CatBERT: Context-Aware Tiny BERT for Detecting Targeted Social Engineering Emails

Younghoo Lee
Sophos AI
younghoo.lee@sophos.com

Joshua Saxe
Sophos AI
jushua.saxe@sophos.com

Richard Harang
Sophos AI
richard.harang@sophos.com

Abstract

Targeted phishing emails are a major cyber threat on the Internet today and are insufficiently addressed by current defenses. In this paper, we leverage industrial-scale datasets from Sophos cloud email security service, which defends tens of millions of customer mailboxes, to propose a novel Transformer-based architecture for detecting targeted phishing emails. Using real-world targeted phishing data as well as millions of benign customer emails for training and evaluation, we show that our proposed CatBERT (Context-Aware Tiny Bert) model achieves a 87% detection rate at a false positive rate of 1%, as compared to DistilBERT [20], LSTM (Long Short-Term Memory) [13], and logistic regression baselines which achieve 83%, 79%, and 54% detection rates respectively. Our model leverages both natural language and email header inputs, is more computationally efficient than competing transformer approaches, and we show that it is less prone to adversarial attacks which deliberately replace keywords with typos or synonyms.

CCS Concepts: • **Security and privacy → Software and application security**; • **Computing methodologies → Artificial intelligence**.

Keywords: phishing email, artificial intelligence, machine learning, BERT

ACM Reference Format:
Younghoo Lee, Joshua Saxe, and Richard Harang. 2021. CatBERT: Context-Aware Tiny BERT for Detecting Targeted Social Engineering Emails. In *KDD '21 Workshop on AI-enabled Cybersecurity Analytics, Aug 14–18, 2021, Virtual Conference.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/1122445.1122456

KDD '21 Workshop on AICA, Aug 14–18, 2021, Virtual Conference

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/1122445.1122456

1 Introduction

Social engineering attacks leveraging hand-crafted emails are a major threat vector today. Because these emails are often hand-written, individually targeted, and incorporate background research on their targets [6], they pose a significant challenge for conventional detection systems which reply on spam-like duplication between previously seen and new malicious emails.

To address these challenges, in this paper we propose a phishing detection strategy based on transformers [22], leveraging a BERT-derived approach that is trained on a self-supervised cloze task on a public corpus of documents [3] and then optimizing the language model to perform phishing detection. This transfer-learning procedure allows the network to learn a useful representation of natural language syntax and semantics before specialization in phishing detection.

To accurately detect malicious emails, we go beyond simply fine-tuning a canonical BERT transformer model in two ways. First, we integrate email header features into the flow of our network, rather than simply relying on sequential textual information. Header fields in an email provide contextual information about the communication between senders and recipients. Our network architecture combines an email's text content with this additional context and improves the classification performance over standard BERT.

Second, we have found, experimentally, that a co-mingled fine-tuning and adapter-based approach, in which we freeze most transformer block parameters and tune fully connected adapter layers, works best, and also that we can ablate a number of transformer blocks in the upstream pretrained model with no loss in accuracy [9]. These modeling choices yield a smaller and faster model able to handle a high volume of email at lower computational cost while maintaining optimal accuracy relative to multiple baselines.

The contributions of our work are listed below.

- We propose a BERT-derived phishing email detection model which combines text content with header context features, outperforming a range of approaches from both historical and deep-learning NLP. To our knowledge, ours is the first published application of transformers to phishing detection.
- Our co-mingled adapter and fine-tuning approach substantially reduces the model's complexity and achieves

a 15% smaller and 160% faster model at no substantial loss in accuracy.
- We demonstrate our results on a real-world dataset of five million emails, including real customer benign emails and fresh, present-day malicious emails seen in-the-wild.
- We show under simulation that our model is less prone to realistic adversarial attacks than a range of baselines.

The rest of the paper is organized as follows. Section 2 describes related work. In section 3, we elaborate our proposed method and then discuss experiments in section 4. The work is concluded in section 5.

2 Related work

In this section, we describe machine learning approaches for detecting phishing emails and then present related work in making Transformer-based models computationally efficient.

2.1 Machine learning approaches

Traditional approaches to machine-learning based phishing detection extract TF-IDF (Term Frequency-Inverse Document Frequency) features from word tokens in email text and use them as inputs to logistic regression and tree-based models [1, 2, 7].

Recent studies leveraged deep neural networks for detecting malicious content across various malicious file types including portable executable (PE), Zip archive, web and Android files [12, 18, 19, 21]. The malicious files in an email can be identified by the specific file detectors, but the attachment files are out of our scope in this paper.

More recent work has employed RNN (Recurrent Neural Network) and CNN (Convolutional Neural Network) models on sequential text data to detect malicious emails [13, 14]. In [14], a CNN-based model was introduced with features only from email headers for detecting spam messages. Our experiments demonstrate that our combined features both from the email's body and header outperform modern neural network models. To our knowledge, our approach is the first to leverage transformers in detecting phishing emails.

2.2 Model compression for BERT models

Transformer-based models including BERT [3] and GPT [17] have achieved great success in text classification tasks, but the full-sized models are computationally expensive and memory intensive, making them hard to use in high volume, low-latency, low-cost inference contexts. Several compressed BERT models, such as DistilBERT [20] and ALBERT [11] have been proposed to address this issue [4, 5].

DistilBERT reduces the number of Transformer blocks, and ALBERT reduces the model size by sharing Transformer blocks across layers but does not improve inference speed. Our approach is to reduce the number of Transformer blocks and replace removed ones with trainable fully connected layers (known in the literature as adapters [9]), which we find leads to more stable and improved results in both run time and detection performance.

3 Proposed method

In this section, we present our model architecture with input features and a partial fine-tuning method that improves the performance of run-time and classification over standard BERT models.

3.1 Model Architecture

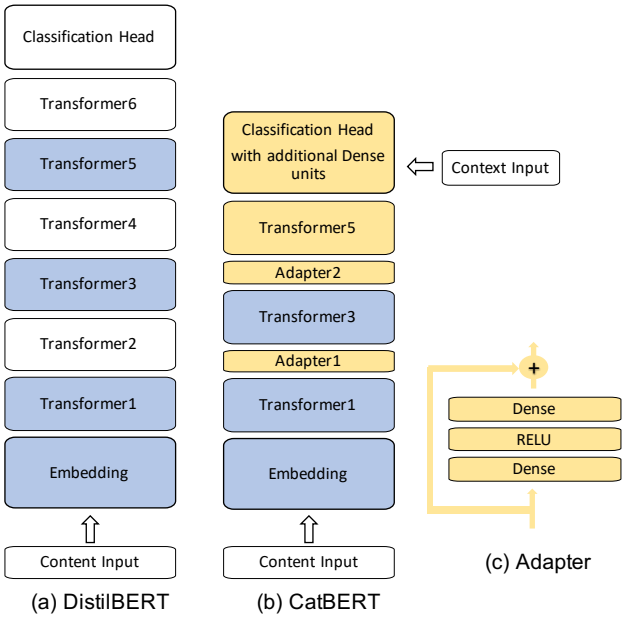


Figure 1. Our proposed CatBERT model is downsized from DistilBERT by taking odd-numbered Transformers and replacing missing Transformers with simple Adapters, which we find reduces model size with no cost to accuracy. Our adapters consists of two dense and a ReLU activation units with a residual connection. The content input from email text is fed to the Embedding block and the context input from email headers is combined in the classification head in CatBERT. The blocks in blue are not trainable on the phishing detection downstream task, whereas the yellow blocks are.

Because we target a multilingual phishing detection use case and seek a low computational cost approach, we selected DistilBERT as our base model, because it has been pre-trained with multilingual text datasets, and has been compressed from a standard BERT model. We further compress DistilBERT by reducing the number of Transformer layers and replacing removed ones with simple adapters as

shown in Figure 1. The adapter blocks are inspired by the idea from parameter-efficient transfer learning [9] which adds tiny adapters into Transformer blocks.

While replacing every other transformer block in our pretrained model with an adapter block may seem counterintuitive, we found that it works well in practice and does not significantly reduce the efficacy of the final model, while significantly reducing computational cost. Figure 1 depicts how we reduced the number of Transformer blocks. The adapter block consists of a fully connected dense unit, a Rectified linear unit (ReLU) and a second dense unit with a residual connection and the dense units have the same dimensionality as Transformers.

3.2 Content features

To extract sequential token features from the email subject and body (which we concatenate), we use BERT tokenizer which splits input character sequences into a small sub-word vocabulary of 30,522 tokens for English and 119,547 tokens for multilingual models [3]. The sub-word tokenizer minimizes out-of-vocabulary issues and inserts two special tokens: [CLS] is the first token and [SEP] is the last token for every text input.

We extract plain text but remove HTML tags from an email's subject and body, and then truncate the text from the end of the concatenated text as the first sentences often convey more important messages than the last ones. It is also the case for email replies which append previous email messages. The hidden state for [CLS] token from the last Transformer block is fed into the final classification head.

3.3 Context features

Our context-aware approach combines the text content with context data from email headers as input pairs whereas standard BERT models accept single text input. We extract the following context features which are fed into the classification head layer in Figure 1.

- Internal communication: this is a binary feature that codifies whether the sender and recipient of the communication belong to the same email domain.
- External reply: this is a binary feature that codifies whether the domains of sender and reply-to are different.
- Number of recipients and carbon copies: These are integer features that specify that number of email recipients, and the number of email carbon copies.

3.4 Partial fine-tuning

We conducted an extensive architecture search, and found that freezing the two bottom Transformer blocks and jointly fine-tuning all adapters and the top Transformer block in our architecture maximized generalization performance, as shown in our Figure 1. Surprisingly, simply removing half of the Transformer blocks and replacing them with trainable adapter blocks was sufficient to surpass the high performance of the full-sized model.

We fit the network using a binary cross entropy loss function. The loss L is defined by given the output of our model $f(\mathbf{x};\theta)$ for input $\mathbf{x}$ and label $y \in \{0,1\}$ and model parameter θ.

$$L(\mathbf{x}, y; \theta) = -y \log(f(\mathbf{x};\theta)) + (1-y)\log(1 - f(\mathbf{x};\theta))$$

We solve for $\hat{\theta}$ the optimal set of parameters that minimize the loss over the dataset:

$$\hat{\theta} = \arg\min_{\theta} \sum_{i=1}^{n} L(\mathbf{x_i}, y_i; \theta)$$

Where n is the number of samples in our dataset, and $\mathbf{x_i}$ and y_i are the feature vector of the i^{th} training sample and the label respectively.

4 Experiments

In our experiments, we used a dataset of 407,161 malicious emails observed in customer environments and threat feeds, and 3,842,772 benign emails from customer traffic. We split the samples into 70% for training, 15% for validation and 15% for test datasets based on emails' first seen times. The training dataset has 285,021 malicious and 2,697,499 benign emails, and the test dataset has 122,140 malicious and 1,145,273 benign samples. For all experiments, we set the maximum BERT token length for email text as 128 which was a sweet spot between runtime speed and detection performance. We trained neural network models using the PyTorch [15] framework with the Adam optimizer [10] and 128 sized mini batches. Our implementation of DistilBERT is from Huggingface [23]. All neural network models use the same multilingual BERT tokenizer for text features and same sized embedding layers, which is designed for each model to have a similar learning capability to handle the input tokens. The models are trained with the best hyperparameters for five epochs, which was enough for the results to converge.

4.1 Classification performance

To baseline our approach, we conducted experiments with two non-BERT models, Long Short-Term Memory (LSTM) [8] and Logistic Regression (LR) [16]. We also baselined with DistilBERT, trained with Adam and with a class-balanced batch size of 128. CatBERT, our proposed model, has three Transformer blocks with content and context features. DistilBERT contains six Transformer blocks that were used in the DistilBERT paper [20]. The LSTM model accepts word sequences as input and contains a single LSTM layer with an embedding layer that has the same size parameters as BERT. The LR model uses TF-IDF features from uni/bi-gram words.

Figure 2 compares the models' ROC (Receiver Operating Characteristic) curves, demonstrating that BERT-based models outperform non-BERT models by a large margin and

that our proposed CatBERT approach achieves the best performance. The top in Figure 2 shows the ROC curves for all test samples and the bottom is for BEC (Business Email Compromise) samples. We assigned a high sample weight for the BEC samples using hyperparameter search, which allows our model to focus on those targeted social engineering emails, as high errors will be given when those samples are miss-classified during training (see Table 1 and Table 2 of the Appendix).

We also conducted an ablation study to investigate the influence of new components in CatBERT. When one of the adapter or context layers was removed from CatBERT, there was a measurable performance drop (see Figure 3 of the Appendix).

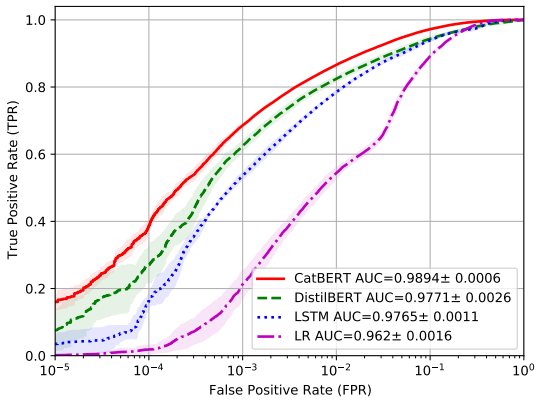
While adversaries often deform malicious emails [1], Figure 4 of the Appendix shows that our approach for the combined features with partial fine-tuning was less prone to realistic typo and synonym attacks in the black box setting where adversaries are allowed to query whether an input email is detected or not. The Figure suggests that our model still recognizes the main intent of an email message even if some of the words are replaced with typographic errors or synonyms. We created adversarial samples from the BEC samples in our test dataset using simple synonyms and typos. Although our adversarial attacks are limited to those simple types, our experiments demonstrate that CatBERT is more robust than non-BERT baseline models.

4.2 Runtime performance

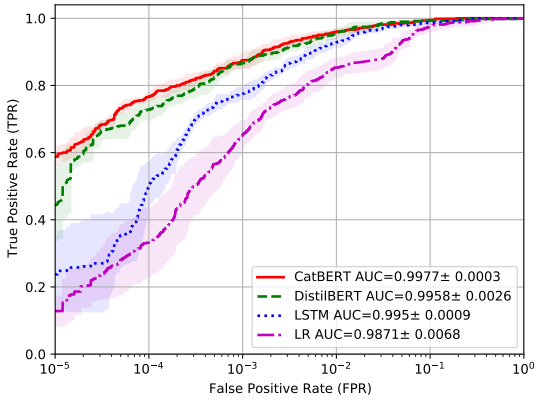
One of the challenges in applying full-sized Transformer-based models to real-time malware detection systems is the runtime performance. As millions of emails need to be processed daily on general-purpose CPU machines in our production environment, the inference speed on CPU is a critical performance metric for model deployment. DistilBERT has 135 million parameters, which is the largest model with 6 Transformer blocks, and a correspondingly long inference time. CatBERT with 3 Transformer blocks has 117 million parameters, is about 15% smaller than DistilBERT and obtains 1.6x speed up in CPU inference time (using an AWS m5.large instance type). The relatively modest reduction in parameter size for CatBERT is due to the fact that all three models use a large embedding layer with 92 million parameters, which accounts for about 70% of all parameters (see Table 3 of the Appendix).

5 Conclusion

Hand-crafted social engineering emails pose a significant challenge for traditional signature and ML detection technologies, as an individually targeted email may not share word sequences or word choices with previously seen attacks. We introduce an efficiently downsized BERT model by fine-tuning a pre-trained, highly pruned BERT model with additional context features from email headers that can detect targeted phishing emails, even in the presence of deliberate misspellings and attempted evasions. Our approach outperforms strong baseline models with adapter and context layers. CatBERT is 15% smaller and 160% faster than DistilBERT which is already 40% smaller than standard BERT.



(a) All samples



(b) BEC samples

Figure 2. Mean ROC curves and standard deviation for CatBERT and baseline models. Top plot compares four models with all test samples and bottom one compares only with BEC (Business Email Compromise) test samples respectively. Mean and standard deviation are computed over five runs.

References

[1] Bhowmick, Alexy & Hazarika, Shyamanta. Machine Learning for E-mail Spam Filtering: Review,Techniques and Trends. In *arXiv preprint arXiv:1606.01042*, 2016.

[2] Asaf Cidon and Lior Gavish, Itay Bleier, Nadia Korshun, Marco Schweighauser and Alexey Tsitkin. High Precision Detection of Business Email Compromise. In *USENIX Security Symposium*, 2019.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *arXiv preprint arXiv:1810.04805*, 2018.

[4] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *arXiv preprint arXiv:1510.00149*, 2015.

[5] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *arXiv preprint arXiv:1503.02531*, 2015.

[6] Grant Ho and Aashish Sharma and Mobin Javed and Vern Paxson and David Wagner. Detecting Credential Spearphishing in Enterprise Settings. In *USENIX Security Symposium*, 2017.

[7] Ho, Grant & Cidon, Asaf & Gavish, Lior & Schweighauser, Marco & Paxson, Vern & Savage, Stefan & Voelker, Geoffrey & Wagner, David. Detecting and Characterizing Lateral Phishing at Scale. In *Proceedings of the 28th USENIX Conference on Security Symposium*, 2019.

[8] Hochreiter, S., Schmidhuber, J. Long short-term memory. In *Neural Computation 9(8), 1735–1780*, 1997.

[9] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, Sylvain Gelly. Parameter-Efficient Transfer Learning for NLP. In *ICML*, 2019.

[10] Diederik Kingma and Jimmy Ba Adam: A method for stochastic optimization. In *arXiv preprint arXiv:1412.6980*, 2014.

[11] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, Radu Soricut. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *arXiv preprint arXiv:1909.11942*, 2019.

[12] Younghoo Lee, Joshua Saxe, Richard Harang. SeqDroid: Obfuscated Android Malware Detection Using Stacked Convolutional and Recurrent Neural Networks. In *Deep Learning Applications for Cyber Security. Advanced Sciences and Technologies for Security Applications. Springer, (pp. 197-210)*, 2019.

[13] Halgaš, Lukáš & Agrafiotis, Ioannis & Nurse, Jason. Catching the Phish: Detecting Phishing Attacks Using Recurrent Neural Networks (RNNs). In *Information Security Applications, pages 219-233*, 2020.

[14] Benkovich, Nikita & Dedenok, Roman & Golubev, Dmitry. DeepQuarantine for Suspicious Mail. In *arXiv preprint arXiv:2001.04168*, 2020.

[15] Paszke, Adam and Gross, Sam and Massa, Francisco and Lerer, Adam and Bradbury, James and Chanan, Gregory and Killeen, Trevor and Lin, Zeming and Gimelshein, Natalia and Antiga, Luca and Desmaison, Alban and Kopf, Andreas and Yang, Edward and DeVito, Zachary and Raison, Martin and Tejani, Alykhan and Chilamkurthy, Sasank and Steiner, Benoit and Fang, Lu and Bai, Junjie and Chintala, Soumith PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, 2019.

[16] Peng, Joanne & Lee, Kuk & Ingersoll, Gary. An Introduction to Logistic Regression Analysis and Reporting. In *Journal of Educational Research*, 2002.

[17] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[18] E. M. Rudd, R. Harang and J. Saxe. MEADE: Towards a Malicious Email Attachment Detection Engine. In *2018 IEEE International Symposium on Technologies for Homeland Security (HST), Woburn, MA, 2018, pp. 1-7*, 2018.

[19] Rudd, E.M., Ducau, F.N., Wild, C., Berlin, K. and Harang, R. ALOHA: Auxiliary Loss Optimization for Hypothesis Augmentation. In *In 28th USENIX Security Symposium (USENIX Security 19) (pp. 303-320)*, 2019.

[20] Victor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *arXiv preprint arXiv:1910.01108*, 2019.

[21] Joshua Saxe, Richard Harang, Cody Wild, Hillary Sanders. A deep learning approach to fast, format-agnostic detection of malicious web content. In *arXiv preprint arXiv:1804.05020*, 2018.

[22] Vaswani, Ashish and Shazeer, Noam and Parmar, Niki and Uszkoreit Attention is all you need In *arXiv preprint arXiv:1706.03762*, 2017.

[23] 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, Alexander M. Rush HuggingFace's Transformers: State-of-the-art Natural Language Processing. In *arXiv preprint arXiv:1910.03771*, 2019.

A Experiment results

Table 1. AUC and TPRs on all test samples for four FPRs. Mean and standard deviation results are aggregated over five runs and best ones are shown in bold.

	AUC	FPR			
		0.0001	0.001	0.01	0.1
CaTBERT	**0.9894 ± 0.0006**	**0.3884 ± 0.0129**	**0.6859 ± 0.0054**	**0.8663 ± 0.0017**	**0.9721 ± 0.0021**
DistilBERT	0.9771 ± 0.0026	0.2696 ± 0.0524	0.6245 ± 0.0156	0.8251 ± 0.0086	0.9436 ± 0.0066
LSTM	0.9765 ± 0.0011	0.1674 ± 0.0239	0.5360 ± 0.0096	0.7852 ± 0.0048	0.9390 ± 0.0033
LR	0.9620 ± 0.0016	0.0186 ± 0.0125	0.2127 ± 0.0406	0.5434 ± 0.0165	0.8918 ± 0.0045

Table 2. AUC and TPRs on test BEC samples for four FPRs. Mean and standard deviation results are aggregated over five runs and best ones are shown in bold.

	AUC	FPR			
		0.0001	0.001	0.01	0.1
CaTBERT	**0.9977 ± 0.0003**	**0.7658 ± 0.0116**	**0.8752 ± 0.0245**	**0.9590 ± 0.0100**	**0.9949 ± 0.0042**
DistilBERT	0.9958 ± 0.0026	0.7282 ± 0.0272	0.8667 ± 0.0128	0.9521 ± 0.0139	0.9932 ± 0.0034
LSTM	0.9950 ± 0.0009	0.5060 ± 0.0652	0.7761 ± 0.0174	0.9282 ± 0.0139	0.9863 ± 0.0042
LR	0.9848 ± 0.0096	0.2821 ± 0.1590	0.7077 ± 0.0598	0.8342 ± 0.0207	0.9863 ± 0.0116

Table 3. Comparison of model size and inference speed, CatBERT achieved the best AUC and inference speed.

	DistilBERT	CatBERT
Number of Transformers	6	3
Number of total parameters (millions)	135 (1.2x)	**117** (1x)
Number of non-embedding parameters (millions)	43 (1.7x)	**25** (1x)
Inference time on CPU (milliseconds)	130 (1.6x)	**79** (1x)
AUC	0.9771	**0.9894**

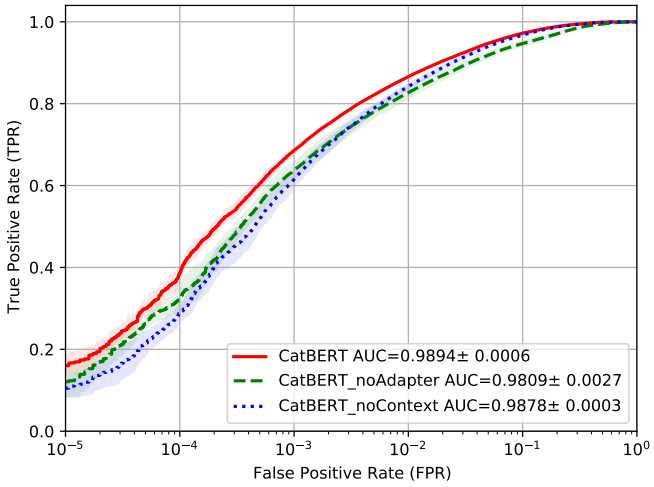
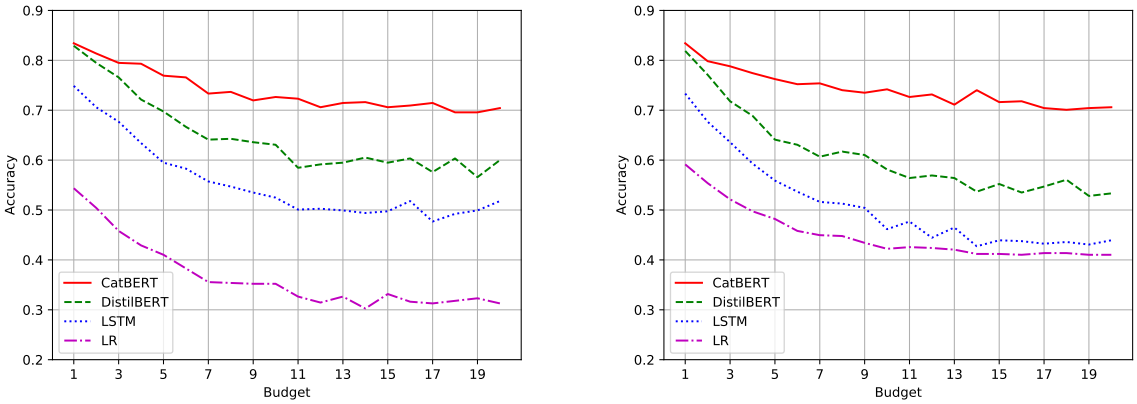


Figure 3. The ROC curves for CatBERT ablation study. Red, green and blue line shows CatBERT, CatBERT without adapters and CatBERT without context input respectively. Mean and standard deviation are computed over five runs.



(a) Synonyms attacks

(b) Typo attacks

Figure 4. The mean detection accuracy against adversarial attacks over five runs. Left shows the accuracy against synonym attacks and right shows the accuracy against typo attacks. The budget is the maximum number of synonym or typo attacks where the first n budget words are replaced with attack words. CatBERT is more robust than non-BERT baseline models.