Ignore Me But Don't Replace Me: Utilizing Non-Linguistic Elements for Pretraining on the Cybersecurity Domain

Eugene Jang[1] **Jian Cui**[2]* **Dayeon Yim**[1]
Youngjin Jin[3] **Jin-Woo Chung**[1] **Seungwon Shin**[3] **Yongjae Lee**[1]
[1]S2W Inc. [2]Indiana University Bloomington [3]KAIST
[1]{genesith,dayeon,jwchung,lee}@s2w.inc
[2]cuijian@iu.edu [3]{ijinjin,claude}@kaist.ac.kr

Abstract

Cybersecurity information is often technically complex and relayed through unstructured text, making automation of cyber threat intelligence highly challenging. For such text domains that involve high levels of expertise, pretraining on in-domain corpora has been a popular method for language models to obtain domain expertise. However, cybersecurity texts often contain non-linguistic elements (such as URLs and hash values) that could be unsuitable with the established pretraining methodologies. Previous work in other domains have removed or filtered such text as noise, but the effectiveness of this approach has not been investigated, especially in the cybersecurity domain. We experiment with different pretraining methodologies to account for non-linguistic elements (NLEs) and evaluate their effectiveness through downstream tasks and probing tasks. Our proposed strategy, a combination of selective MLM and jointly training NLE token classification, outperforms the commonly taken approach of replacing NLEs. We use our domain-customized methodology to train CyBERTuned, a cybersecurity domain language model that outperforms other cybersecurity PLMs on most tasks.

1 Introduction

Cybersecurity is a critical concern as the world continues to grow reliant on technology. Modern cybersecurity practice emphasizes the need for preemptive defense utilizing Cyber Threat Intelligence (CTI) — actionable information on possible cyber-threats (Farnham and Leune, 2013). However, due to the unstructured and complex nature of such information, leveraging CTI requires extensive manual inspection by human experts (Husari et al., 2017). Although automating cyber threat intelligence has been regarded as important (Fernández Vázquez et al., 2012; Kampanakis, 2014), it has been considered highly challenging (Wagner et al., 2019).

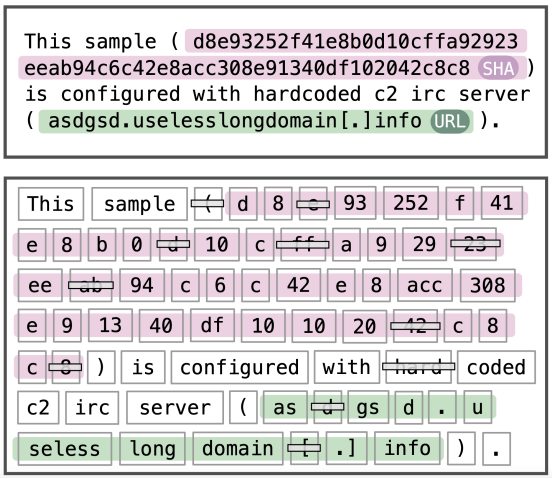

Figure 1: A threat report excerpt after tokenization and masking with 15% probability. The tokens inside the SHA hash and URL are highlighted. Masked tokens are indicated by a gray bar.

Meanwhile, pretrained language models (PLMs) have shown great potential for text comprehension (He et al., 2020). However, PLMs are unlikely to have developed the necessary expertise for domains that require significant domain knowledge, such as the cybersecurity domain. This could be somewhat addressed by extremely large models (Sergeev, 2023), but this option is costly to train and run. A more common approach to teach domain expertise to PLMs has been to pretrain on a domain-specific corpus. The effectiveness of such domain-pretrained PLMs has been demonstrated in the biomedical (Lee et al., 2019), scientific (Beltagy et al., 2019), and legal (Chalkidis et al., 2020) domains to name a few. Several cybersecurity domain PLMs (Ranade et al., 2021; Aghaei et al., 2022; Bayer et al., 2022) were also trained in a similar manner.

However, cybersecurity texts often incorporate non-linguistic elements (NLEs) that could be inappropriate for self-supervised pretraining. Self-supervised objectives like masked language mod-

*Work performed while at S2W Inc.

Findings of the Association for Computational Linguistics: NAACL 2024, pages 29–42

eling (MLM) (Devlin et al., 2019) or de-noising objectives (Lewis et al., 2020) learn by recovering original texts from a masked or modified state. While this is mostly beneficial for natural language, such tasks might not be effective when trying to recover tokens in non-linguistic parts of text. Figure 1 shows an excerpt from a malware threat report containing a SHA hash and a URL. The SHA hash tokens are linguistically random, and therefore training a model to correctly recover these tokens may not be beneficial. Similarly, the URL tokens are less predictable compared to the natural language text surrounding it, and therefore potentially unsuitable for pretraining.

Outside of the cybersecurity domain, previous works addressed such NLEs through replacement (e.g. replace all URLs with "*[URL]*") (Dai et al., 2020; Caselli et al., 2021; Jin et al., 2023) or filtering (Le et al., 2020; Raffel et al., 2020; Hung et al., 2022). However, no attempt has been made to verify whether these approaches actually benefit pretraining. It is also unclear whether such practices would have similar benefits in the cybersecurity domain, where it is more common for NLEs to be used alongside natural language. Conversely, pretraining with NLEs could be beneficial to utilize the informational value of NLEs. For instance, a model may learn to identify suspicious domains in URLs or recognize familiar hash values in the way human cybersecurity experts can.

We investigate different strategies of pretraining on the cybersecurity domain. We first identify commonly occurring NLE types that can be extracted using regular expressions. We then pretrain models using different MLM strategies, testing the effectiveness of selective masking and NLE token classification and comparing to the vanilla MLM and replacement strategy. Our experiments suggest that replacement benefits on downstream tasks but harms performance on probing tasks, especially near NLEs. Instead, we find that a strategy of selective masking while jointly training with NLE token classification generally outperforms the replacement strategy. Using this strategy, we train CyBERTuned (Cybersecurity BERT-like Utilizing Non-linguistic Elements of the Domain), a cybersecurity domain PLM trained with the domain-customized pretraining methodology. We show CyBERTuned outperforms comparable cybersecurity domain PLMs in most tasks. CyBERTuned model weights, training resources, and code are publicly available at `https://github.com/genesith/CyBERTuned`.

Our contributions are as follows:

- We propose and test multiple strategies to deal with NLEs when pretraining on a cybersecurity corpus.
- Through experiments on a variety of domain tasks, we find a strategy that is preferable to the common practice of replacing NLEs.
- We use our methodology to train CyBERTuned, a cybersecurity domain encoder model that outperforms other cybersecurity models.
- We provide our model weights, training resources, and code.

2 Related Work

Cybersecurity NLP Automating cyber threat intelligence has been often discussed in literature (Kampanakis, 2014; Wagner et al., 2019; Jo et al., 2021). Classical off-the-shelf NLP methods, such as regex processing and dependency parsing, have been used to extract attack patterns (Husari et al., 2017) or malware behaviors (Zhu and Dumitraş, 2016) from cybersecurity texts. Recent works explore the potential of using BERT (Devlin et al., 2019) for more complex tasks such as exploitability prediction (Yin et al., 2020), malware detection (Rahali and Akhloufi, 2021), and dark web analysis (Jin et al., 2023).

Domain PLMs PLMs train with large text corpora using self-supervision tasks (Devlin et al., 2019; Lewis et al., 2020; He et al., 2020). Many domain PLMs (Lee et al., 2019; Chalkidis et al., 2020; Beltagy et al., 2019) were able to outperform general PLMs on domain-specific tasks by simply replicating existing pretraining procedures on domain corpora. PLMs for the cybersecurity domain using this approach have been suggested by several works (Ranade et al., 2021; Aghaei et al., 2022; Bayer et al., 2022). Our work differs in that we use a domain-customized methodology after investigating the effectiveness of various strategies.

Pretraining Strategies Self-supervised tasks for pretraining have been investigated by many works (Lewis et al., 2020; Aroca-Ouellette and Rudzicz, 2020; Yamaguchi et al., 2021). Some works find improvements by modifying the masking procedure of the MLM task. Changing mask-

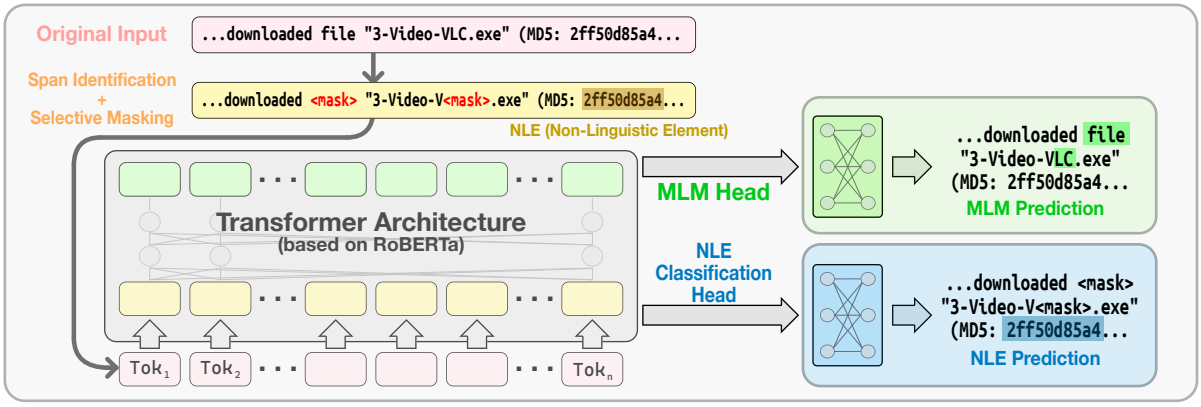

Figure 2: The overall architecture of CyBERTuned. NLE spans from the original input is used in the masking step and for NLE classification.

ing from token-level to word-level significantly improved BERT pretraining for Chinese (Cui et al., 2021). Works on selective masking suggested that masking tokens important to tasks (Gu et al., 2020), entities (Lin et al., 2021), or reasoning (Sanyal et al., 2023) more frequently was effective. Conversely, our selective masking method skips masking tokens that are ineffective for MLM training. Previously, dropping less important tokens midtraining was suggested as a way to increase training efficiency (Hou et al., 2022), but at the cost of losing semantic sensitivity (Zhong et al., 2023). In our work, ineffective tokens are identified beforehand via NLE spans, and are skipped during masking rather than dropped during training.

3 Method

In this section, we first discuss the types of NLEs in cybersecurity texts and how to extract NLE instances. We then propose some methods to utilize the extracted NLE spans in pretraining with cybersecurity texts.

3.1 Non-linguistic Elements

Cybersecurity texts often feature non-linguistic text alongside natural language. Among non-linguistic texts, certain types of are extractable with regular expressions (Husari et al., 2017). We narrow our scope to non-linguistic elements that can be identified by regular expressions, since our aim is to apply them into self-supervised tasks. After manual inspection of cyber threat reports, we select the NLE types that are both frequent and identifiable with regular expressions. The following 7 types were selected: *URLs*, *email addresses*, *IP addresses*, *MD5 hashes*, *SHA hashes*, *Bitcoin addresses*, and *CVE IDs*[1]. Note that we do not consider NLE types that require significant effort to extract precisely, such as filepaths or code blocks. We also extend detection to defanged NLEs (e.g., *hxxp*://example.com, 192.168*[.]*1.192) by utilizing the iocide Python library[2].

3.2 Leveraging NLE Spans in Pretraining

We study two methods to leverage extracted NLE spans to guide the pretraining. Figure 2 shows a model utilizing both methods.

NLE Classification: The model is explicitly instructed to predict which tokens belong to NLEs in the pretraining text. This can be modeled as a simple token classification task and can be trained alongside the MLM task. Each token is labeled with its NLE type (0 if outside of NLE span), which is predicted by a token classification head (linear layer).

Since this task is a more semantically shallow task compared to the original MLM task, it should not dominate the total loss function (Aroca-Ouellette and Rudzicz, 2020). Therefore, we apply a scaling factor (0.1) before adding with the MLM loss to produce the total loss.

Selective MLM: In the masking stage, the NLE spans are used to avoid masking tokens that are inside NLEs. However, since informational content varies between NLE types it must be investigated whether all NLE types should be avoided.

[1]CVE (Common Vulnerabilities and Exposures) IDs are unique identifiers assigned to publicly disclosed vulnerabilities.

[2]https://pypi.org/project/iocide/

Strategies	Masks		NLEC	Example (MLM)	Example (NLE Classification)
	SLE	FNLE			
Vanilla MLM	✓	✓		The Dropper drops a zipped SysJoker (53f1bb23f670d331c9041748e7e8e396) from C2 https[://]github[.]url-mini[.] com/msg.zip, copies it to	N/A
Replace All	(replaced)			The Dropper drops a zipped SysJoker (<MD5>) from C2 <URL>, copies it to	N/A
Vanilla + NLEC	✓	✓	✓	The Dropper drops a zipped SysJoker (53f1bb23f670d331c9041748e7e8e396) from C2 https[://]github[.]url-mini[.] com/msg.zip, copies it to	The Dropper drops a zipped SysJoker (53f1bb23f670d331c9041748e7e8e396) from C2 https[://]github[.]url-mini[.] com/msg.zip, copies it to
Mask-Semis	✓			The Dropper drops a zipped SysJoker (53f1bb23f670d331c9041748e7e8e396) from C2 https[://]github[.]url-mini[.] com/msg.zip, copies it to	N/A
Mask-Semis + NLEC	✓		✓	The Dropper drops a zipped SysJoker (53f1bb23f670d331c9041748e7e8e396) from C2 https[://]github[.]url-mini[.] com/msg.zip, copies it to	The Dropper drops a zipped SysJoker (53f1bb23f670d331c9041748e7e8e396) from C2 https[://]github[.]url-mini[.] com/msg.zip, copies it to
Mask-None + NLEC			✓	The Dropper drops a zipped SysJoker (53f1bb23f670d331c9041748e7e8e396) from C2 https[://]github[.]url-mini[.] com/msg.zip, copies it to	The Dropper drops a zipped SysJoker (53f1bb23f670d331c9041748e7e8e396) from C2 https[://]github[.]url-mini[.] com/msg.zip, copies it to

Table 1: Comparisons between how each text types are processed in different strategies. In the MLM examples, highlighted sections indicate text that are considered for masking. In the NLE Classification examples, each token is predicted for its NLE type (indicated by color).

We note that NLEs that involve human generated text (URLs and emails), unlike protocol-generated values (IP addresses, hash values, etc.), can contain linguistically meaningful information. For instance, a human expert may identify the URL *github[.]url-mini[.]com/msg.zip* as a malicious file download link from a fake domain masquerading as the legitimate GitHub domain.

To make a simple distinction between NLE types, we group the NLEs based on whether they are generated by humans or protocol. Specifically, we group *URLs* and *emails* as **semi-linguistic elements** (SLEs) and *IP addresses*, *MD5 hashes*, *SHA hashes*, *Bitcoin addresses*, and *CVE IDs* as **fully non-linguistic elements** (FNLEs). We then test two settings of selective masking: **Mask-None**: all NLE types are avoided during masking. **Mask-Semis**: fully non-linguistic elements are avoided but semi-linguistic NLEs are allowed to be masked.

4 Pretraining the Models

To evaluate our pretraining methodology on the cybersecurity domain, we pretrain models on a cybersecurity text corpus using a number of strategies. We first describe the tested pretraining strategies, including the vanilla MLM and ablation settings. We also describe our cybersecurity corpus and show statistics that suggest NLEs are more frequent in cybersecurity texts.

4.1 Pretraining Strategies

We compare a total of 6 pretraining strategies. First we include two commonly used strategies as baselines. As ablation studies, we also test two strategies using only one method. Then two strategies that utilize both NLE classification and selective MLM are described. Examples of these strategies can be seen in Table 1.

Vanilla MLM: The original masking strategy (Devlin et al., 2019). After tokenization, 15% of the input tokens are selected for prediction. Following the original implementation, 80% are converted into the mask token, 10% are converted into a random token, and 10% are unchanged.

Replace All: A commonly used strategy to reduce the impact of NLEs in pretraining (Caselli et al.,

Data Source	Count	Data Size
Full		
Online security articles	150K	680.8 MB
Security paper abstracts	7.3K	9.1 MB
Wikipedia articles	3.4K	15.7 MB
CVE descriptions	185K	52.5 MB
Pretraining Subset		
Online security articles	34K	170.4 MB

Table 2: Statistics of data sources used in the corpus. The data used to pretrain the models is a subset of the total data.

NLE	Ours (PS)	Ours (Full)	Wiki	C4
URL	16,272	5,172	62	404
EMAIL	3,282	901	< 1	33
IP	2,503	780	3	15
MD5	2,651	754	< 1	1
SHA	550	161	< 1	< 1
BTC	1,024	273	< 1	< 1
CVE	1,225	550	< 1	3

Table 3: Distribution of non-linguistic elements (per million words) in our pretraining subset (PS) corpus, full corpus, Wikipedia, and C4.

2021; Jin et al., 2023). The MLM method is unchanged, but the NLEs in the input corpus is converted to an identifier of the NLE type (e.g., all CVE IDs are replaced with "<CVE>"). However, comes with a risk of reducing informational content in the pretraining corpus.

Vanilla + NLEC: The MLM method is unchanged, but the joint task of token-level NLE classification (NLEC) is also performed. While MLM is still done on NLE tokens, NLEC could instruct the model to understand the different role the tokens have.

Mask-Semis: A selective MLM method that avoids masking of FNLEs (hash values, IP addresses, etc.) while allowing masking of SLEs (URLs and emails).

Mask-Semis + NLEC: A strategy using both the selective masking and NLEC. In this setting, tokens in SNLEs are allowed to be masked and tokens in FNLEs are avoided.

Mask-None + NLEC: A strategy using both the selective masking and NLEC. In this setting, tokens in all NLEs are avoided during masking.

4.2 Cybersecurity Corpus

We collect and curate a large amount of text from publicly available online sources. Like other cybersecurity PLMs, we construct our corpus with data from a variety of sources: Online Security Articles, Security Paper Abstracts, Wikipedia Articles, and CVE Descriptions. A detailed description of the components and collection of the corpus can be found in Appendix A and D.

Pretraining subset. We further identify a subset of the corpus focused on threat reports to pretrain on. This is because the full corpus covers a variety of styles, including news articles written for non-expert audiences. Such articles contain little technical information and few NLEs. Since our goal is to compare pretraining strategies of teaching technical expertise of analysts to models, we filter to find sources that publish for expert audiences. From 60 total online source sites, we select 30 sites that more often feature technical information to make up the pretraining subset. The data size of our sources used for constructing the corpus and the pretraining subset is listed in Table 2.

NLE Statistics. To demonstrate the frequency of non-linguistic elements in our cybersecurity text corpus, we compare our corpus with two general domain text corpora: the Wikipedia corpus and the C4 corpus (Raffel et al., 2020). The Wikipedia corpus, used in pretraining BERT and other models, consists of text content from Wikipedia articles. The C4 corpus, first used for pretraining T5 (Raffel et al., 2020), is a collection of crawled web pages. Unlike our corpus, the C4 corpus aims to include only natural language text and use heuristics to filter text with non-natural language. Due to the large size of this dataset, we sample 0.1% of the total size (365,234 documents) for our analysis.

To compare between corpora, we calculate the frequency of NLEs. We first count the number of instances of each NLE type, using our detection methodology (discussed in Section 3.1) on each corpus. We use the NLTK (Bird and Loper, 2004) tokenizer to count the number of words in each corpus. Table 3 shows the frequencies of each non-linguistic element per million words. We observe that the frequency of NLEs in our corpus is significantly higher compared to the two general domain corpora.

Strategies	Downstream Tasks			Probing Tasks	
	CyNER	CySecED	MTDB	All	Near-FNLEs
RoBERTa	0.637	0.504	0.802	0.278	0.270
Vanilla MLM	0.648	0.510	0.822†	0.382	0.460
Replace All	0.664†*	0.544	0.827†	0.381	0.438
Vanilla + NLEC	0.652	0.526	0.820	0.380	0.455
Mask-Semis	0.638	0.538	0.817	**0.386**	0.463
Mask-Semis + NLEC	**0.667**†*	**0.544**†	0.825†	0.383	**0.464**
Mask-None + NLEC	0.643	0.533†	**0.829**†	0.382	0.452

Table 4: Experimental results on multiple pretraining strategies. Downstream tasks show median values over 10 runs. **Boldface** represents the best score and underlined values represents the second best score. The † symbols indicates statistically significant distributions from the RoBERTa-base baseline. The * symbols indicates statistically significant distributions from the Vanilla MLM baseline.

4.3 Pretraining Setup

For our experiments, we pretrain further on the pretrained RoBERTa-base model (Liu et al., 2019). We choose the RoBERTa model as the base architecture because its minimal pre-tokenization scheme and coverage is suitable to our corpus[3]. For efficiency, we choose to pretrain further on the pretrained model, following findings that suggest that this method is as effective as training a model from scratch (Chalkidis et al., 2020; El Boukkouri et al., 2022). We mostly follow RoBERTa's training hyperparameters, with few modifications to account for our smaller corpus size (details can be found in Appendix C). Note that the *Replace All* strategy modifies the pretraining corpus size. For fair comparison, all models were trained for 500 steps ($\sim$12 epochs for the *Replace All* model, $\sim$10 epochs for other models).

5 Experiments

We evaluate the models trained by each pretrained strategy with both downstream tasks and probing tasks. For comparison, we also experiment with the base RoBERTa model.

5.1 Downstream Tasks

We compare the ability of each model to fine-tune onto downstream tasks using challenging cybersecurity datasets.

[3]The BERT pretokenizer assumes there are spaces between the ':', '/', '.' characters common in URLs. The corpus also contains obscure characters that aren't considered by other tokenizers(the T5 tokenizer does not have the '\' character in its vocabulary.

CyNER (Alam et al., 2022): A named entity recognition dataset of annotated malware threat reports. The reports are annotated for five entity types: Malware, System, Organization, Indicator, and Vulnerability.

CySecED (Man Duc Trong et al., 2020): An event detection dataset of annotated articles from The Hacker News. The articles are annotated for 30 fine-grained events types describing cyber-attacks or vulnerabilities.

MalwareTextDB (MTDB) (Lim et al., 2017; Phandi et al., 2018): A dataset of malware reports annotated for four types of attributes ActionName, Capability, StrategicObjectives and TacticalObjectives. The labels are cast into a multiple choice question format, where the objective is to identify the correct attribute given a passage, attribute type, and answer choices.

5.2 Probing Tasks

A disadvantage of comparing performance with downstream tasks is that fine-tuning modifies the model weights learned from pretraining. In order to evaluate the model weights themselves, we probe the model's ability to produce correct MLM answers for relevant tokens similar to the LAMA (Petroni et al., 2019) framework. We follow the domain-specific version by Chalkidis et al. (2023), in which a list of legal terminology was used to find instances of the terms from a target corpus. Models are then evaluated by its ability to recover the correct terminology after it is masked.

Our implementation tests model ability to cor-

rectly identify cybersecurity terminology in text context. We first construct a list of relevant terminology by taking words used in MITRE's database of enterprise attack techniques[4]. After processing and filtering, we identify 226 tokens to be used for probing (see Appendix B). To probe the ability of each model, we evaluate the MLM performance on the validation split of the full corpus after masking all target tokens. A total of 77,983 tokens were masked. We also mark tokens in the vicinity (within 20 tokens away) of FNLEs, to see if the presence of FNLEs affects the probing performance. A total of 4,906 tokens were near FNLEs.

5.3 Results

The results of the experiments are presented in Table 4. For downstream tasks, we report the median values over 10 seeds. We mark statistical significance of $p < 0.05$ compared with the base RoBERTa and *Vanilla MLM* baselines. F1 scores are shown for the CyNER and CySecED tasks and accuracy is shown for the MTDB task and probing tasks.

NLEC. Comparing the *Vanilla MLM* with the *Vanilla + NLEC model* suggests that the classification task, on its own, does not provide meaningful benefits. However, when comparing the *Mask-Semis* and *Mask-Semis + NLEC* settings, the addition of the NLEC task provides a noticeable benefit in downstream tasks. In both comparisons, NLEC caused a slight decrease in the probing tasks.

Selective Masking. Comparing the *Vanilla MLM* with the *Mask-Semis* model suggest that selective masking does not produce consistent gains in downstream tasks, although it benefits probing tasks. Comparing *Mask-Semis + NLEC* and *Mask-None + NLEC* settings, the strategy of masking SLEs seems to benefit more consistently across downstream tasks and the probing tasks. This results suggests that there is value in performing masking on URLs and emails.

Best performers. While different pretraining methods suit different tasks (Lewis et al., 2020), the *Mask-Semis + NLEC* model performed consistently well across all tasks. The *Replace All* model was also very capable in downstreaming tasks, but was weaker in probing tasks. Especially, the model probing performance was worst of all the pretrained models when the probed token was near an FNLE. This is an undesirable characteristic of the model, since the model is expected to encounter multiple FNLEs in the domain. We argue *Mask-Semis + NLEC* is the best strategy because it allows the model to utilize NLEs while achieving high downstream performance.

6 CyBERTuned Experiments

6.1 Pretraining CyBERTuned

With our findings, we train our final model CyBERTuned. We train on a larger scale with the *Mask-Semis + NLEC* strategy. We compare our model with other language models on a larger array of downstream tasks in the cybersecurity domain. The CyBERTuned model is trained on our full cybersecurity corpus using a similar setup. Compared to the previous experiments, we train longer for a total of 200 epochs on a larger corpus.

6.2 Downstream Tasks

We conduct downstream tasks[5] on a wider variety of cybersecurity tasks. The new tasks are described below.

CASIE (Satyapanich et al., 2020): An event detection dataset of annotated news articles for non-expert audiences. The articles are annotated for five event types: data breach, phishing, ransom, discover, and patch.

TwitterThreats (TT) (Zong et al., 2019): A binary sequence classification dataset of annotated tweets that mention threat keywords. Each tweet is annotated on whether the tweet describes a threat to the mentioned entity.

CYDEC (Yagcioglu et al., 2019): A binary sequence classification dataset of annotated tweets that mention cybersecurity keywords. Each tweet annotated on whether the tweet describes a cybersecurity-related event.

6.3 Baselines

We compare CyBERTuned with the base RoBERTa model and other cybersecurity domain PLMs. All models follow the 12-layer Transformer encoder architecture.

RoBERTa-base (Liu et al., 2019): The RoBERTa-base model that was used to initialize CyBERTuned.

[4] https://attack.mitre.org/techniques/enterprise/

[5] Note that we do not do the probing tasks, since only our model was trained on the same sources of text with the validation corpora.

	Token Class.			Sequence Class.		MCQA
	CASIE	CyNER	CySecED	CYDEC	TT	MTDB
RoBERTa-base	0.748	0.637	0.504	0.829	0.831	0.802
CyBERT	0.711†	0.462†	0.361†	0.798	0.832	0.731†
CySecBERT	0.734	0.572†	0.491	0.814	0.845†	0.808
SecureBERT	**0.753**	0.638	0.529†	0.816	0.828	0.825
CyBERTuned (Ours)	0.750	**0.654**	**0.585**†	**0.844**	**0.857**†	**0.861**†

Table 5: Experimental results of CyBERTuned and baselines on downstream cybersecurity tasks, showing median F1 scores across 10 seeds. **Boldface** values represents the best score and underlined values represents the second best score. The symbols † (positive) and † (negative) indicates statistically significant distributions from the baseline (RoBERTa-base).

CyBERT (Ranade et al., 2021): A cybersecurity BERT-based model, further pretrained on the base BERT model. The BERT vocabulary is extended by 1,000 tokens from the training corpus identified by TF-IDF.

CySecBERT (Bayer et al., 2022): A cybersecurity BERT-based model, further pretrained on the base BERT model. The model uses the BERT vocabulary.

SecureBERT (Aghaei et al., 2022): A cybersecurity RoBERTa-based model, further pretrained on the base RoBERTa model. The model uses a custom vocabulary with adjusted token weights.

6.4 Results

The experiment results are presented in the Table 5. As before, we conduct each experiment over 10 seed values and report median values.

The base RoBERTa model, despite being pretrained on the general domain, performed better than some domain PLMs in several tasks. Of the two BERT-based models, CyBERT performed poorly on most tasks while CySecBERT generally showed competitive performance. However, even CySecBERT performed poorly in the CyNER task. This is possibly due to its usage of the uncased BERT tokenizer, which not only distorts texts with special characters but is case-insensitive (possibly important for NER).

SecureBERT was the only model that beat CyBERTuned in a task. It showed high performance in the token-level tasks, suggesting some benefit of their custom tokenizer. On the other hand, CyBERTuned performed consistently, achieving best or second-best performance in all tasks.

7 Discussion

RoBERTa's performance. The base RoBERTa model achieved good performance on certain tasks after fine-tuning, often outperforming domain-pretrained models. A possible interpretation is that previously challenging cybersecurity tasks, such as binary sequence classification of threat tweets, do not require extensive domain knowledge to achieve high performance. It should be noted that the CYDEC dataset reports a human F1-score of 0.59 and the TwitterThreats dataset reports a Cohen's κ of 0.66, suggesting these models have possibly already exceeded human and annotator performance on these tasks. This underscores the need for more challenging datasets for benchmarking models in the cybersecurity domain.

NLE Classification as auxiliary task. Although MLM loss takes long to plateau, NLE classification loss plateaus quite early during pretraining. We note that the NLE classification overhead is not large, training with and without NLE classifications only had a 0.6% difference in training time. One possible method to increase training efficiency might be to drop NLE classification task after the loss plateaus. Whether this would achieve comparable performance could be investigated in further work.

NLEs of other domains. While it is common practice to remove NLEs in other domains, our investigation suggests proper modifications to training may be preferable. However, our experiments were conducted in the cybersecurity domain, where certain NLE types can contain informative content. The optimal pretraining strategy is likely different

across text domains, and dependent on informational content of NLEs.

8 Conclusion

We investigate methods to modify pretraining to suit the cybersecurity domain. We find that a strategy of selective MLM that allows for masking of semi-linguistic elements but not fully-linguistic elements with an auxiliary NLE classification task showed best performance. With these findings, we present CyBERTuned, a cybersecurity PLM with our modified pretraining methodology. The final CyBERTuned model shows strong performance across all cybersecurity downstream tasks. Our findings support the importance of adapting pretraining methodologies to suit target domains.

Limitations

Non-linguistic Element Types. The types of non-linguistic element discussed in this work represent a subset of a large set of textual data that are atypical to natural language. While there are more text types that fall under this category, our scope was limited to types that are easily identifiable to be practical for self-supervision. We note the existence of more complex types of text that are relevant to understanding cybersecurity text such as code blocks, filepaths, or log entries. We leave the detection and effective utilization of such text types for future work.

Another limitation is that the strategies tested utilized broad distinctions between SLEs and FNLEs. Due to computational restraints, it was not possible to pretrain while treating each NLE type uniquely. Therefore, even with our method that utilizes masking SLEs to improve performance, it is difficult to attribute the performance gains to specific individual NLE types. For now we focus discussions on our empirical results, and leave fine-grained analysis to future work.

Downstream tasks. There are many factors to consider when fine-tuning downstream tasks across multiple models. We attempt to find stable settings that allow all runs to be successful, but there are inconsistent runs. To mitigate this, we report median values of 10 seed values and suggest the probing task as an alternative. Since our experiments are run across multiple models, multiple tasks, and multiple hyperparameters, there may be cases of novel untested hyperparameter combinations on model-task combinations that have not been explored.

NLEs in different domains. As stated in Section 7, the findings in this work were investigated only in the cybersecurity domain. For example, the URL NLE type also occurs frequently in other domains, but might not have the similar information value of performing MLM as the cybersecurity domain. Therefore, the decision to do MLM on URL tokens could depend on the domain. An example of a domain where MLM of URL tokens might be inappropriate is in the Twitter text domain, where links are randomized by the Twitter URL shortener.

Acknowledgements

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT). (No. 2022-0-00740, The Development of Darkweb Hidden Service Identification and Real IP Trace Technology)

References

Ehsan Aghaei, Xi Niu, Waseem Shadid, and Ehab Al-Shaer. 2022. Securebert: A domain-specific language model for cybersecurity.

Md Alam, Dipkamal Bhusal, Youngja Park, and Nidhi Rastogi. 2022. Cyner: A python library for cybersecurity named entity recognition.

Stéphane Aroca-Ouellette and Frank Rudzicz. 2020. On Losses for Modern Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4970–4981, Online. Association for Computational Linguistics.

Markus Bayer, Philipp Kuehn, Ramin Shanehsaz, and Christian Reuter. 2022. Cysecbert: A domain-adapted language model for the cybersecurity domain.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, and Michael Granitzer. 2021. HateBERT: Retraining BERT for abusive language detection in English. In

Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021), pages 17–25, Online. Association for Computational Linguistics.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online.

Ilias Chalkidis, Nicolas Garneau, Catalina Goanta, Daniel Katz, and Anders Søgaard. 2023. LeXFiles and LegalLAMA: Facilitating English multinational legal language model development. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15513–15535, Toronto, Canada. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese BERT. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.

Xiang Dai, Sarvnaz Karimi, Ben Hachey, and Cecile Paris. 2020. Cost-effective selection of pretraining data: A case study of pretraining BERT on social media. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1675–1681, Online. 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*, pages 4171–4186, Minneapolis, Minnesota.

Hicham El Boukkouri, Olivier Ferret, Thomas Lavergne, and Pierre Zweigenbaum. 2022. Re-train or train from scratch? comparing pre-training strategies of BERT in the medical domain. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2626–2633, Marseille, France. European Language Resources Association.

Greg Farnham and Kees Leune. 2013. Tools and standards for cyber threat intelligence projects. *SANS Institute*, 3(2):25–31.

Diego Fernández Vázquez, Oscar Pastor Acosta, Christopher Spirito, Sarah Brown, and Emily Reid. 2012. Conceptual framework for cyber defense information sharing within trust relationships. In *2012 4th International Conference on Cyber Conflict (CYCON 2012)*, pages 1–17.

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

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced BERT with disentangled attention. *CoRR*, abs/2006.03654.

Le Hou, Richard Yuanzhe Pang, Tianyi Zhou, Yuexin Wu, Xinying Song, Xiaodan Song, and Denny Zhou. 2022. Token dropping for efficient BERT pretraining. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3774–3784, Dublin, Ireland. Association for Computational Linguistics.

Chia-Chien Hung, Anne Lauscher, Simone Ponzetto, and Goran Glavaš. 2022. DS-TOD: Efficient domain specialization for task-oriented dialog. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 891–904, Dublin, Ireland. Association for Computational Linguistics.

Ghaith Husari, Ehab Al-Shaer, Mohiuddin Ahmed, Bill Chu, and Xi Niu. 2017. Ttpdrill: Automatic and accurate extraction of threat actions from unstructured text of cti sources. In *Proceedings of the 33rd Annual Computer Security Applications Conference*, ACSAC 2017, page 103–115, New York, NY, USA.

Youngjin Jin, Eugene Jang, Jian Cui, Jin-Woo Chung, Yongjae Lee, and Seungwon Shin. 2023. DarkBERT: A language model for the dark side of the Internet. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7515–7533, Toronto, Canada. Association for Computational Linguistics.

Hyeonseong Jo, Jinwoo Kim, Phillip Porras, Vinod Yegneswaran, and Seungwon Shin. 2021. Gapfinder: Finding inconsistency of security information from unstructured text. *IEEE Transactions on Information Forensics and Security*, 16:86–99.

Panos Kampanakis. 2014. Security automation and threat information-sharing options. *IEEE Security & Privacy*, 12(5):42–51.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoit Crabbé, Laurent Besacier, and Didier Schwab. 2020. FlauBERT: Unsupervised language model pre-training for French. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2479–2490, Marseille, France. European Language Resources Association.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

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

Swee Kiat Lim, Aldrian Obaja Muis, Wei Lu, and Chen Hui Ong. 2017. MalwareTextDB: A database for annotated malware articles. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1557–1567, Vancouver, Canada.

Chen Lin, Timothy Miller, Dmitriy Dligach, Steven Bethard, and Guergana Savova. 2021. EntityBERT: Entity-centric masking strategy for model pretraining for the clinical domain. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 191–201, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Hieu Man Duc Trong, Duc Trong Le, Amir Pouran Ben Veyseh, Thuat Nguyen, and Thien Huu Nguyen. 2020. Introducing a new dataset for event detection in cybersecurity texts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5381–5390, Online.

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

Peter Phandi, Amila Silva, and Wei Lu. 2018. SemEval-2018 task 8: Semantic extraction from CybersecUrity REports using natural language processing (SecureNLP). In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 697–706, New Orleans, Louisiana.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Abir Rahali and Moulay A Akhloufi. 2021. Malbert: Using transformers for cybersecurity and malicious software detection. *arXiv preprint arXiv:2103.03806*.

Priyanka Ranade, Aritran Piplai, Anupam Joshi, and Tim Finin. 2021. Cybert: Contextualized embeddings for the cybersecurity domain. In *2021 IEEE International Conference on Big Data*, pages 3334–3342.

Soumya Sanyal, Yichong Xu, Shuohang Wang, Ziyi Yang, Reid Pryzant, Wenhao Yu, Chenguang Zhu, and Xiang Ren. 2023. APOLLO: A simple approach for adaptive pretraining of language models for logical reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6308–6321, Toronto, Canada. Association for Computational Linguistics.

Taneeya Satyapanich, Francis Ferraro, and Tim Finin. 2020. Casie: Extracting cybersecurity event information from text. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8749–8757.

Victor Sergeev. 2023. Ioc detection experiments with chatgpt.

Thomas D. Wagner, Khaled Mahbub, Esther Palomar, and Ali E. Abdallah. 2019. Cyber threat intelligence sharing: Survey and research directions. *Computers & Security*, 87:101589.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Semih Yagcioglu, Mehmet Saygin Seyfioglu, Begum Citamak, Batuhan Bardak, Seren Guldamlasioglu, Azmi Yuksel, and Emin Islam Tatli. 2019. Detecting cybersecurity events from noisy short text. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, pages 1366–1372, Minneapolis, Minnesota.

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

Jiao Yin, MingJian Tang, Jinli Cao, and Hua Wang. 2020. Apply transfer learning to cybersecurity: Predicting exploitability of vulnerabilities by description. *Knowledge-Based Systems*, 210:106529.

Jun Zhao, Qiben Yan, Jianxin Li, Minglai Shao, Zuti He, and Bo Li. 2020. Timiner: Automatically extracting

and analyzing categorized cyber threat intelligence from social data. *Computers & Security*, 95:101867.

Qihuang Zhong, Liang Ding, Juhua Liu, Xuebo Liu, Min Zhang, Bo Du, and Dacheng Tao. 2023. Revisiting token dropping strategy in efficient BERT pretraining. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10391–10405, Toronto, Canada. Association for Computational Linguistics.

Ziyun Zhu and Tudor Dumitraş. 2016. Featuresmith: Automatically engineering features for malware detection by mining the security literature. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 767–778, New York, NY, USA.

Shi Zong, Alan Ritter, Graham Mueller, and Evan Wright. 2019. Analyzing the perceived severity of cybersecurity threats reported on social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, pages 1380–1390, Minneapolis, Minnesota.

A Corpus

We construct our corpus with data from the following sources:

- **Online Security Articles**: Following work in CTI (Zhao et al., 2020) we identify news outlets, corporate blogs, and personal blogs that discuss cybersecurity content. We find a total of 60 online sources (see Appendix D). We extract text using Scrapy[6] and Selenium[7].
- **Security Paper Abstracts**: We identify security conferences that make the abstracts of accepted papers publicly available. In total, we collect 7,301 abstracts from 8 different security conferences.
- **Wikipedia Articles**: We start with the cybercrime category[8], recursively visiting its subcategories one by one and collecting all the pages under each category while discarding irrelevant pages through manual inspection. We collect a total of 3,411 pages.
- **CVE Descriptions**: The CVE database[9] is a database of publicly disclosed vulnerabilities. Each vulnerability is assigned an ID and given a short description. We process the database to remove duplicate descriptions, incomplete (reserved or unused CVEs) descriptions, and descriptions that are too short (less than 10 words). We retain a total of 184,956 CVE entries of unique and informative descriptions.

B Tokens for Probing Task

We collect a total of 556 phrases from the attack techniques and subtechniques listed in the MITRE database. For simplification, we only select single-token target words. From the phrases, we seperate into words and check if the word is in the RoBERTa tokenizer. This way we find a total of 871 target tokens. Since many common tokens such as " and" or " to" are selected in this way, we apply a simple filter by token IDs to remove common tokens (ID < 25,000). This results in a target token list of 226 tokens including " Unix" and " runtime".

[6] https://scrapy.org
[7] https://www.selenium.dev
[8] https://en.wikipedia.org/wiki/Category:Cybercrime
[9] https://cve.mitre.org/

C Experiment settings

C.1 Pretraining

To pretrain the models for Section 4, we train on 2 NVIDIA A100 80GB GPUs. We use a slightly lowered effective batch size of 2024 to accomodate the smaller corpus size, and a warmup ratio of 0.048 following RoBERTa (which uses fixed steps). Other hyperparameters regarding including learning weights, weight decay, and adam hyperparameters are kept the same as RoBERTa's.

To pretrain the full CyBERTuned model described in Section 6.1, we train on 4 NVIDIA A100 80GB GPUs on our full corpus. The hyperparameter settings are kept the same as above with the exception of the maximum epochs, which is set to 200.

C.2 Fine-tuning

For all tasks, fine-tuning is done with 20 max epochs, warmup ratio of 0.06, and an early stopping patience of 4 based on evaluation loss on the dev set. For the token classification tasks, evaluation is done every epoch. For sequence classification tasks, evaluation is done every 200 steps. For the multi-choice QA task, evaluation is done every 200 steps.

To best compare the models themselves, we keep implementations simple for downstream tasks. We use the Hugging Face (Wolf et al., 2020) implementations of each task. For token classification, sequence classification, and multichoice QA tasks, we use the AutoModelForTokenClassification, AutoModelForSequenceClassification, and AutoModelForMultipleChoice, respectively.

To simplify hyperparameter selection, we select a batch size for each task following observations that input types varied heavily on the task (such as sentences and documents). First we conducted a grid search of learning rates $\in$ {2e5, 3e5, 5e5, 1e-4} and batch sizes $\in$ {1, 2, 4, 8, 32} for the CyNER task. We identified learning rate of 5e-5 and batch size 32 worked best. This combination worked well with all tasks involving single-sentence inputs. CYDEC and TwitterThreats use this setting. For CySecED (document), we did grid search of batch sizes $\in$ {1, 2, 4, 8, 32} and find that batch size of 1 works best. CASIE also uses this setting. For MTDB (QA), we did grid search of batch sizes $\in$ {1, 2, 4, 8, 32} and find that batch size of 8 works best.

We use the train/eval/test given in the datasets

if possible (CyNER, CySecED, MTDB). If the dataset does not have splits (CASIE, TwitterThreats, CyDEC), we split randomly at a 8:1:1 ratio. Since CyNER deals with identifying exact spans while CySecED and CASIE deals with identifying event triggers, we use a stricter matching for CyNER (with seqeval[10]) but use a loose matching scheme for CySecED and CASIE.

D Article Sources

In Table 6, we list detailed online data sources from which we collect cybersecurity domain text.

Source	Type	# Pages Collected
InfoSecurity	News	23,217
ThreatPost	News	15,742
The Hacker News	News	10,049
Bleeping Computer*	News	8,852
Infosec Institute	News	6,086
Security Intelligence	News	1,824
The Record	News	1,471
Cyber Security Hub	News	902
Schneier on Security	Personal Blog	8,008
TaoSecurity Blog*	Personal Blog	3,044
Krebs on Security	Personal Blog	2,151
Darknet	Personal Blog	2,081
Ddanchev Blog*	Personal Blog	1,575
hpHosts Blog*	Personal Blog	1,057
Hexacorn Blog	Personal Blog	784
Garwarner Blog*	Personal Blog	570
Kahu Security*	Personal Blog	194
SkullSecurity*	Personal Blog	144
Carnal0wnage*	Personal Blog	124
SecNiche*	Personal Blog	94
DeepEnd Research*	Personal Blog	23
Naked Security	Corporate Blog	13,233
State of Security*	Corporate Blog	5,233
WeLiveSecurity	Corporate Blog	5,186
Palo Alto Networks	Corporate Blog	3,482
Malwarebytes	Corporate Blog	3,359
Securosis	Corporate Blog	3,302
Microsoft	Corporate Blog	2,902
Securelist	Corporate Blog	2,897
Sophos*	Corporate Blog	1,987
Sucuri*	Corporate Blog	1,718
MSRC	Corporate Blog	1,473
Spider Labs*	Corporate Blog	1,463
Webroot*	Corporate Blog	1,429
Recorded Future	Corporate Blog	1,280
Zscaler*	Corporate Blog	782
Unit42*	Corporate Blog	771
NETSCOUT	Corporate Blog	731
Radware	Corporate Blog	720
Trustwave Blog	Corporate Blog	676
Forcepoint*	Corporate Blog	665
SecureAuth	Corporate Blog	583
Trend Micro (News)*	Corporate Blog	494
Cloudflare	Corporate Blog	449
Infoblox*	Corporate Blog	403
BitDefender	Corporate Blog	400
Honeynet Project*	Corporate Blog	395
Mandiant*	Corporate Blog	355
CoreSecurity	Corporate Blog	257
Intezer*	Corporate Blog	236
Symantec Enterprise Blogs*	Corporate Blog	219
LookingGlass	Corporate Blog	214
Veracode	Corporate Blog	203
SEI (CERT/CC)*	Corporate Blog	174
FireEye*	Corporate Blog	148
CrowdStrike*	Corporate Blog	144
Trend Micro (Research)*	Corporate Blog	141
Juniper*	Corporate Blog	122
Fox IT*	Corporate Blog	109
Verisign Blog	Corporate Blog	100

Table 6: Full list of security news articles and security blogs used for corpus collection. The sources included in our pretraining subset are marked by *.

[10] https://github.com/chakki-works/seqeval