.

	BioB (PPL)	BioB_MSLM (PPL)	Pub (PPL)	Pub_MSLM (PPL)
BC2GM	1.2	2.3 (+1.1)	1.2	1.9 (+0.7)
BC5CDR-Chem	1.1	1.5 (0.4)	1.1	1.2 (+0.1)
JNLPBA	1.4	4.3 (+2.9)	1.4	3.2 (+1.8)
NCBI-Disease	1.2	1.3 (+0.1)	1.1	1.2 (+0.1)

Table 4: Validation perplexity (PPL) recorded when the best NER F1 performance was obtained for vanilla and MSLM models. Biob is BioBERT & Pub is PubMedBERT, and the change in perplexity when vanilla flavors are replaced by MSLM is indicated in brackets.

3.2 Is DSFT destructive?

The success in increasing the sensitivity of LMs towards the DS-terms (via DSFT) is strongly positive as discussed in §3.1, but at what cost? We investigate whether the increased sensitivity comes at the expense of downstream performance, training times and the inherent knowledge of the PLM. For the downstream performance and training times, we monitor the validation NER F1 performance of the MSLM and vanilla flavors over a training time of 20 epochs. Figure 3 shows the MSLM-fine-tuned models consistently outperform the vanilla BioBERT and PubMedBERT during the course of training across the 4 datasets. Furthermore, we observe that MSLM-fine-tuned models achieve the best vanilla performance in a much shorter training time of at most 7 epochs (blue dotted line).

For the inherent knowledge of PLMs, we investigate the validation perplexity to check how well the models understand the domain datasets. As seen in Table 4, perplexity increases when MSLM-fine-tuned models replace vanilla models, however, only by a few percentage points. We hypothesize that, diminishing the penalties incurred when predicting non DS-terms (as constrained by (3)) will most likely limit the model's capability to reconstruct corrupted non DS-terms, hence affecting the net perplexity of the models. This change however proves that low perplexity does not necessarily correlate with good performance, a hypothesis also discovered by Wettig et al. 2022.

Overall, the performance improvement achieved by DSFT is evidence supporting the earlier defined hypothesis; i.e. The awareness of or sensitivity of PLMs towards DS-terms can be appropriately elevated without hurting downstream performance.

4 Varying the Masking rates

Devlin et al. 2019 choose the 15% masking rate with caution, suggesting that a higher rate risks leaving insufficient context for the LM to learn good representations. However, this caution can be misleading because, several other factors can influence the optimal masking rates such as the model size and type of the task (Liao et al., 2020). We therefore vary the BLM and ELM rates and study the performance changes of the model. To do this, we design the experiments as follows,

1. ELM: We select a range of ELM rates from 25% to 100% with interval gaps of 25%. The interval is kept to 25% because values < 25% would not change the overall number of DS-terms to mask, following the ELM mask computation we establish in our setup in §3.

2. BLM: We select a minimum rate of 0% and maximum rate of 22.5% with intervals of 7.5%. We cap the masking budget for BLM to 22.5% because we use base models (ca. 125M parameters), which have been reported to struggle in high masking regimes (>20%) (Wettig et al., 2022). Using a 7.5% interval is our strategy that enables inclusion of the popular 15% rate in our set of rates to investigate.

The resulting sets of rates used in the experiments are [0.25, 0.50, 0.75, 1] and [0, 0.075, 0.15, 0.225] for ELM and BLM respectively.

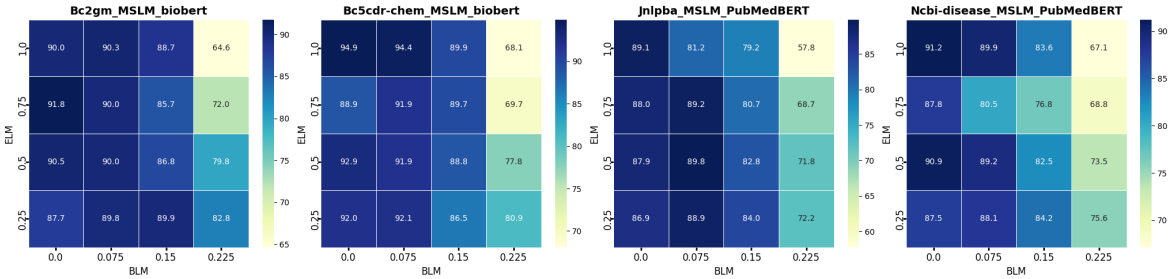

Figure 4: Test Exact match (EM) scores of varying ELM and BLM rates when two MSLM-fine-tuned models (MSLM_biobert and MSLM_PubMedBERT) are evaluated on the datasets.

ELM(%)	BLM(%)	<AvgSentLen [51]	>AvgSentLen [5104]
100	22.5	19.4	85.4
75	15.0	41.2	84.4
50	7.5	75.1	84.0
25	0.0	66.3	77.9

Table 5: Comparisons of the EM performance of low and high masking regimes for short and long sequences using MSLM_BioBERT. <AvgSenLen [51] implies, 51 sentences that are shorter than the average sentence length and similarly, >AvgSentLen, 5104 sentences that are longer than the average sentence length.

From Figure 4, we see that increasing both the ELM and BLM rates consistently degrades the performance of the models across all four datasets (i.e. the lowest performance is certainly obtained when both BLM and ELM are high as seen at the top right of all plots). As seen, increasing the BLM rate is only beneficial up to a certain point (7.5%), and that irrespective of a high or low ELM rate, performance dramatically drops when BLM hits 15%. These two noticed revelations point to the fact that a high net corruption/masking rate leaves very minimal context to learn from and hence effectively re-construct DS-terms in input sequences, which are already not very long sequences as shown in Table 2. Overall, we observe that the optimal Joint ELM-BLM masking is task dependent, however is usually $\geq$0.25 for ELM and $\leq$0.15 for BLM.

Masking Rate and Sequence Length: To further understand how much context is necessary when fine-tuning the MLM, we study the performance of different rates with different sequence lengths on BC2GM.[1] We constrain the rates to low masking regimes, which we define as ELM $\leq$ 0.5 and BLM $\leq$ 0.075, and high masking regimes as ELM $\geq$ 0.75 and BLM $\leq$ 0.15. Because of the laborious nature of the task of constructing a test set with sufficient samples for varying sequence lengths, we use the average sentence length (AvgSentLen in Table 2) as a cut off point, where sentences above it are considered as relatively long (>AvgSentLen) and those below as relatively short (<AvgSentLen). We do not perform separate experiments but rather compute the EM scores of the predictions on the short and long sentences.

In Table 5, we observe that high masking regimes favour long sentences (i.e. overall, highest rates produce the best performance for long sentences and worst performance for short ones). This implies that the models are still able to learn sufficiently from long sequences despite a high masking rate. We also observe, while the performance on long sentences is consistently better, it does not significantly differ from that of short ones for the low rates, implying that low rates have minimal impact on varying sequence lengths, and hence LM relies heavily on its inherent pre-trained knowledge.

5 Comparisons against Prior Masking Strategies

Besides our proposed masking strategy (i.e. Joint ELM-BLM), there are various other advanced masking strategies such as PMI-Masking (PMI) (Levine et al., 2020) and Random-Span masking (SPAN) (Joshi et al., 2020). With PMI, spans of co-occurring words (2-4) (a.k.a collocations) are identified, ranked based on PMI scores computed using the PMI measure proposed by Levine et al. (2020) and stored in a vocabulary. The ranked spans discovered in an input sequence are masked. In the SPAN approach, spans of varying lengths (2-4) are arbitrarily selected and masked. In both approaches, the total masking budget (number of tokens to mask) is maintained to avoid biasing the comparative analysis. Extended details of how we

[1]We use BC2GM as it has the largest number of sentences below average length compared to the other datasets, which are dominated (ca. 95%) by sentences above average length

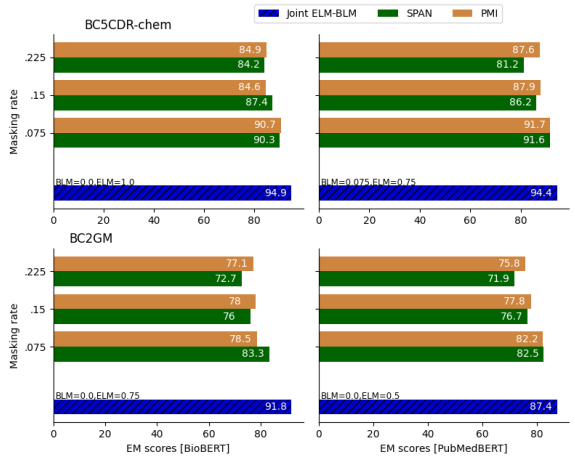

Figure 5: Comparing performance of other masking strategies across various rates with the best performance of our proposed Joint ELM-BLM. Results of BioBERT (left) and PubMedBERT (right) evaluated on BC2GM and BC5CDR-chem and hatch the bars with best scores.

implement SPAN and PMI are in Appendix E.

In Figure 5, we directly replace Joint ELM-BLM with either SPAN or PMI and vary the total masking budget while maintaining the optimal budget for Joint ELM-BLM. For instance, we keep ELM = 0.75 and BLM = 0.0 for Joint ELM-BLM, when evaluating MSLM_Biobert on BC2GM dataset because they are the optimal rates. However, we vary the rates for both PMI and SPAN across the values in the set of BLM rates established in §4.

We observe that Joint ELM-BLM outperforms other strategies across all experiments. PMI produces majority of the second best results despite SPAN masking being quite competitive. We attribute PMI's performance to the fact that the PMI's vocabulary from which spans to mask are drawn has a high concentration (> 50%) of DS-terms (details in Appendix E.3)), which effectively makes it similar to ELM that directly masks DS-terms. As noticed earlier in §4, masking DS-terms is highy effective even with no BLM masking (i.e. BLM = 0.0). We also observe the slight drop in performance as the masking rate increases across PMI and SPAN, which further confirms the fact that LMs are likely to struggle when decoding highly corrupted sequences (Devlin et al., 2019).

6 Related work

Domain adaptation of PLMs for NER: The conventional approach in prior work tackling domain adaptation for NER has focused pre-training on unlabelled target domain corpora and then fine-tune on downstream target domain dataset (Lee et al., 2020; Beltagy et al., 2019). Recent work has explored minimising the discrepancy between the source and target embedding distributions (Zhang et al., 2021; Poerner et al., 2020). Our work mostly aligns with Poerner et al. (2020) who also adopt "non-target domain pre-training".

Masking: The originally proposed masking approach that involved replacing a percentage of tokens at random (TOKEN masking) with [MASK] tokens (Devlin et al., 2019) has been modified in recent works to improve MLM. Sun et al. (2019) and Abaho et al. (2022) mask named entity spans (entity masking), Joshi et al. (2020) mask random spans of tokens (SPAN masking) and Levine et al. (2020) mask groups of co-occurring words (PMI masking). With the exception of PMI, our proposed Joint ELM-BLM masking approach aligns well with all recent masking modifications. It simultaneously masks disjoint sets of random tokens and entity spans. Targeting multiple units in a sentence makes it greedier than prior works, however, we emphasize mask rate tuning and upholding a masking budget to achieve optimal performance.

7 Conclusion

We considered the problem of DSFT aiming to improve an LM's sensitivity (i.e. awareness of) towards DS-terms. We proposed MSLM, an approach that jointly masks DS-terms and random words, while conditioning the LM to larger penalties during optimisation for incorrect predictions of DS-terms. Using the biomedical domain as a testbed, the performed experiments reveal improvements MSLM makes over vanilla fine-tuning in exact DS-term match detection. MSLM's efficiency is proven when models achieve higher NER F1 scores in a much shorter training time. We substantiate the recent narrative, dismissing 15% as a universally optimal rate in MLM (Wettig et al., 2022), by proving that optimal performance is influenced by varying masking rates and length of sequences.

The Joint ELM-BLM masking strategy we propose dominates performance when compared to advanced masking methods. Although we focus on biomedical NER, our proposed MSLM approach can be be adapted for DSFT for other domains. The positive impact of our proposed masking method motivates us to investigate its effectiveness during pre-training of MLMs in future work.

8 Limitations

The list of pre-trained biomedical LMs we use in our experiments can be considered as a representative sample that is used frequently for biomedical text mining. However, there are few other biomedical LMs such ClinicalBERT (Alsentzer et al., 2019) and BlueBERT (Peng et al., 2020), whose inclusion can quantitatively improve results of our analysis. Despite casting it as an NER task focused on not simply detecting DS-terms, but confidently detecting them for that matter, some other tasks worthy of consideration for investigating sensitivity may include but not limited to, question and answering (Choi et al., 2018), common sense reasoning (Davis and Marcus, 2015), event detection (Weng and Lee, 2011) etc. Furthermore, studying the performance of domain sensitive fine-tuning in other domains besides biomedicine would be a qualitative addition and is recommendable for future research under the guise of improving LM sensitivity.

9 Ethics

This work addresses insensitive fine-tuning that arises from the neglection of the disparities and nuances between source and target domains. In addressing this problem, our proposed fine-tuning method neither guards against nor removes any present biases (social, gender etc) in the pre-trained MLMs.

Additionally, we do not annotate any data for the datasets we adopt as they are all existing datasets that are commonly used for biomedical text mining.

Furthermore, we credit all prior work whose output directly or indirectly influences our work especially with the datasets and the methods. In our evaluation experiments, we declare some results that were not generated from a seperate set of experiments but instead obtained by selectively retrieving a set of sentences that conform to the evaluation criteria we targeted i.e. short and long sentences. In comparing our masking strategy to the advanced bench-marking strategies, we study performance across various masking budgets in order to provide a fair comparison with our proposed method. To further remove any modelling bias, we elaborately discuss implementation details of compared methods in Appendix.

References

Michael Abaho, Danushka Bollegala, Paula Williamson, and Susanna Dodd. 2021. Detect and classify–joint span detection and classification for health outcomes. *arXiv preprint arXiv:2104.07789*.

Micheal Abaho, Danushka Bollegala, Paula Williamson, and Susanna Dodd. 2019. Correcting crowdsourced annotations to improve detection of outcome types in evidence based medicine. In *CEUR workshop proceedings*, volume 2429, pages 1–5.

Micheal Abaho, Danushka Bollegala, Paula Williamson, and Susanna Dodd. 2022. Position-based prompting for health outcome generation. *arXiv preprint arXiv:2204.03489*.

Emily Alsentzer, John R Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical bert embeddings. *arXiv preprint arXiv:1904.03323*.

Hyunah Baek. 2022. Prosodic disambiguation in first and second language production: English and korean. *Language and Speech*, 65(3):598–624.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Yanda Chen, Ruiqi Zhong, Sheng Zha, George Karypis, and He He. 2021. Meta-learning via language model in-context tuning. *arXiv preprint arXiv:2110.07814*.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. Quac: Question answering in context. *arXiv preprint arXiv:1808.07036*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

João Gustavo Claudino, Tim J Gabbett, Frank Bourgeois, Helton de Sá Souza, Rafael Chagas Miranda, Bruno Mezêncio, Rafael Soncin, Carlos Alberto Cardoso Filho, Martim Bottaro, Arnaldo Jose Hernandez, et al. 2018. Crossfit overview: systematic review and meta-analysis. *Sports medicine-open*, 4(1):1–14.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9268–9277.

Ernest Davis and Gary Marcus. 2015. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58:92–103.

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:Volume 1 (Long and Short Papers), pages 4171–4186. Association for Computational Linguistics.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23.

Jan Hajic, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, M Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, et al. 2009. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18.

Richard Jackson, Ismail Kartoglu, Clive Stringer, Genevieve Gorrell, Angus Roberts, Xingyi Song, Honghan Wu, Asha Agrawal, Kenneth Lui, Tudor Groza, et al. 2018. Cogstack-experiences of deploying integrated information retrieval and extraction services in a large national health service foundation trust hospital. *BMC medical informatics and decision making*, 18(1):1–13.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

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

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. 2019. Mixout: Effective regularization to fine-tune large-scale pretrained language models. *arXiv preprint arXiv:1909.11299*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Yoav Levine, Barak Lenz, Opher Lieber, Omri Abend, Kevin Leyton-Brown, Moshe Tennenholtz, and Yoav Shoham. 2020. Pmi-masking: Principled masking of correlated spans. *arXiv preprint arXiv:2010.01825*.

Yi Liao, Xin Jiang, and Qun Liu. 2020. Probabilistically masked language model capable of autoregressive generation in arbitrary word order. *arXiv preprint arXiv:2004.11579*.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. *arXiv preprint arXiv:2006.04884*.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):1–69.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. Scispacy: fast and robust models for biomedical natural language processing. *arXiv preprint arXiv:1902.07669*.

Yifan Peng, Qingyu Chen, and Zhiyong Lu. 2020. An empirical study of multi-task learning on bert for biomedical text mining. *arXiv preprint arXiv:2005.02799*.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: an evaluation of bert and elmo on ten benchmarking datasets. *arXiv preprint arXiv:1906.05474*.

Matthew E Peters, Sebastian Ruder, and Noah A Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. *arXiv preprint arXiv:1903.05987*.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. Inexpensive domain adaptation of pretrained language models: Case studies on biomedical ner and covid-19 qa. *arXiv preprint arXiv:2004.03354*.

Kamal raj Kanakarajan, Bhuvana Kundumani, and Malaikannan Sankarasubbu. 2021. Bioelectra: pretrained biomedical text encoder using discriminators. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 143–154.

Erik F Sang and Jorn Veenstra. 1999. Representing text chunks. *arXiv preprint cs/9907006*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. 2017. Learning to model the tail. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 7032–7042.

Jianshu Weng and Bu-Sung Lee. 2011. Event detection in twitter. In *Proceedings of the international aaai conference on web and social media*, volume 5, pages 401–408.

Alexander Wettig, Tianyu Gao, Zexuan Zhong, and Danqi Chen. 2022. Should you mask 15% in masked language modeling? *arXiv preprint arXiv:2202.08005*.

Yonghui Wu, Jun Xu, Yaoyun Zhang, and Hua Xu. 2015. Clinical abbreviation disambiguation using neural word embeddings. In *Proceedings of BioNLP 15*, pages 171–176.

Fulian Yin, Yanyan Wang, Jianbo Liu, and Lisha Lin. 2020. The construction of sentiment lexicon based on context-dependent part-of-speech chunks for semantic disambiguation. *IEEE Access*, 8:63359–63367.

Tao Zhang, Congying Xia, Philip S Yu, Zhiwei Liu, and Shu Zhao. 2021. Pdaln: Progressive domain adaptation over a pre-trained model for low-resource cross-domain named entity recognition. In *EMNLP*.

Yi Zhou and Danushka Bollegala. 2021. Learning sense-specific static embeddings using contextualised word embeddings as a proxy. *arXiv preprint arXiv:2110.02204*.

Appendices

A Dataset statistics

The full table containing dataset statistics partially presented in Table 2, is shown in Table 8.

B Hyperparameters

Parameter	Tuned-range	Optimal
Train Batch size	[8,16,32]	8
Eval Batch size	[8,16,32]	8
Epochs	[10,20,30,50]	20
k	[50, 100,200,300]	100
d	[50,100,200,300]	100
Optimizer	[Adam, SGD]	Adam
Learning rate	[5e-5, 1e-4, 5e-3, 1e-3]	5e-5

Table 6: Parameter settings for the MSLM-fine-tuned models. k and d are dimensions of the the randomly initialised trainable weight vectors $\boldsymbol{W}^{(ed)} \in \mathbb{R}^{1\times k}$ defined in 7 and $\boldsymbol{W}^{(ec)} \in \mathbb{R}^{1\times d}$ defined in 10 respectively.

C Sensitivity towards DS-terms

Table 7 presents the full results of EM scores in detecting full or exact mentions of DS-terms. We observe an average increment of +3.2 points across all datasets when all four LMs are used.

		Vanilla	MSLM BLM=0.075 ELM=1
BC2GM	BioBERT	88.4	**90.3**$_{\pm 0.5}$
	PubMedBERT	86.8	**89.8**$_{\pm 0.4}$
	BioELECTRA	87.6	89.1$_{\pm 0.2}$
	SciBERT	85.7	**87.1**$_{\pm 0.4}$
NCBI-disease	BioBERT	89.1	**90.1**$_{\pm 0.1}$
	PubMedBERT	**89.9**	**89.9**$_{\pm 0.2}$
	BioELECTRA	88.5	**88.9**$_{\pm 0.2}$
	SciBERT	88.4	**89.9**$_{\pm 0.1}$
BC5DCR-chem	BioBERT	93.3	**94.0**$_{\pm 0.2}$
	PubMedBERT	94.0	**94.4**$_{\pm 0.2}$
	BioELECTRA	90.8	**94.0**$_{\pm 0.2}$
	SciBERT	90.7	**93.7**$_{\pm 0.2}$
EBM-NLP	BioBERT	64.3	**75.4**$_{\pm 0.4}$
	PubMedBERT	65.5	**76.2**$_{\pm 0.3}$
	BioELECTRA	63.7	**73.2**$_{\pm 0.3}$
	SciBERT	69.7	**73.4** $_{\pm 0.2}$
BC5DCR-dis	BioBERT	91.7	**93.4** $_{\pm 0.2}$
	PubMedBERT	92.3	**94.1** $_{\pm 0.1}$
	BioELECTRA	89.7	**93.5** $_{\pm 0.3}$
	SciBERT	90.1	**93.4** $_{\pm 0.2}$
JNLPBA	BioBERT	86.3	**88.9** $_{\pm 0.2}$
	PubMedBERT	85.7	**89.8** $_{\pm 0.2}$
	BioELECTRA	80.0	**83.4** $_{\pm 0.2}$
	SciBERT	82.4	**85.4** $_{\pm 0.2}$

Table 7: Full Exact match scores obtained when MSLM is initialized with various pre-trained biomedical LMs. These scores are compared against Vanilla versions of the LMs. Best and second-best are bold and underlined. Partial results of the table are presented in the main body in Table 3.

D Domain Adaptation

Even if we do not technically have a source and target domain for respectively pre-training fine-tuning, our work aligns with prior work which achieves domain adaptation without pre-training on a target domain. Poerner et al. 2020 build a model called greenBioBERT in a relatively less expensive approach and fine-tune it on the same datasets we do. greenBioBERT is word2vec trained on PubMed+PMC articles and with an updated embedding layer and tokenizer following BERT's architecture. The authors consider this as an LM not pre-trained on target domain.

We compare test NER F1 perfomance in our experiments with both grteenBioBERT and vanilla BioBERT. Results in Table 9 show our MSLM-fine-tuned BioBERT outperform all the others by at least +2.3 points. This further indicates the heitened awareness of DS-terms that MSLM is able to achieve hence effectively improving its entity detection performance.

	#Sents Train | Val | Test	#Classes	AvgSentLen	#Ments Train | Val | Test	AvgMents	AvgMentsLen
BC2GM	12632 | 2531 | 5065	2	25.17	15197 | 3061 | 632	1.20	2.4
NCBI-disease	5432 | 923 | 942	2	25.24	5134 | 787 | 960	0.95	2.2
BC5CDR-chem	4812 | 4602 | 4582	2	25.75	5385 | 5203 | 5347	1.12	1.3
BC5CDR-dis	4812 | 4602 | 4582	2	25.75	4182 | 4246 | 4424	0.87	1.7
JNLPBA	14731 | 3876 | 3873	2	30.05	32178 | 8575 | 6241	2.18	3.0
EBM-NLP	32074 | 4009 | 4010	5	24.68	21498 | 2677 | 2736	2.67	2.0
MIMIC III	9937 | 1242 | 1243	3	1943.85	863732 | 106539 | 107330	8.67	2.0

Table 8: Dataset statistics. #Sents and #Ments are the number of sentences and number of DS-term mentions respectively for the train, validation and test splits, AvgSentLen is the Average length of sentences, AvgMents is the Average number of DS-terms mentioned per sentence obtained as (# of train Ent_Ments)/(# of train sents) and AvgMentsLen is the average length of DS-terms.

	BioBERT (Lee et al., 2020)	GreenBioBERT (Poerner et al., 2020)	MSLM-BioBERT ELM=1,BLM=0.075
BC5CDR-disease	87.15	85.08	**89.45**
NCBI-disease	89.71	85.94	**91.91**
BC5CDR-chem	93.47	93.08	**96.79**
BC2GM	84.72	83.45	**92.17**
JNLPBA	77.49	76.89	**83.24**

Table 9: Downstream NER test F1 scores when different variants of BioBERT are fine-tuned on the datasets. Reference scores from compared methods (Lee et al., 2020) and (Poerner et al., 2020). Best and second best results are in bold and underlined respectively.

E Masking strategies

We compare our proposed joint ELM-BLM masking strategy to two other advanced masking strategies, PMI (Levine et al., 2020) and Random SPAN (Joshi et al., 2020) whose implementation we respectively present in a pseudo code in the algorithms 2 and 1.

E.1 SPAN Masking (1)

Given a tokenized input sequence and a masking rate m_r as input (line 1), we initialize a pool of indices (of the same size as the input sequence $|s|$) randomly ordered ($s_{\text{random_pool}}$). Each random index is a possible starting index of a contiguous span to be masked. We compute the masking budget m_b as product between rate and input sequence size to get number of tokens to be masked e.g. if $|s| = 10$ and $m_r = 0.15$, $mb = 0.15 \times 10$. For each random index in the pool $s_{random_{pool}}$, we initialize a span length s_l randomly $s_l \in 2, 3, 4$ at line 4 i.e. this is the length of the contiguous span to be masked. Three different constraints satisfied as we iteratively select random spans to be masked include, 1) the number of already masked tokens summed up with span length s_l should be less than the masking budget m_b (line 7-9), 2) then the end

Algorithm 1 SPAN Masking

1: **Input**: Tokenized input sequence:- s, masking_rate:- m_r, mask token:- [MASK], **Output**: Masked Tokenized Input sequence s_{M}
2: Initialize the below,
 - A pool of indices ($s_{\text{random_pool}}$) randomly ordered, where $|s|$ = $|s_{\text{random_pool}}|$
 - masked_budget mb = $\text{math.ceil}(m_r \times |s|)$
 - masked_so_far $msf = 0$
3: **for** index i in $s_{\text{random_pool}}$ **do**
4: Initialize random_span_length $sl_{i=2}^{4}$
 i.e. span to be masked could vary from length 2 to 4.
5: sl = $\min(sl, mb)$
6: start, end = i, i+sl
 Don't mask beyond the masking budget [7-10]
7: **if** $(msf + sl) > mb$: **then**
8: sl = mb - msf
9: end = i+sl
10: **end if**
 Don't mask beyond sequence bounds [11-13]
11: **if** $end \geq |s| - 1$ **then**
12: end = $i + sl$
13: sl = end - start
14: **end if**
 Don't mask already masked spans [15-17]
15: **if** s_{M}[start:end] has no [MASK] tokens **then**
16: $s_{\text{M}}[start{:}end]$ = [MASK] $*$ sl
17: msf += sl
18: **end if**
19: **if** msf $\geq$ mb **then**
20: break
21: **end if**
22: **end for** return s_{M}

index of span to masked should not be greater than the end index of the input sequence (line 11-13), then finally the selected span to be masked should not contain already masked tokens inhibiting overlapping masking (line 15-17). Once all constraints are satisfied, the span's tokens within the input sequence are masked or replaced with mask token [MASK].

E.2 PMI Masking (2)

With PMI, we begin by constructing a PMI vocabulary of word n-grams of lengths 2–4. These n-grams contain words that co-occur in sentences a minimum of 5 times within the entire dataset. A PMI score for each collocation (n-gram of co-occurying words) is computed using the PMI measure (Levine et al., 2020). The collocations are ranked and ordered in their respective lengths.
NB: Each dataset has its own PMI vocabulary.

Given a tokenized input sequence and a masking rate m_r as input (line 1). The masking budget m_b is computed similar to the SPAN approach (line 2). For each collocation (gram) in the vocabulary, we check if collocation is a subsequence (contiguous) of the input sequence. One constraint satisfied is 1) the number of already masked tokens summed up with span length s_l should be less than the masking budget m_b (line 8-10), 2). Once constraint is satisfied, the span's tokens within the input sequence are masked or replaced with mask token [MASK].

Algorithm 2 PMI Masking

1: **Input**: Tokenized input sequence:- s, masking_rate:- m_r, mask token:- [MASK], PMI_vocabularly (PMI$_v$)
Output: Masked Tokenized Input sequence s_{M}
2: Initialize the below,
- masked_budget $mb = \mathrm{math.ceil}(m_r \times |s|)$
- masked_so_far $msf = 0$
3: **while** $msf \leq mb$ **do**
4: **for** gram in PMI$_v$ **do**
5: **if** gram is a subsequence in s_{M} **then**
6: Get start (st) and end (ed) indices of gram in s_{M}
7: gram$_l$ = |gram|
8: **if** msf + gram$_l > mb$ **then**
9: gram$_l$ = mb - msf
10: end = st+gram$_l$
11: **end if**
12: $s_{\mathrm{M}}[st{:}ed]$ = [MASK]$*$ gram$_l$
13: msf += gram$_l$
14: **end if**
15: **end for**
16: **end while**
17: return s_{M}

E.3 PMI vocabularly overlapping DS-terms

#DS-terms	#PMI-vocab	#Overlap (# | %)
18890	15787	8130 | 51.5

Table 10: Number of vocabularly terms that overlap across with DS-terms in the BC2GM dataset. "#" implies number of, % implies percentage of the vocabularly that are DS-terms.

Table 10 shows that 51.5% of the phrases in the constructed PMI's vocabularly (for the BC2GM dataset) are DS-terms. This high concentration of DS-terms in the PMI vocabularly implies that there is a high similarity between PMI masking and Entity Level Masking (ELM) and hence making PMI masking nearly as effective as standalone ELM masking (i.e. even without BLM masking). Table 13 shows a sample of the DS-terms that overlap (in blue) across with the PMI vocabularly.

F Is DSFT destructive?

We present the complete list of all plots from the experiments investigating whether DSFT is destructive hence exploring an answer to the hypothesis in the introduction, i.e. the awareness of or sensitivity towards DS-terms can be appropriately elevated when fine-tuning without hurting downstream performance.

As observed in Figure 6, we observe better results achieved by the MSLM fine-tuned models, more so, achieving the best performance of the vanilla models in a much shorter training time. A couple of other things we notive include, performance during the course of training of bioelectra models doesn't seem to signigicantly differ from that of the MSLM_bioelectra models across all datasets. We also notice that unlike all the other models, with bioelectra, MSLM_fine-tuned models achieve the best performance of the vanilla models after 10 epochs, i.e. longer than the other models. We attribute bioelectra's competitiveness to its inherent architectures (ELECTRA; Clark et al., 2020)

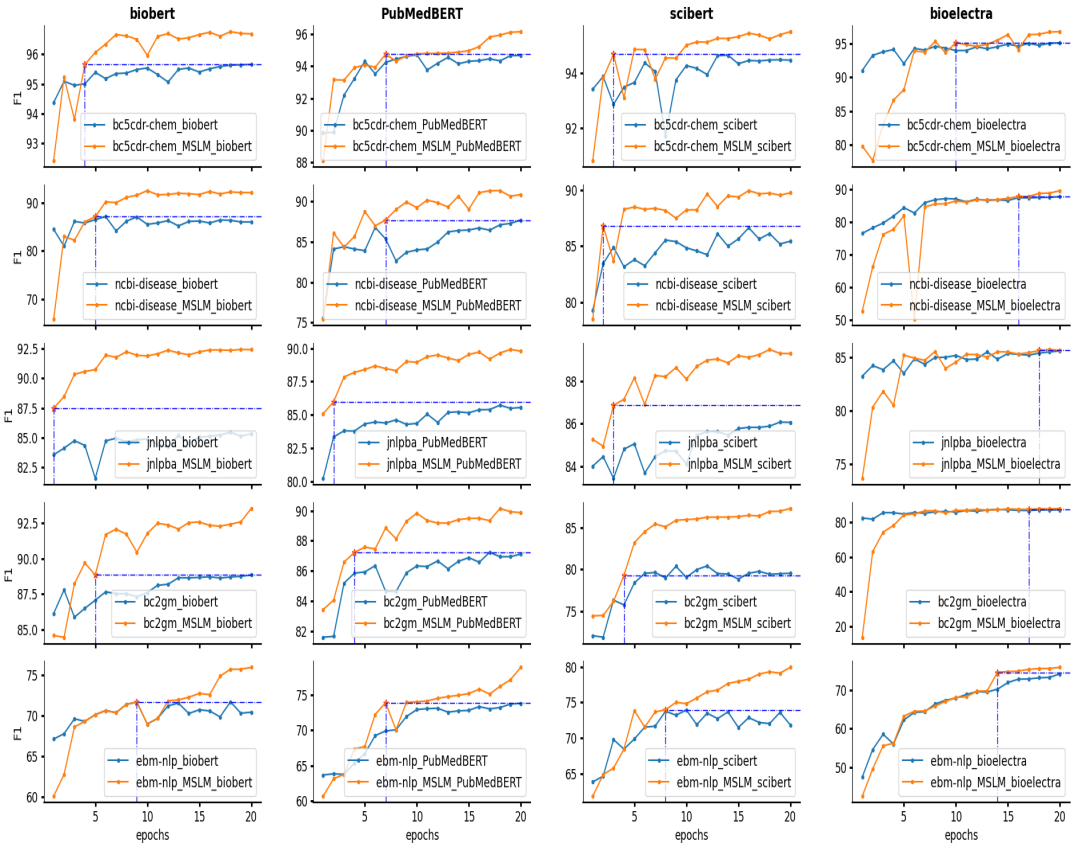

Figure 6: Downstream NER F1 performance of the vanilla and the MSLM-fine-tuned models. "MSLM" is used to uniquely identify DSFT models. ELM and BLM rates used in §3.1 are maintained. Each row contains results specific to a dataset e.g. first row has BC5CDR-chem, second has NCBI-disease etc. Similarly each column contains results specific to pre-trained biomedical LM.

which, similar to MSLM, it adds a model to detect whether MLM has correctly replaced a token or not (token replacement detection). Electra trains a generator (which is an MLM) to predict tokens for masked slots, and additionally trains a discriminator to predict whether a token has been replaced or the original masked token is what the generator predicted. Whereas MSLM doesn't add any model on top of the MLM, it targets MLM components i.e. tilting the MLMs sensitivity towards masks tokens corresponding to DS-terms.

G Additional Analysis

Due to space limitations, we defer additional investigations to further validate our MSLM approach to this Appendix. We investigate MSLM in a weakly supervised setting and detail everything in following sections.

G.1 Weak supervision of MIMIC-III

Specifically, we employ MIMIC-III v1.4 (Johnson et al., 2016) dataset, and retrieve a sample of 5000 patient records from the NOTESEVENT table (within the MIMIC-III v1.4 database) containing de-identified free text entries recorded by physicians and other care providers during patient-care. Figure 7 illustrates the pipeline used in annotating mimic-III in a weakly supervised process.

We use Cogstack medcat[2], a biomedical annotation tool, to extract and categorise medical concepts based on medical semantic types defined[3] in UMLS and Snomed.

Because of the unequal distribution of the semantic types across the annotations, we narrow down the scope of target UMLS semantic concepts with the help of a clinical consultant who clusters concepts into three high-level clinical concepts of Diseases, Symptoms and Treatments, as shown in Table 11.

After the annotations, we then use SpaCy[4] (Neumann et al., 2019) for sentence segmentation of each record (a row containing multiple paragraphs)

[2] https://medcat.readthedocs.io/en/latest/index.html

[3] https://lhncbc.nlm.nih.gov/ii/tools/MetaMap/documentation/SemanticTypesAndGroups.html

[4] https://spacy.io/

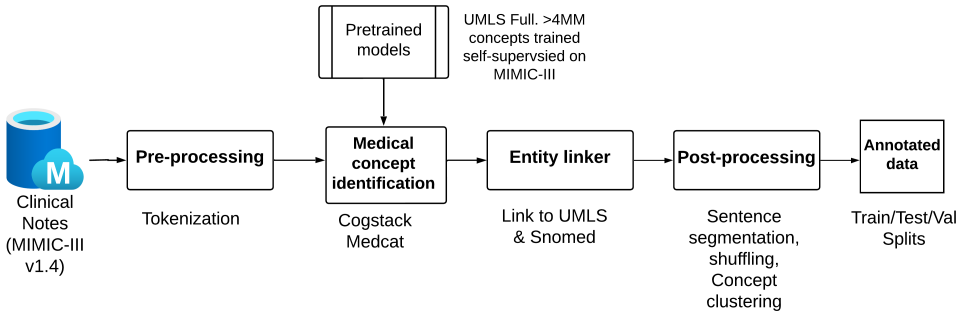

Figure 7: Weakly supervised annotation of MIMIC-III v1.4.

Cluster Category	Associated UMLS Semantic Types
Treatments	["Pharmacologic Substance", "Clinical Drug", Antibiotic]
Diseases	["Acquired Abnormality", "Anatomical Abnormality", "Bacterium", "Archaeon", "Congenital Abnormality", "Cell or Molecular Dysfunction", "Disease or Syndrome", "Virus", "Neoplastic process"]
Symptoms	["Social Behavior", "Sign or Symptom", "Mental or Behavioral Dysfunction"]

Table 11: UMLS semantic types that Cogstack can link to are clustered into three high level categories by a clinical consultant. These clusters encapsulate the semantic types in an easy-to understand manner

and split the resulting list of sentences into train, validation and test sets (9937, 1242 and 1243 sentences respectively), which are then subsequently used in fine-tuning.

G.2 Results

After preliminary tuning of BLM and ELM rates on validation set, we find the optimal BLM and ELM rate as 0.075 and 0.5 respectively, which achieves an average improvement of 2.7 points in EM scores over the vanilla approach as seen in Table 12. This improvement further indicates how beneficial MSLM is in improving extraction of DS-terms from clinical patient data. rather than just scientific literature in BLURB datasets.

		Vanilla	**MSLM** BLM=0.075 ELM=0.5
MIMIC-III	BioBERT	90.1	**92.6**$_{\pm 0.2}$
	PubMedBERT	89.8	**93.8**$_{\pm 0.4}$
	BioELECTRA	88.1	**90.1**$_{\pm 0.2}$
	SciBERT	87.5	**89.7**$_{\pm 0.4}$

Table 12: Exact match (EM) scores. Average scores (across 5 runs) obtained for fine-tuning LMs on weakly supervised dataset constructed using MIMIC-III patient records.

BC2GM PMI Vocabulary		
Bombyx mori	**tyrosine kinase receptor**	**glucocorticoid receptor**
IE promoter	**tyrosine kinase**	ad lib
CASE REPORT	dystrophic epidermolysis	ad libitum
Codonopsis pilosula	exacerbate cryoblobulinemia	aggregative fimbriae
E2 proteins	fluoromethyl ketone	**thyroid hormone receptor**
HMR locus	**uPA mRNA**	amylose cornstarch
LY 294002	police officers	**prolyl isomerase**
Leptomonas seymouri	**uPA mRNA**	fenfluramine anorexia
OAE screener	Enterococcus faecalis	hexamethylpropyleneamine oxime
latent membrane protein 2A	Fugu rubripes	**IgG antibodies**
Pisum sativum	**ets family**	**rheumatoid factor**
protein tyrosine kinase	**RNA polymerase**	myasthenia gravis
Punta Toro	Nicotiana tabacum	otoacoustic emissions
Rhodosporidium toruloides	P22 R17	substantia innominata
PDH complex	San Francisco	**PDGF receptors**
dopamine D2 receptor	**thymidine kinase promoter**	synovial chondromatosis
Trait Personality	bicycle ergometer	**LDL cholesterol**
Van der	**paired domain**	vena cava
Veterans Affairs	dura mater	Dirofilaria immitis
human chorionic gonadotropin	fluticasone propionate	**alpha 2AP**
chengchi tang	**recombinant human erythropoietin**	**Cre recombinase**
cysteine proteinase	**CAT reporter gene**	Spodoptera frugiperda
dig1 dig2	orientational anisotropy	Zea mays
dihydrolipoyl transsuccinylase	patent ductus	reticulocyte lysate
bacterial chloramphenicol acetyltransferase	pia mater	**polypyrimidine tract binding protein**
chloroacetate esterase	**translation upstream factor**	BACTEC 9000
.	.	.
.	.	.
.	.	.
bHLH proteins	epidermolysis bullosa	**TCR beta**
ta chengchi	**fork head**	**NMDA receptor**
pleckstrin homology domain	**dopamine receptor**	SELECTION CRITERIA
Aedes aegypti	PCC 7120	acoustic neuroma
Autographa californica	Selected topics	acoustic startle
RNAP II	chloromethyl alkyl	**exonuclease III**
sigma 54	**firefly luciferase gene**	aphthous stomatitis
El Paso	irritation sensation	**SR family**
Expiratory Flow	**viral LTR**	flexor motoneurons
Gulf War	Fusarium moniliforme	plan spared
Hematopoietic growth factors	Jenkins Activity	**antithrombin III**
Rhodobacter capsulatus	**histone H3**	**epidermal growth factor**
Src homology	Medical Radiology	rear corner
Task Force	**S1 nuclease**	vas deferens
Toxocara canis	NnS neurones	vinyl siloxane
monoamine oxidase	Rhizobium leguminosarum	**ERK MAPK**
cytochrome oxidase	**9804 gene**	interspecific backcross
acne vulgaris	**cyclin D1**	**growth hormone**
aluminium hydroxide	emollient cream	SB 203580
binocular pregeniculate	imino protons	circular dichroism
U5 RNA	nontumorigenic Ad5	**beta receptor**
campestris pv	**MAP kinase**	**TK gene**
Ogg1 protein	proportional hazards	hypoxaemic resuscitation
forward projection	**pertussis toxin**	intraindividual fluctuations
preformed triplexes	volatile solvents	northern Norway
areA product	Karger AG	**capsid proteins**
alkaline phosphatase	**env genes**	prizidilol hydrochloride
Aryl hydrocarbon	**integrin subunits**	**SH3 domain**
CEN ENV	aryl hydrocarbon	thiazide diuretics
Epidemiologic Follow	dyad symmetry	von Willebrand
PKC beta	multifocal leukoencephalopathy	**proliferating cell nuclear antigen**

Table 13: PMI vocabulary constructed from BC2GM dataset. DS-terms (in blue) discovered within the constructed PMI vocabularly