Lightweight Transformer-Based Phishing Email Detection: A DistilBERT Approach

1st Muhammad Tayyab Jabbar
Dept. of Information and Communication Engineering
The Islamia University of Bahawalpur
Bahawalpur, Pakistan
fastn8555@gmail.com

2nd Abi Saloom Bhatti
Dept. of Information and Communication Engineering
The Islamia University of Bahawalpur
Bahawalpur, Pakistan
abisaloombhatti6@gmail.com

3rd Aoun Muhammad
Dept. of Information and Communication Engineering
The Islamia University of Bahawalpur
Bahawalpur, Pakistan
aoun.muhammad@iub.edu.pk

4th Sana Tariq
Dept. of Information and Communication Engineering
The Islamia University of Bahawalpur
Bahawalpur, Pakistan
Sana.tariq@iub.edu.pk

***Abstract*—Phishing email is one of the most common entry points for a cyberattack, and it works by manipulating the reader rather than exploiting software. Transformer models classify such text well, but fine-tuning a full-size BERT model usually needs hardware that small research groups do not have. This paper describes a reproducible, low-resource pipeline for phishing email detection built on a fine-tuned DistilBERT classifier, and it places that model between a cheap classical baseline and a full BERT-base teacher with proper statistical validation. On a near-duplicate-cleaned test set of 2,495 emails, averaged over five seeds, DistilBERT reached 98.6% $\pm$ 0.2% accuracy and a phishing-class F1 of 0.980 $\pm$ 0.002, training in about three minutes on one NVIDIA T4 GPU.A McNemar test shows that DistilBERT is basically the same as a BERT-base model.Also DistilBERT trains about half as long as well. It is also substantially superior to TF-IDF + logistic regression baseline with an accuracy of more than 0.001.**

The DistilBERT model is chosen due to its good performance and not being too small.

To test the models, we also performed a test to determine the efficacy of the models on data. We discovered that a model which is trained on a different set of data is not very useful for another set of data. For example a model that is trained on our set of data only gets 0.84 F1 on a different public set of data. The precision of this model also goes down from 0.98 to 0.74.. A model trained on a larger data set performs equally well on another set of data.

***Index Terms*—phishing detection, DistilBERT, TinyBERT, transformer fine-tuning, NLP, email security, reproducible research**

I. Introduction

The Anti-Phishing Working Group recorded 4,744,699 phishing attacks in 2022, the highest annual total it had reported at the time and about 1.7 times the 2021 figure [1]. Email remains the dominant delivery channel for these attacks [2]. What makes phishing hard to stop is that it attacks the person, not the software: a message only has to talk someone into clicking a link or typing a password. Because the text itself is the weapon, filters built on fixed rules or blocklists lose ground as attackers reword their messages to look like ordinary mail [3].

Early detectors paired hand-built features with a classifier such as naive Bayes, a support vector machine, or a boosted tree. These features count words or read header fields, so they cannot tell that "please verify your account" is harmless in a real bank notice but suspicious in a forged one; to a bag-of-words model the two are identical [19]. Reviews of these methods report that their accuracy stops climbing once attackers change their wording to slip past known patterns [3]. Neural models remove some of this brittleness by learning their own features [4], [5], but the convolutional and recurrent encoders used in early work read text in one direction or inside a fixed window, so they miss long-range context.

BERT built on self-attention with deep bidirectional pre-training and set new results across text-classification tasks [7]. The cost is the catch. Fine-tuning BERT-base, with its 110 million parameters, needs a GPU that many student teams and small security units cannot get hold of. DistilBERT is a smaller model trained to copy BERT through knowledge distillation, and its authors report that it keeps about 97% of BERT's accuracy on downstream tasks at lower cost [8]. One contribution of this paper is to test that claim directly on phishing email: we find DistilBERT keeps 99.5% of BERT-base's F1 here, with no statistically significant gap, while training in about half the time.

Work on transformer-based phishing detection has grown quickly, yet the published systems are often hard to reproduce. Many depend on private corporate email, fixed local file paths, or training across more than one GPU [17]. Our aim is narrow and practical: to show that the whole process. We make five

DOI: https://doi.org/10.67535/tsp.000003.023
Published under the ICACNC 2026 Proceedings

contributions.

- A fine-tuning pipeline for DistilBERT that pulls a public phishing email corpus straight from the HuggingFace Hub, with no manual file transfers, and trains one model on the Colab free tier in about three minutes.
- We are doing a comparison of four methods on the same test set. This test set has been cleaned to remove duplicates. The four methods are: a model that uses TF-IDF and logistic regression, TinyBERT-4L DistilBERT and a BERT-base teacher model. We ran each of these methods five times. Took the average to get the results.
- Paired McNemar significance tests between models, which establish that DistilBERT beats the classical floor, ties its BERT-base teacher, and beats TinyBERT, rather than leaving these conclusions to eyeballed bar heights.
- A reproducibility package: pinned library versions, fixed seeds, a near-duplicate deduplication step, and the released notebook that produced every number in this paper.
- A leakage-controlled cross-corpus test: after removing emails shared between two public corpora, we train on each and test on the other, and report the resulting generalization gap, which turns out to be large and asymmetric.

Relative to the closest prior work, Uddin *et al.* [14], who fine-tune a distilled transformer for phishing email detection and add post-hoc explanations, our addition is not a new model. It is a reproducible free-tier recipe, a baseline-anchored and significance-tested comparison, and a clear answer to a practical question: which size of model is actually worth running on this task.

The rest of this paper is organized in the way. Section II looks at what other people have done before us. Section III explains how we did our work. Section IV talks about the data we used and how we set up our experiments. Section V shares our results. What we think they mean. Section VI sums everything up.

II. Related Work

A. Limits of Classical Machine Learning

People usually detect phishing by using features and a supervised classifier.. The problem is that these features do not work well after a while. Kustiawan and Ghauth [19] looked at how to detect phishing websites. Found that the features they used had to be changed often. This is because the features that worked for one group of attacks did not work for the group. Ahmad *et al.* [3] and his team found the thing when they tried different methods like naive Bayes and support vector machines. They found that these methods stopped working once the attackers changed their tactics. However a simple model that uses a bag-of-words approach can still work well if it is set up correctly. That is why we used this kind of model as a baseline to compare with the methods. We wanted to see if the other methods, including the transformer models like TinyBERT-4L DistilBERT and BERT-base teacher would work better, than this model.

B. Transformer-Based Phishing and Spam Detection

The shift to learned representations runs through the transformer. Vaswani *et al.* [6] introduced self-attention, which weighs every token against every other token instead of reading inside a fixed window, and Devlin *et al.* [7] turned that design into BERT, a bidirectional model that transfers to classification after light fine-tuning; Rathore *et al.* [5] place such pre-trained models at the centre of current practice. Applied to email, BERT-family models consistently beat from-scratch classifiers: Shazad *et al.* [16] fine-tune BERT for spam detection by transfer learning, Gaurav *et al.* [17] build a cloud-based BERT detector with an eye on deployment cost, and Alauthman *et al.* [15] push on robustness by training against contrastive and generative attacks.The paper that is closest, to this one is the work of Uddin *et al.* [14]. They were using a type of computer code known as a distilled transformer to aid in identifying phishing e-mails. This program may also provide information for a human to understand why it considers an email to be phishing.

C. Lightweight and Distilled Models

Knowledge distillation trains a small student to reproduce a larger teacher, keeping most of the accuracy at lower cost, and two students dominate the literature. DistilBERT [8] distils at the pre-training stage, while TinyBERT [9] distils at both the pre-training and the task-specific stage. Vakili *et al.* [12] confirm that distilled BERT models hold up on classification, and Yang *et al.* [13] add multi-level alignment to recover more of the teacher's accuracy. That a compact BERT works on short-text binary problems is shown by Kaliyar *et al.* [18] on fake-news detection, a task close in shape to phishing classification.

D. Research Gap

Two gaps frame this work. First, reproducible pipelines tend to assume either a manual dataset download or a private corpus, so there is no end-to-end recipe a reader can run on free hardware without setup work [17]. There is another issue that can only be exacerbated. Furthermore, Hazell [20] demonstrated that language models can generate highly convincing phishing emails, complicating detection. This complicates the task of determining what is genuine and what isn't in the emails.

While Uddin *et al.* [14] effectively identified phishing emails, their approach lacked a seamless, download-free interface and omitted scalability testing across varied dataset sizes. In this paper, the problem is attempted to be solved and a beginning is made in the second one. We tried it on several sets of data and the results vary greatly from one data set to another. Sometimes a program that appears to be successful with one data set fails to be successful with another set. We discovered this (Section V-D) can help to differentiate about 15 points, and this is significant to know since the phishing email is becoming more difficult to detect.

III. Methodology

Fig. 1 shows the pipeline from raw email to prediction.

A. Problem Formulation

We treat phishing detection as binary sequence classification. An email is tokenized into a sequence $x = [x_1, \ldots, x_n]$ with $n \leq 128$, and the model learns a function $f : x \mapsto y \in \{0, 1\}$, where $y = 1$ marks a phishing message and $y = 0$ a legitimate one. The goal is to catch as many phishing messages as possible while keeping the false-positive rate on legitimate mail low.

B. Architecture

The transformer models are encoder-only. The input, prefixed with a `[CLS]` token, passes through L stacked self-attention blocks, and the contextual `[CLS]` embedding $h_{[\mathrm{CLS}]} \in \mathbb{R}^d$ is the fixed-length representation handed to the classifier. DistilBERT uses L=6, d=768 (about 66 M parameters); TinyBERT-4L uses L=4, d=312 (about 14.5 M); BERT-base uses L=12, d=768 (about 110 M). The classical baseline replaces the encoder with a TF-IDF vector over word 1–2 grams.

C. Classification Head and Training Objective

For the transformers, the `[CLS]` embedding passes through a dropout layer and a linear layer to produce two class logits, $z = W\,\mathrm{Dropout}(h_{[\mathrm{CLS}]}) + b$, with $W \in \mathbb{R}^{2\times d}$. A softmax turns the logits into class probabilities, and the model is trained with categorical cross-entropy over the two classes:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=0}^{1} \mathbb{1}[y_i = k] \log P(y = k \mid x_i), \qquad (1)$$

where

$$P(y = k \mid x_i) = \frac{e^{z_{i,k}}}{\sum_j e^{z_{i,j}}}. \qquad (2)$$

This matches the released code, which uses a two-logit softmax head (`AutoModelForSequenceClassification`, `num_labels=2`) with the default cross-entropy loss. At inference the predicted label is $\arg\max_k P(y{=}k \mid x)$, a 0.5 threshold on the phishing probability; Section V-B discusses moving that threshold. Optimization uses AdamW with linear warmup then linear decay, and early stopping with patience two on validation F1.

D. Input Preprocessing

There are 5 pre-tokenization steps. The first is that HTML markups are removed, One `[URL]` token is replaced with hyperlinks to help the model know a hyperlink doesn't require a learning of any temporary domains; second, The white space normalization is done and thirdly, the text is tokenized with using, The model's WordPiece tokenizer; fourth, text is truncated/padded up to 128 tokens. If you cap the number of tokens to 512, not 128, you're limiting. It has quadratic attention complexity and supports a batch size of 16 bits to fit. Installed in the 16 GB memory of the T4 GPU. Replacing URLs with a generic [URL] token prevents the model from overfitting to ephemeral domains, though it deliberately removes a feature relied upon by URL-aware detectors. This will be discussed again in Section V-E.

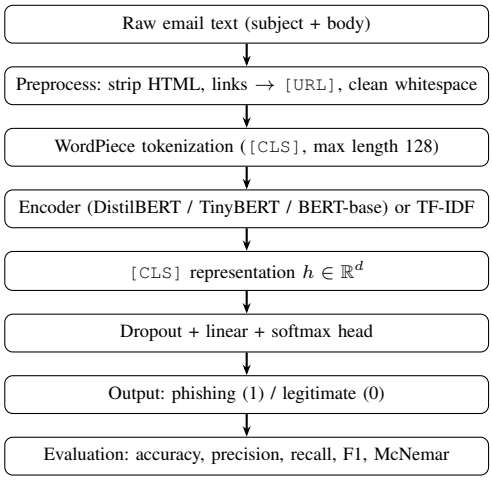

Fig. 1. End-to-end phishing email detection pipeline.

E. Evaluation Protocol

The corpus is imbalanced, so we treat F1 on the phishing class as the primary metric and keep accuracy secondary. Each transformer is fine-tuned five times with seeds $\{13, 21, 42, 87, 100\}$, and we report the mean and standard deviation of every metric. To decide whether a gap between two models is real rather than seed noise, we run McNemar's test on their paired predictions over the shared test set (the correct paired test when two classifiers are scored on the same examples); we report the discordant counts b and c and the p-value. The TF-IDF baseline is deterministic on a fixed split, so it is reported as a single value.

For the cross-corpus test (Section V-D) we add a second public corpus and train DistilBERT on one corpus, then evaluate it on the other. Because public phishing corpora share lineage, we first remove from each evaluation set any email that is a near-duplicate (MinHash, Jaccard ≥ 0.85) of anything in the other corpus's training split, so the cross-domain score is measured only on genuinely unshared mail. This protocol uses a single seed (42) and an 85/15 split, so its in-domain numbers are point estimates consistent with the five-seed results in Table III rather than a separate sweep.

IV. Dataset and Experimental Setup

A. Dataset

The public dataset of phishing emails is used, hosted by HuggingFace The Hub [10] is a set of bodies of emails from several old public datasets. We performed two deduplication passes before splitting: exact duplicate removal, followed by near-duplicate filtering using MinHash and Locality-Sensitive Hashing with a Jaccard similarity threshold of ≥ 0.85. Due to a 0.85 Jaccard similarity threshold, MinHash and locality-sensitive hashing was used. To create a set of nearly-identical messages from the source datasets, using a template. Table I gives the counts. The held-out test set is 2,495 emails, 1,618 legitimate and 877 phishing, so phishing is about 35% of the test data; a trivial all-legitimate classifier would score 64.9%

accuracy and zero recall, which is why we select on phishing-class F1.

For the cross-corpus test we use a second public corpus, the text subset of the `ealvaradob/phishing-dataset` [11], which aggregates email and short-text messages from different sources and uses the same two-column `text`/`label` format. After cleaning and exact-plus-near-duplicate removal it holds 19,939 messages (12,452 legitimate, 7,487 phishing); we cap each corpus at 16k stratified samples to bound runtime. We call the main corpus [10] A and this second corpus B. The two share substantial lineage: when we later remove from each test split any message that near-matches the other corpus's training split, about three quarters of each cross-evaluation set is discarded, leaving 598 B-messages and 485 A-messages that are genuinely unshared. The cross-corpus results are reported only on those unshared remainders.

B. Baselines

The cost-efficient DistilBERT is sandwiched between a low budget and an expensive ceiling. The vectorizer is a TF-IDF vectorizer over the word vectorizer. This is a 1–2 grams (mindf =2, sublinear term frequency, 50k features) data used for the standard linear model.the same data as for the standard linear model, into a logistic-regression classifier. Splits in no time on CPU. The base-uncased version is BERT-base (the base is un-cased). The hyperparameters have been carefully tuned with the same settings as DistilBERT, so the "keeps most of BERT's accuracy" relationship is measured This task was not quoted, but is one performed by them. TinyBERT-4L is included as the following the smallest transformer.

C. Hardware and Software

All experiments ran on the Google Colab free tier with a single NVIDIA T4 GPU (16 GB). The pinned stack was Python 3.12, PyTorch 2.11.0, Transformers 5.10.2, Datasets 5.0.0, scikit-learn 1.9.0, statsmodels 0.14.6, and datasketch 1.10.0; the full `requirements.txt` is released with the notebook. Mixed-precision training (fp16) was enabled to fit the T4 and use its Tensor Cores.

D. Training Configuration

Table II lists the hyperparameters. We ran no hyperparameter search; the values follow common DistilBERT fine-tuning practice, set conservatively for the Colab budget, and the same settings were used for all three transformers. The checkpoint with the best validation F1 was kept for the final test evaluation.

V. Results and Discussion

A. Classification Performance

Table III reports all four models on the shared test set, with the three transformers averaged over five seeds and the deterministic TF-IDF baseline as a single value. DistilBERT reached 98.6% $\pm$ 0.2% accuracy and 0.980 $\pm$ 0.002 phishing-class F1. BERT-base was a touch higher at 98.9% $\pm$ 0.1% and 0.985 $\pm$ 0.001, and DistilBERT therefore retains 99.5% of BERT-base's F1. The two cheap models sat well below:

TABLE I
DATASET SPLIT (AFTER EXACT AND NEAR-DUPLICATE REMOVAL)

Split	Total	Legitimate	Phishing
Training†	11,638	7,561	4,077
Validation†	2,494	1,620	874
Test	2,495	1,618	877
Total†	16,627	10,799	5,828

†After exact and near-duplicate removal, stratified 70/15/15 split.

TABLE II
HYPERPARAMETER CONFIGURATION

Parameter	Value
Transformer checkpoints	distilbert-base-uncased; TinyBERT-4L-312D; bert-base-uncased
Classical baseline	TF-IDF (1–2 grams) + logistic regression
Max sequence length	128 tokens
Train / eval batch size	16 / 64
Learning rate	2×10^{-5}
Epochs	3
Warmup ratio / decay	0.10 / linear
Weight decay	0.01
Optimizer	AdamW
Mixed precision (fp16)	Yes (NVIDIA T4)
Early stopping	Patience 2 (validation F1)
Seeds	13, 21, 42, 87, 100 (5 runs per model)
Near-duplicate filter	MinHash/LSH, Jaccard ≥ 0.85

TF-IDF plus logistic regression at 0.967 F1 and TinyBERT-4L at 0.960 $\pm$ 0.005.

B. Confusion Matrix

Fig. 2 shows the count and row-normalized confusion matrices for DistilBERT at seed 42. The model labels 99.2% of legitimate email correctly and catches 98.1% of phishing, missing 17 phishing messages and raising 13 false alarms on this run. Because the two error types are close in size, the 0.5 decision threshold can be shifted to trade one for the other depending on whether a deployment cares more about missed phishing or false alarms. The missed messages tend to carry no obvious malicious link and read like routine account notices, which is the kind of content sender-level metadata would help with.

C. Which Model Is Worth Running?

Fig. 3 and Table V answer the practical question with paired McNemar tests on the seed-42 predictions.

Three things follow. First, **the classical floor beats the smallest transformer.** TF-IDF plus logistic regression (0.967 F1) edges out TinyBERT-4L (0.960 F1) and trains in 4.8 seconds on CPU against TinyBERT's 81 seconds on the GPU. On this task there is no reason to reach for the tiny transformer; the bag-of-words model is both faster and more accurate. Second,**DistilBERT is worth its cost over the floor:** beats TF-IDF is significantly (p = 4.6×104) different from the null hypothesis.) and roughly halves from 43-21, the missed phishing. Third, DistilBERT ties The difference between the **DistilBERT ties its teacher:** not significant (p = 0.57, and

TABLE III
TEST-SET PERFORMANCE (MEAN ± STD OVER 5 SEEDS; TF-IDF IS DETERMINISTIC)

Model	Accuracy	Precision	Recall	F1 (phish)
TF-IDF + LogReg	0.977	0.983	0.951	0.967
TinyBERT-4L	0.972 ±.003	0.970 ±.008	0.951 ±.017	0.960 ±.005
DistilBERT	0.986 ±.002	0.983 ±.006	0.976 ±.008	0.980 ±.002
BERT-base	**0.989** ±.001	**0.991** ±.002	**0.978** ±.003	**0.985** ±.001

TABLE IV
MEAN ERROR COUNTS AND COST (TEST SET; PER-MODEL AVERAGES OVER SEEDS)

Model	Missed phish	False alarms	Train (s)	Infer (s)
TF-IDF + LogReg	43.0	14.0	**4.8**	**0.01**
TinyBERT-4L	43.4	25.8	81.1	2.02
DistilBERT	20.8	14.6	186.2	2.55
BERT-base	**19.0**	**7.4**	370.6	4.90

even the discordant counts are slightly more. slightly toward BERT, 12 versus 16), but DistilBERT trains in about half the time (186 versus 371 seconds) . DistilBERT is therefore the recommended operating point: significantly better than the cheap options, statistically level with the expensive one, at half the teacher's training cost. The DistilBERT versus TinyBERT gap is also significant ($p = 1.9\times10^{-7}$), confirming the ordering is not seed noise.

D. Cross-Corpus Generalization

Every number above is in-domain: train and test come from the same corpus. The more important question for deployment is whether a model trained on one public corpus works on email from a different source. Table VI and Fig. 4 answer it, after the leakage removal described in Section III-E left 598 unshared B-messages and 485 unshared A-messages to test on. The result is large and asymmetric. A DistilBERT trained on corpus A scores 0.986 F1 in-domain but only **0.837** on corpus B, a 15-point drop; its precision falls from 0.98 to 0.74, meaning it flags roughly a third of the unfamiliar legitimate mail as phishing (98 false alarms in 598 messages). A DistilBERT trained on the broader corpus B, by contrast, scores 0.981 in-domain and **0.982** on corpus A, with essentially no drop.

Two lessons follow. First, a single-corpus headline overstates robustness: the 98.6% the rest of this paper reports for corpus A does not survive a move to a different source, and a reader of the 'range' is not moved to a different source. should treat in-domain phishing numbers, including ours, as it should treat any other number. upper bounds. Secondly, the asymmetry states that the training corpus The same architecture is more general than the model: or fails, based on which dictionary it is based on. The wider the corpus, the more it costs to buy transfer. The failure mode is informative: the model overfits to its training data's specific definition of legitimate mail, flagging unfamiliar but benign emails as suspicious. This high false-positive rate is unacceptable for deployed filters.

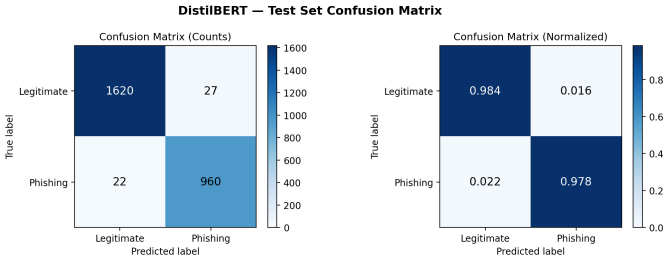

Fig. 2. DistilBERT test-set confusion matrices at seed 42 (left: counts; right: row-normalized).

TABLE V
MCNEMAR TESTS ON PAIRED PREDICTIONS (SHARED TEST SET, SEED 42). b: FIRST MODEL RIGHT, SECOND WRONG; c: THE REVERSE.

Comparison	b / c	χ^2	p-value
DistilBERT vs. TinyBERT-4L	54 / 11	27.14	1.9×10^{-7}
DistilBERT vs. TF-IDF+LogReg	41 / 14	12.29	4.6×10^{-4}
DistilBERT vs. BERT-base	12 / 16	0.32	0.57

E. Discussion

A few points are worth drawing out for anyone repeating the work.

Selecting on F1 changes which model you keep. With phishing at 35% of the test data, a trivial all-legitimate classifier already scores 64.9% accuracy and zero recall. Picking checkpoints by validation accuracy would quietly favour models that trade phishing recall for a higher overall number; selecting on validation F1 avoids that.

Variance is not negligible, especially for the small models. TinyBERT's recall varied by ±0.017 across seeds, wide enough that a single run could have made it look much better or much worse than it is. This is the clearest argument for reporting multiple seeds: the one-run version of this study would have given a misleading impression of how stable the small models are.

Mixed precision is what makes this fit. Running in fp16 kept memory low enough that batch size 16 at length 128 trained without gradient-accumulation tricks on the T4, and it sped up each step on the Tensor Cores. We recommend it as the default for this kind of fine-tuning on Colab.

Threat-model realism. The 35% phishing rate is far higher than a real inbox, where phishing is rare, so the absolute error counts here do not translate to a deployment base rate; at a realistic 1% prevalence the same false-positive rate produces many more false alarms per true catch, which is the operating point a deployed filter actually lives at. The models also read only message text, ignoring sender, headers, and attachments. The cross-corpus result in Section V-D sharpens this: the gap there is not the LLM-written phishing the introduction warns about but a plainer one, an ordinary second corpus already breaks an A-trained model, so the move to a genuinely modern, LLM-generated corpus can only widen it. Testing against such a corpus is the natural next experiment, and the same free-tier recipe and the released cross-corpus notebook carry it directly.

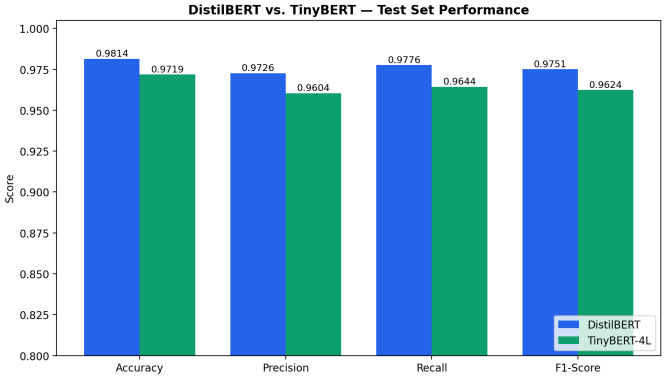

Fig. 3. Test-set metrics, mean ± std over five seeds (TF-IDF is deterministic, so its bars have no error). The widest error bar is TinyBERT recall.

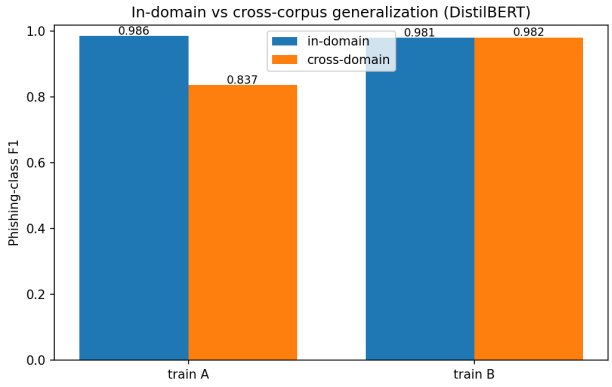

Fig. 4. In-domain versus cross-corpus phishing-class F1. The model trained on the narrower corpus A fails to transfer; the one trained on the broader corpus B transfers cleanly.

TABLE VI
CROSS-CORPUS RESULTS (DISTILBERT, SEED 42). CROSS ROWS ARE EVALUATED ONLY ON MESSAGES NOT SHARED WITH THE TRAINING CORPUS.

Train on	Test on	n	Acc	Prec	Rec	F1
A	A (in-domain)	2061	0.988	0.984	0.988	0.986
A	B (cross)	598	0.823	0.735	0.971	**0.837**
B	B (in-domain)	2205	0.983	0.979	0.984	0.981
B	A (cross)	485	0.990	0.985	0.978	**0.982**

Reproducibility checklist. Five seeds ($\{13, 21, 42, 87, 100\}$) on a single NVIDIA T4 (16 GB) on the Colab free tier, fp16 enabled, exact and near-duplicate deduplication before a stratified 70/15/15 split, the pinned versions in Section IV-C, and the hyperparameters in Table II. The notebook that produced every number in this paper is released as supplementary material.

VI. CONCLUSION

We presented a reproducible phishing email detection pipeline built on a fine-tuned DistilBERT classifier and designed for the Google Colab free tier, and we placed it between a classical baseline and a BERT-base teacher with multi-seed variance and paired significance tests. On a near-duplicate-cleaned test set of 2,495 emails, averaged over five seeds, DistilBERT reached 98.6% accuracy and 0.980 phishing-class F1 in about three minutes of T4 time. It significantly outperformed a TF-IDF plus logistic-regression floor and was statistically indistinguishable from BERT-base while training in half the time, whereas TinyBERT-4L was beaten by the classical baseline on both speed and accuracy. The practical message is that the distilled-but-not-tiny model is the operating point worth running on modest hardware. The more consequential result is the cross-corpus one: a model trained on our main corpus dropped to 0.84 F1 on a different public corpus through a precision collapse, while a model trained on the broader corpus transferred without loss. In-domain phishing numbers, including the headline above, are therefore upper bounds, and the breadth of the training corpus matters more than the last point of accuracy. The natural next step is a genuinely modern, LLM-generated corpus, which the released cross-corpus notebook is built to take as a drop-in second dataset.

REFERENCES

[1] F. P. E. Putra, A. Zulfikri, G. Arifin, R. M. Ilhamsyah, et al., "Analysis of Phishing Attack Trends, Impacts and Prevention Methods: Literature Study," *Brilliance: Research of Artificial Intelligence*, vol. 4, no. 1, pp. 413–421, 2024.

[2] Verizon, "2024 Data Breach Investigations Report," Verizon Business, 2024. [Online]. Available: https://www.verizon.com/business/resources/reports/dbir/

[3] S. Ahmad, M. Zaman, A. S. Al-Shamayleh, R. Ahmad, S. M. Abdulhamid, I. Ergen, and A. Akhunzada, "Across the spectrum in-depth review AI-based models for phishing detection," *IEEE Open J. Commun. Soc.*, vol. 6, pp. 2065–2089, 2024.

[4] S. K. Birthriya, P. Ahlawat, and A. K. Jain, "An efficient spam and phishing email filtering approach using deep learning and bio-inspired particle swarm optimization," *Int. J. Comput. Digit. Syst.*, vol. 15, no. 1, pp. 1–11, 2024.

[5] S. P. S. Rathore, S. Chamoli, A. Sundaram, N. Varshney, V. Sharma *et al.*, "Deep learning approaches for natural language understanding," in *Proc. 2025 IEEE Int. Conf. Interdiscipl. Approaches Technol. Manag. Soc. Innov. (IATMSI)*, vol. 3, 2025, pp. 1–6.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 5998–6008. arXiv:1706.03762.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186. arXiv:1810.04805.

[8] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," arXiv:1910.01108, 2019.

[9] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "TinyBERT: Distilling BERT for natural language understanding," in *Findings of the Assoc. Comput. Linguistics: EMNLP*, 2020, pp. 4163–4174. arXiv:1909.10351.

[10] M. M. Diop, B. Mamadou, K. Sylla, and S. Ouya, "A multilingual transformer based framework for phishing email detection and deployment in real-world pipelines," in *Proc. IEEE 8th Int. Conf. on Communications, Information System and Computer Engineering (CISCE)*, pp. 1099–1104, 2026.

[11] S. Mahendru and T. Pandit, "Securenet: A comparative study of DeBERTa and large language models for phishing detection," in *Proc. IEEE 7th Int. Conf. on Big Data and Artificial Intelligence (BDAI)*, pp. 160–169, 2024.

[12] Y. Z. Vakili, A. Fallah, and H. Sajedi, "Distilled BERT model in natural language processing," in *Proc. 2024 14th Int. Conf. Comput. Knowl. Eng. (ICCKE)*, 2024, pp. 243–250.

[13] T. Yang, Y. Cheng, Y. Qi, and M. Wei, "Distilling semantic knowledge via multi-level alignment in TinyBERT-based language models," *J. Comput. Technol. Softw.*, vol. 4, no. 5, 2025.
[14] M. A. Uddin, M. Mahiuddin, and I. H. Sarker, "An explainable transformer-based model for phishing email detection: A large language model approach," *Comput. Netw.*, vol. 277, art. 112061, 2026. arXiv:2402.13871.
[15] M. Alauthman, A. Aldweesh, A. Al-Qerem, O. Almomani, and A. Almomani, "Adversarially robust phishing detection via contrastive learning and generative attacks," in *Proc. 2025 10th Int. Conf. Inf. Technol. Trends (ITT)*, 2025, pp. 311–316.
[16] A. Shazad, M. N. Chaudhry, M. K. Abid, and N. Aslam, "Spam email detection using transfer learning of BERT model," *J. Comput. Biomed. Inform.*, 2024.
[17] A. Gaurav, B. B. Gupta, J. Wu, V. Arya, and K. T. Chui, "Lightweight cloud-based phishing email detection using BERT and deep learning," in *Proc. 2024 IEEE Globecom Workshops (GC Wkshps)*, 2024, pp. 1–7.
[18] R. K. Kaliyar, A. Goswami, and P. Narang, "FakeBERT: Fake news detection in social media with a BERT-based deep learning approach," *Multimed. Tools Appl.*, vol. 80, no. 8, pp. 11765–11788, 2021.
[19] Y. A. Kustiawan and K. I. Ghauth, "Feature engineering for phishing website detection using machine learning: A systematic review," *IEEE Access*, vol. 13, pp. 192080–192104, 2025.
[20] J. Hazell, "Large language models can be used to effectively scale spear phishing campaigns," arXiv:2305.06972, 2023.