LVForge: PE Malware Detection from Strong Baselines to Deep Metric Learning A Continuity-Driven and Reproducible Study

Ly Ngoc Vu
Industrial University of Ho Chi Minh City
Ho Chi Minh City, Vietnam
dezzhuge@gmail.com

***Abstract*—Windows PE malware detection is operationally sensitive because even small false-positive increases can create substantial analyst cost. Building on prior baseline-oriented work, this study examines whether objective design on a fixed Transformer backbone improves low-FPR behavior under class imbalance. We implement a unified Flax/JAX LVModel pipeline and compare five variants: baseline, ArcFace, Contrastive, Triplet, and Multi-Similarity. To emulate deployment pressure, training uses a 1:19 benign:malware regime (907:17,235), and evaluation reports aggregate metrics, operating-point metrics (TPR@FPR at 10^{-2}, 10^{-3}, 10^{-4}), and imbalance-focused indicators (specificity, FPR, FNR, balanced accuracy, MCC). Across five seeds, Multi-Similarity shows the strongest overall profile (F1 $= 0.9970 \pm 0.0028$, TPR@FPR$= 10^{-2} = 0.9879 \pm 0.0078$, specificity $= 0.9658 \pm 0.0353$, MCC $= 0.9466 \pm 0.0457$), while ArcFace degrades at strict low-FPR points. The baseline remains competitive and stable. These findings indicate that objective selection can materially change deployment-relevant behavior even when the encoder backbone is unchanged.**

***Index Terms*—malware detection, Windows PE, Transformer, deep metric learning, class imbalance, low-FPR evaluation**

I. Introduction

Windows PE malware detection is an operationally constrained classification problem: high aggregate accuracy is not sufficient when false positives directly increase analyst workload. In practical SOC pipelines, model selection must be based on low-FPR behavior, not only global ranking metrics.

Our prior ATC 2024 study [1] already established strong baseline performance using classical machine learning and text-based deep learning. The remaining gap is objective-level behavior under strict operating points. This paper addresses the following research question:

> *On a shared Transformer backbone, which training objective provides the best trade-off between aggregate quality and low-FPR behavior under imbalance?*

This manuscript is written as a continuity paper for first-time readers. It connects old and new stages in one coherent narrative and explicitly separates inherited components from new contributions.

Contributions.

1) A reproducible unified pipeline for baseline and DML variants, with one backbone and controlled objective changes.
2) Implementation-level documentation of LVModel and its Flax Transformer building blocks.
3) A low-FPR, imbalance-aware evaluation protocol with multi-seed statistical reporting.
4) Evidence-based deployment guidance on baseline versus DML objectives.

The remainder of this paper is organized as follows: Section II maps prior and current scope, Section III details dataset/task settings, Section IV presents methodology, Section V defines the experimental protocol, Section VI reports results, and Sections VII–IX discuss implications, limitations, and conclusions.

II. Project Continuity and Scope

Table I maps prior and current scopes to make the extension explicit.

TABLE I
Scope map between prior and current stages

Component	Prior stage (ATC 2024)	Current stage (LVForge)
Dataset	Multi-source PE collection and preprocessing	Reused and audited; source-level vs local-run accounting
Models	LR, RF, SVC, XG-Boost, LSTM, BiL-STM, DistilBERT	Shared Flax/JAX Transformer baseline (LVModel)
Metric learning	Not included	ArcFace, Contrastive, Triplet, Multi-Similarity
Evaluation	Aggregate classification metrics	Multi-seed + threshold-aware low-FPR + imbalance metrics
Reproducibility	Experimental scripts/paper outputs	Unified `run_all.py`, JSON aggregation, checkpoint artifacts

Accordingly, the novelty of this stage is an objective-controlled extension, rather than a replacement of the prior baseline study.

III. Dataset and Task Setting

A. Task Definition

The task is binary malware detection from text-like PE representations, with label mapping `0`=*benign*, `1`=*malware*.

B. Data Origin and Local Run Statistics

Following [1], samples originate from VX Underground, VirusShare, Softonic, and SourceForge. The local experimental file is `finData.csv` (`Texts`, `label`).

TABLE II
DATASET SUMMARY FROM SOURCE-LEVEL TO CURRENT RUN

Data view	Benign	Malware	Ratio (B:M)
Source-level (prior stage)	17,150	17,235	1:1.00
Current local `finData.csv`	17,135	17,235	1:1.01
Subsampled pool for training/eval	907	17,235	1:19.00

Imbalance is intentionally induced by benign subsampling with benign proportion $r = 0.05$:

$$N_{\text{benign,target}} = \left\lfloor \frac{r}{1-r} N_{\text{malware}} \right\rfloor . \tag{1}$$

With $N_{\text{malware}} = 17{,}235$, we obtain $N_{\text{benign,target}} = 907$ and total $N = 18{,}142$.

For each seed, data is shuffled then split 80/20; evaluation keeps full batches only (`batch_size`=128), yielding 3,584 validation samples per seed.

To make feature semantics explicit, Table III lists representative PE-derived attributes. Table IV provides context against widely cited malware datasets.

Compared with very large corpora, this medium-scale setup is suitable for controlled objective-level ablations and frequent reproducible runs.

IV. Methodology

A. Unified Pipeline

All variants share preprocessing, tokenizer, backbone, optimizer schedule, split logic, and evaluation code. Only the objective/head branch changes. This design isolates the effect of the learning objective while limiting confounding implementation differences. Fig. 1 summarizes the pipeline.

B. LVModel Architecture Details

The shared encoder is implemented with `FlaxMultiHeadSelfAttention` and `FlaxTransformerLayer` modules. For input IDs $\mathbf{X} \in \mathbb{N}^{B\times T}$:

$$\mathbf{H}_0 = E_{\text{tok}}(\mathbf{X}) + E_{\text{pos}}(1{:}T). \tag{2}$$

Each Transformer block uses pre-normalization residual connections:

$$\tilde{\mathbf{H}} = \mathrm{LN}(\mathbf{H}_{l-1}), \tag{3}$$
$$\mathbf{H}' = \mathbf{H}_{l-1} + \mathrm{MHA}(\tilde{\mathbf{H}}), \tag{4}$$
$$\mathbf{H}_l = \mathbf{H}' + \mathrm{FFN}(\mathrm{LN}(\mathbf{H}')). \tag{5}$$

In attention, Q/K/V are generated by a single dense projection:

$$\mathrm{QKV} = W_{qkv}\mathbf{H}, \quad \mathbf{A} = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_h}}\right). \tag{6}$$

Sequence representations are mean-pooled, then passed through dense+tanh, dropout, LayerNorm, and a linear classifier:

$$\mathbf{z} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{H}_{L,t}. \tag{7}$$

Configuration: $d_{\text{model}} = 256$, heads$= 8$, $d_{ff} = 512$, layers$=2$, dropout$= 0.1$, max length≈ 380.

C. Objective Heads and Losses

Baseline. The baseline uses the same LVModel with focal loss [2] during training:

$$\mathcal{L}_{\text{focal}} = -\alpha(1-p_t)^\gamma \log(p_t), \tag{8}$$

with $\alpha = 0.25$, $\gamma = 2.0$.

ArcFace. ArcFace [3] applies additive angular margin in cosine space:

$$\mathcal{L}_{\text{arc}} = -\log \frac{e^{s\cos(\theta_y+m)}}{e^{s\cos(\theta_y+m)} + \sum_{j\neq y} e^{s\cos(\theta_j)}}. \tag{9}$$

Contrastive/Triplet/Multi-Similarity. These variants use normalized embeddings and optimize a hybrid objective:

$$\mathcal{L} = \lambda\mathcal{L}_{\text{CE}} + (1-\lambda)\mathcal{L}_{\text{metric}}, \tag{10}$$

with $\lambda = 0.5$ in this implementation for Contrastive, Triplet (batch-hard), and Multi-Similarity [4]–[6].

V. Experimental Protocol

A. Reproducible Setup

All variants are launched by one command:

```
python scripts/run_all.py
```

The latest full run finished with 5/5 PASS in 558.6 seconds.

Common setup:

- Seeds: $\{42, 43, 44, 45, 46\}$.
- Batch size: 128; epochs: 5; early stopping patience: 2.
- Optimizer: AdamW with cosine decay and gradient clipping.
- Learning rate: 2×10^{-4}.
- Split: shuffled 80/20 validation protocol per seed.

B. Metrics and Thresholding

Thresholds are tuned per seed by maximizing validation F1 on the PR curve. Reported metrics include Accuracy, Precision, Recall, F1, ROC-AUC, PR-AUC, TPR@FPR(10^{-2}, 10^{-3}, 10^{-4}), and imbalance-focused metrics:

$$\text{Specificity} = \frac{TN}{TN+FP}, \tag{11}$$
$$\text{FPR} = \frac{FP}{FP+TN}, \tag{12}$$
$$\text{FNR} = \frac{FN}{FN+TP}, \tag{13}$$
$$\text{MCC} = \frac{TP\cdot TN - FP\cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}. \tag{14}$$

TABLE III
DATASET STRUCTURE INFORMATION

Feature field	Description	Type	Example
Filename	Executable file name	String	setup.exe
MD5	File hash signature	String	d41d8cd98f00b204e9800998ecf8427e
File Size	Binary size in bytes	Number	1048576
Entropy	Byte-level entropy	Float	6.72
PE Header Machine	Machine architecture identifier	Number	332
PE Header Number of Sections	Sections count in PE header	Number	3
Sections	Parsed section-level metadata	Object	section name=.text; size=4096
Import Table	Imported DLL/API symbols	Object	quazip.dll: extractFile, compressFile
Export Table	Exported symbols if available	Object / Null	NaN when absent
Type	Ground-truth label	String	Malware / Benign

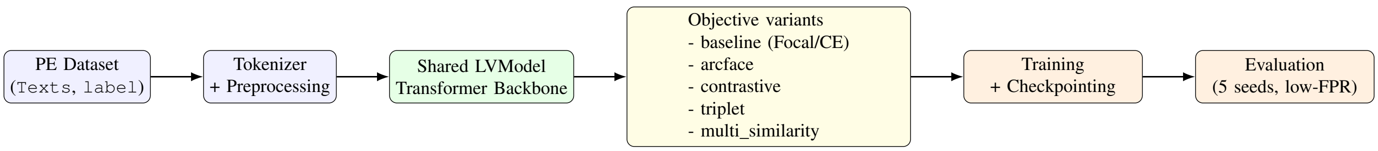

Fig. 1. Unified pipeline for baseline and DML variants.

TABLE IV
COMPARING DATASET CHARACTERISTICS

Dataset	Benign	Malware	Total
Android Malware Dataset	9,476	5,560	15,036
SOREL-20M [7]	8.6M	11.4M	20M
BODMAS [8]	57,293	77,142	134,437
Our dataset (source-level)	17,150	17,235	34,385

For each metric, we report mean and sample standard deviation over five seeds, and compute 95% confidence intervals using a t-interval. This reporting choice is intended to emphasize operational stability rather than single-run point performance.

VI. RESULTS

A. Prior-Stage Reference

Tables V and VI provide prior-stage results from [1]. They are included for context and continuity, not direct protocol-matched comparison.

TABLE V
ML BASELINES FROM PRIOR PAPER [1]

Model	Precision	Recall	F1	Accuracy
Logistic Regression	0.990086	0.990232	0.990111	0.990112
Random Forest	0.990000	0.990382	0.990402	0.990403
SVC	0.990000	0.988060	0.988075	0.988076
XGBoost	0.990000	0.991542	0.991566	0.991566

B. Current LVForge Results (5-Seed Mean $\pm$ Std)

Overall, the results indicate that objective choice affects low-FPR behavior more strongly than aggregate metrics alone suggest. Multi-Similarity maintains favorable operating-point behavior, whereas ArcFace appears sensitive under strict false-positive constraints.

TABLE VI
DEEP LEARNING BASELINES FROM PRIOR PAPER [1]

Model	Precision	Recall	F1	Accuracy
DistilBERT	0.9864895	0.9865220	0.9864705	0.986471
LSTM	0.9674135	0.9674490	0.9674130	0.967413
BiLSTM	0.9507005	0.9499545	0.9500705	0.950102

TABLE VII
OBJECTIVE-LEVEL COMPARISON ON THE CURRENT PIPELINE

Variant	Acc	F1	ROC-AUC	PR-AUC	TPR@1e-2	TPR@1e-3	TPR@1e-4
baseline	0.9923±0.0040	0.9959±0.0021	0.9983±0.0016	0.9999±0.0001	0.9754±0.0157	0.9468±0.0512	0.9468±0.0512
arcface	0.9858±0.0018	0.9925±0.0009	0.9643±0.0053	0.9981±0.0002	0.0000±0.0000	0.0000±0.0000	0.0000±0.0000
contrastive	0.9942±0.0050	0.9969±0.0026	0.9971±0.0040	0.9998±0.0004	0.9830±0.0132	0.7151±0.3431	0.7151±0.3431
triplet	0.9928±0.0060	0.9962±0.0032	0.9979±0.0029	0.9999±0.0002	0.9768±0.0176	0.9146±0.0883	0.9146±0.0883
multi_similarity	**0.9944±0.0053**	**0.9970±0.0028**	**0.9984±0.0022**	**0.9999±0.0001**	**0.9879±0.0078**	**0.9561±0.0316**	**0.9561±0.0316**

TABLE VIII
IMBALANCE-FOCUSED METRICS (5 SEEDS)

Variant	Specificity	FPR	FNR	Balanced Acc	MCC
baseline	0.9387±0.0372	0.0613±0.0372	0.0047±0.0020	0.9670±0.0195	0.9236±0.0348
arcface	0.8802±0.0813	0.1198±0.0813	0.0086±0.0026	0.9358±0.0395	0.8567±0.0310
contrastive	0.9533±0.0434	0.0467±0.0434	**0.0035±0.0027**	0.9749±0.0230	0.9430±0.0449
triplet	0.9439±0.0326	0.0561±0.0326	0.0045±0.0045	0.9697±0.0183	0.9303±0.0504
multi_similarity	**0.9658±0.0353**	**0.0342±0.0353**	0.0040±0.0036	**0.9809±0.0193**	**0.9466±0.0457**

TABLE IX
RUNTIME FROM LATEST FULL RUN (RUN_ALL.PY)

Variant	Time (s)
baseline	93.6
arcface	100.0
contrastive	122.5
triplet	125.8
multi_similarity	116.7

TABLE X
WELCH t-TEST: MULTI-SIMILARITY VS BASELINE (N=5)

Metric	Mean diff. (MS - Base)	p-value
F1	+0.0011	0.5014
TPR@FPR=1e-2	+0.0124	0.1659
TPR@FPR=1e-3	+0.0093	0.7402
Specificity	+0.0271	0.2715
MCC	+0.0230	0.3997

VII. DISCUSSION

RQ1: Does deep metric learning help on imbalanced PE data? Yes, but not uniformly across objectives. Multi-Similarity is strongest overall in this run, while ArcFace underperforms at strict low-FPR points despite acceptable aggregate scores.

RQ2: Is baseline still useful? Yes. The baseline remains strong and stable, and can be treated as a robust deployment fallback.

RQ3: Why not claim strict superiority yet? The sample size is limited (five seeds), and Table X reports non-significant differences between baseline and Multi-Similarity. This aligns with overlapping confidence intervals and indicates that the practical advantage is promising but not definitive.

Operational implication. Objective choice strongly affects low-FPR behavior. Deployment screening should therefore include TPR@FPR, specificity, FNR, and MCC in addition to Accuracy/F1/AUC.

VIII. THREATS TO VALIDITY AND LIMITATIONS

Internal validity. Validation uses shuffled 80/20 splits with fixed seed set, not an external hold-out.

External validity. Results are from one medium-scale PE corpus; transferability to other malware families, time periods, and packing/obfuscation regimes is untested.

Statistical conclusion validity. Although five-seed reporting improves robustness over single-run comparisons, statistical power remains limited for small effect sizes.

IX. CONCLUSION

This continuity paper consolidates inherited baseline foundations and new objective-level extensions in LVForge. Under controlled comparison on a shared backbone, Multi-Similarity provides the strongest operating profile in this run, while baseline remains a stable reference model. Future work should validate these trends on external or temporal hold-out data and include calibration analysis before stronger deployment claims are made.

ACKNOWLEDGMENT

This work extends the author's previous study and reframes the pipeline for reproducible objective-level analysis for new supervision context.

REFERENCES

[1] L. N. Vu, "Windows Malware Detection: Exploring from Machine Learning to Text-Based Deep Learning Approaches," in Proc. ATC, 2024.
[2] T.-Y. Lin *et al.*, "Focal Loss for Dense Object Detection," ICCV, 2017.
[3] J. Deng *et al.*, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," CVPR, 2019.
[4] X. Wang *et al.*, "Multi-Similarity Loss with General Pair Weighting for Deep Metric Learning," CVPR, 2019.
[5] F. Schroff *et al.*, "FaceNet: A Unified Embedding for Face Recognition and Clustering," CVPR, 2015.
[6] R. Hadsell *et al.*, "Dimensionality Reduction by Learning an Invariant Mapping," CVPR, 2006.
[7] A. Harang and E. M. Rudd, "SOREL-20M: A Large Scale Benchmark Dataset for Malicious PE Detection," arXiv:2012.07634, 2020.
[8] BODMAS Dataset, "Benchmark for malware analysis at scale," 2021.