Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Cascaded Deep Stacking of State-Space and Tabular Deep Models for Detecting and Attributing Network Covert Timing Channels

Mohammed Youssef

mo.ausf95@gmail.com

Imam Mohammad ibn Saud Islamic University

Mohammed Al Khatib

Imam Mohammad ibn Saud Islamic University

Fahman Saeed

Imam Mohammad ibn Saud Islamic University

Article

Keywords: covert timing channel, network intrusion detection, deep stacking ensemble, state-space model, transformer, technique attribution

Posted Date: November 25th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11284557/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Abstract

Covert timing channels (CTCs) hide information in the inter-packet delays of otherwise legitimate network traffic, evading payload inspection. Detectors reported on the public Iglesias *feat_analysis* corpus decide only whether a flow is covert, leaving an analyst without the technique behind an alert. This study develops a two-layer cascaded deep-stacking ensemble: the first layer screens covert versus overt traffic, and the second, triggered only on covert-flagged flows, attributes one of eight known timing techniques. Each layer stacks the two deep base learners that validation selected from a pool of tabular deep networks, Transformer encoders and Mamba state-space models, with classical machine learning kept as a baseline. On a held-out test partition the first layer raises covert-flow recall to 95.45% from the published decision-tree baseline's 84.09% (F1 0.945 versus 0.876). Attribution reaches a macro-F1 of 0.9736 over the eight techniques, 3.3 points above the best single deep model, although a classical Extra Trees ensemble scored on the same basis reaches 0.9847. Every model trained on the full thirteen-feature representation, a single decision tree included, detected a previously unseen technique far better than the published figure. A Mamba model was the strongest single binary detector evaluated, and SMOTE rebalancing did not improve detection.

1. Introduction

A covert channel is a communication path that lets two parties exchange information in a way the security policy never intended to allow. Lampson first identified the risk in 1973 [1], and later surveys have catalogued such channels at almost every layer of the network protocol stack [2,3]. Covert channels are usually grouped by the carrier they exploit: storage channels hide data in the content or header fields of packets, whereas timing channels hide it in the timing of packet transmission [2]. Covert timing channels (CTCs) are the harder of the two to defend against: rather than hiding data in the content of packets, they encode it in *when* packets are sent, modulating the inter-packet delays that a payload-inspecting firewall never examines. Because the carrier is timing rather than content, encryption does not close the channel and signature matching has nothing to match, which is what makes CTCs a durable route for data exfiltration and stealthy command-and-control [3]. Figure 1 illustrates the setting: a covert flow whose inter-packet delays encode hidden bits, an overt flow whose timing varies only through natural jitter, and a monitoring point that must tell the two apart.

[1]College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh 11432, Saudi Arabia. *Corresponding authors: Mohammed Youssef, email: mo.ausf95@gmail.com; Mohammed Al Khatib, email: mhalkhatib@imamu.edu.sa

The detection literature, reviewed in the Literature Review, has moved through three phases. It began with statistical tests on inter-arrival times [4,5]. It then turned to supervised classifiers over hand-engineered flow features, the most influential being the Descriptive Analysis of Traffic (DAT) framework and decision-tree baseline that Iglesias and colleagues built on the public *feat_analysis* corpus [6,7]. Most recently it has moved to deep detectors built on convolutional, recurrent, and generative models [8–10].

Two observations about that third phase motivate this study. First, deep learning for CTC detection already exists, but the architectures that have reshaped sequence modelling elsewhere have barely reached it. Transformers [11] are now common in flow-based intrusion detection [12] yet almost unused for network CTCs. The selective state-space models of the Mamba family [13] have begun to appear in general traffic and intrusion classification [14,15], but to our knowledge they have not been applied to covert timing detection at all. Second, and more consequentially, almost every detector in the literature stops at a binary verdict, covert or overt, and never says *which* timing technique produced the alert. An analyst told only that a flow is covert must still reverse-engineer the mechanism before choosing a response, whereas a detector that named the technique would let a channel-specific countermeasure be applied directly.

This study addresses both gaps with a single design: a two-layer cascaded deep-stacking ensemble that detects CTCs and then attributes each covert flow to its specific technique. The first layer performs binary covert-versus-overt screening; the second layer, a closed-set classifier over eight known timing techniques, is triggered only on the flows the first layer has already flagged as covert. Each layer is a stacked generalisation [16] of the two strongest deep base learners drawn from a pool of tabular deep networks [17,18], Transformer encoders [11], and Mamba state-space models [13], combined through a class-balanced meta-learner, with the two layers joined by an explicit confidence gate rather than fused into a single flat decision. The design departs deliberately from the flat, single-level deep stacking network that Tang and colleagues applied to intrusion detection [19]. Decomposing detection from attribution lets each layer face a more homogeneous decision boundary, and gating the expensive attribution stage behind the binary screen keeps the classifier cheap on the overwhelmingly overt traffic that a monitor actually sees. Classical machine learning is trained throughout as a comparison baseline but is kept out of the ensemble pool, so that the study measures what end-to-end deep models add over the published classical result without borrowing the trees' strength inside the cascade. Framed this way, the object of study is a layered decision system rather than a single classifier: a screening stage, a gated attribution stage, and a meta-learner that combines heterogeneous components at each stage, evaluated under the false-alarm budget and per-flow cost that a monitoring deployment imposes.

To our knowledge, this is the first cascaded deep-stacking ensemble for joint covert-timing-channel detection and per-technique attribution on the public Iglesias *feat_analysis* corpus, and the first to draw its base learners from a pool spanning tabular deep, Transformer and state-space architectures under a selection rule driven by validation performance alone. That selection is itself informative. In the unaugmented configuration it placed Mamba state-space variants in three of the four base-learner slots and ranked all three Transformer encoders last for the binary layer, and a single Mamba model was the strongest binary detector of every model trained, classical ones included; to our knowledge this is the first evidence that a selective state-space model is competitive for covert timing channel detection. The advantage is not unconditional: under the fully balanced SMOTE ratio a plain Transformer encoder did enter the binary layer. The claim is deliberately narrow. Deep learning for CTC detection, stacking for intrusion detection, and Mamba for network traffic each have prior art; the contribution is their specific combination on this task and this corpus, not any one of them in isolation.

This study makes three contributions, each evaluated on the held-out test partition.

- • **More accurate detection.** The cascade's binary layer improves on the published decision-tree baseline [6] on the same corpus, and detects timing techniques absent from training far better than the published figures, a gain it shares with single models trained on the same thirteen features.
- • **Technique attribution.** The gated second layer names the timing technique behind each detected flow, a capability the binary-only baselines on this corpus do not provide.
- • **An honest augmentation result.** Rebalancing the covert training data with the synthetic minority over-sampling technique (SMOTE) [20] is evaluated and reported as a controlled negative result rather than omitted.

These outcomes are developed in Results and the Discussion. On the held-out test partition the cascade's first layer raises covert-flow recall to 95.45% from the baseline's 84.09%. Technique attribution reaches a macro-F1 of 0.9736 over the eight known techniques, above every single deep model but below a classical Extra Trees ensemble scored on the same basis, a result reported as found. The large gain on the unseen techniques over the published figures turns out to belong to the thirteen-feature training pipeline rather than to the cascade, since single models trained in it, a decision tree included, show it too. No SMOTE ratio improves on the unaugmented model, a controlled negative result reported in full rather than set aside. Throughout, comparisons against the baseline are matched by technique-set, the assignment of partitions to training, validation and test roles is stated wherever it bears on a result, and every reported figure is weighed against the threats to its validity, in line with recent guidance on sound evaluation in security machine learning [21].

2. Literature Review

This section reviews the main lines of prior work on covert timing channel detection, from statistical tests through classical machine learning to deep learning and ensemble methods, and closes by stating the gap that the present study addresses. It is restricted to the studies that bear directly on the design choices of this paper.

2.1. Statistical Detection

Detection of covert timing channels began with statistical tests on inter-arrival times. Cabuk et al. [4,22] observed that legitimate timing carries natural randomness from routing jitter and queueing, whereas a covert encoder must regularise timing to carry a message, and proposed a regularity score and an entropy test that treat the loss of randomness as the signal. Berk et al. [23] took a related route, comparing the observed inter-arrival distribution against the channel-capacity-optimal one. These tests worked against the first generation of channels but were soon defeated by encoders that randomise their timing. Gianvecchio and Wang answered with corrected conditional entropy, which measures the entropy of a timing sequence given its own history and so captures temporal dependencies that plain entropy misses [5,24]. They also showed that model-based channels can be built to fit the distribution of legitimate traffic,

which raised the bar for every statistical detector that followed [25]. Luo et al. [26] extended the design-and-detection analysis to TCP timing channels. Archibald and Ghosal's comparative evaluation later confirmed that no single statistical metric is sufficient against modern channels [27], which is the point at which the field moved to learned detectors.

2.2. Classical Machine Learning

Classical machine learning replaced the fixed threshold with a trained decision boundary over a hand-engineered feature vector. Shrestha et al. [28] built a support-vector-machine framework for CTC detection, and Rezaei et al. [29] addressed reliable detection over real-time traffic. The most influential body of work in this line is that of Iglesias and colleagues, whose Descriptive Analysis of Traffic (DAT) framework condenses a flow into thirteen statistical features spanning the regularity-versus-randomness axis [6,30]. Evaluating decision trees and several other shallow learners over that feature space, they reported binary accuracies above ninety-nine per cent on the public *feat_analysis* corpus [6], and a companion study induced decision-tree rules for the same task [7]. One earlier paper from the group did go past a binary verdict: a single Bayesian network labelled seven timing techniques and overt traffic from five DAT features [31], but on a different and smaller dataset (16,381 flows), with a random split and no unseen techniques. The group released three public datasets with these studies: ARES-2017 with seven techniques, CD-MAKE-2017 with eight, and *feat_analysis*-2017 with ten and the thirteen features used here (Supplementary Table S1). To our knowledge only the ARES set has been reused by other authors: Mao et al. [32] trained an attention-enhanced LSTM on it as a binary detector under a random 80:20 split, reporting 99.51% accuracy and 96.76% recall. We found no study by other authors on *feat_analysis*. That corpus, and the decision-tree result reported on it, is the baseline against which the present study is measured; both are binary detectors that decide only whether a flow is covert. Tree ensembles remain strong on this small, dense feature space [33]; related detectors have been proposed for timing channels on other traffic [34] and for other covert-channel families [35].

2.3. Deep Learning

Deep learning entered CTC detection with feed-forward and recurrent models. Darwish et al. [36] paired a multi-scale representation of inter-arrival times with a deep neural network. Convolutional and recurrent detectors then came to dominate: SnapCatch rendered timing sequences as images for a CNN [8,37], a hybrid LSTM and one-dimensional CNN was trained directly on inter-arrival sequences [9], and a pure LSTM detector was applied to DNS covert channels [38]. Sun et al. [10] trained an auxiliary-classifier GAN on Gramian-angular-field images to detect and discriminate several timing channels at once, and Li et al. [39] targeted channels whose features are unknown in advance, the same novelty problem that the unseen techniques of the present corpus pose.

Two architectures that have reshaped sequence modelling elsewhere have barely reached CTC detection. Transformers [11] are now common in flow-based intrusion detection [12] and have been applied to DNS-over-HTTPS and IPv6 covert channels [40,41], but not to network covert timing channels on a public feature

corpus. Selective state-space models of the Mamba family [13] have appeared in general traffic classification and intrusion detection [14,15], but to our knowledge not in covert timing detection. For tabular inputs such as the DAT vector, the benchmark of Gorishniy et al. [17] established that well-tuned multilayer perceptrons, residual networks, and the FT-Transformer are competitive with tree ensembles, and TabNet added an attention-based feature-selection mechanism [18]; these are the tabular deep models from which the present base-learner pool is drawn.

2.4. Ensembles and Stacking

Stacked generalisation [16] trains a meta-learner on the out-of-fold predictions of several base learners, letting it discover which learner to trust in which region of the feature space [42]. Tang et al. [19] brought this design to intrusion detection as a deep stacking network, fusing the outputs of three base models in a single layer, and later work widened the base pool and the meta-learner choices [43,44]. Within covert-channel detection specifically, Elsadig and Gafar stacked several shallow learners under a logistic-regression meta-classifier to detect packet-length channels [45,46], and Li and Chasaki applied a hierarchical ensemble to covert-channel attacks on cyber-physical systems [47]. These studies establish that stacking is effective for covert-channel detection. None of them, however, targets timing channels on a public benchmark, and none separates detection from technique attribution.

Table 1
Summary of representative prior work on covert-channel detection, and the limitation each leaves open. CTC denotes covert timing channel.

Era / Work	Approach	Output	Limitation left open
Cabuk et al. [4,22]	Regularity and entropy tests on inter-arrival times	Covert / overt	One signature per test; evaded by shaped timing
Gianvecchio & Wang [5,24]	Corrected conditional entropy	Covert / overt	Model-based channels fit legitimate distributions [25]
Iglesias et al. [6,7,30]	DAT features with decision trees and other shallow learners	Covert / overt	Binary only; no technique attribution; low published recall on unseen techniques
Iglesias & Zseby [31]	Outlier detectors; Bayesian network on five DAT features	Seven techniques + overt, one flat classifier	Different, smaller dataset; random split; no unseen techniques
Shrestha et al. [28]	Support vector machine on flow features	Covert / overt	Binary only; hand-tuned feature set
Al-Eidi et al. [8,9]	CNN on timing images; hybrid LSTM and 1-D CNN	Covert / overt	Binary only; private or non-public feature data
Chen et al. [38]; Li et al. [39]	LSTM; learned variation patterns	Covert / overt	Binary only; not evaluated on this corpus
Mao et al. [32]	Transformer encoder, LSTM and additive attention	Covert / overt	Binary only; ARES-2017 set with a random split
Rahman et al. [48]	Tree ensembles and neural networks, detection then categorisation	Covert / overt, then channel type	Storage channels in IPv6 headers, not timing; single models, no stacking
Tang et al. [19]; Lazzarini et al. [43]	Flat deep stacking for intrusion detection	Attack / benign	Single-level stack; no gated attribution stage
Elsadig & Gafar [45,46]	Stacked shallow learners, logistic-regression meta-learner	Covert / overt	Packet-length storage channels, not timing; no attribution

2.5. Research Gap

Table 1 summarises that literature and the limitation each line of work leaves open. Read together, and to our knowledge, four gaps motivate the present study. First, the published detectors on the *feat_analysis* corpus, and almost all detectors elsewhere, stop at a binary verdict; no prior work on this corpus attributes a detected flow to its specific timing technique, and the one technique-level experiment by the same group used a single flat classifier on a different dataset [31]. Second, no prior work evaluates a Mamba-class state-space model, alongside Transformer encoders, for covert timing channel detection. Third, no prior stacking ensemble for covert channels gates a multi-class attribution stage behind a binary detection stage; the

existing designs are flat, single-level stacks. The closest design we found, by Rahman et al. [48], also detects first and then names the channel type, but for storage channels in IPv6 header fields and with single models rather than stacked ones. Fourth, the closed-set limitation that arises when evaluation data contain techniques never seen in training is rarely treated explicitly. The cascaded ensemble described next addresses the first three gaps directly and reports the fourth rather than averaging it away.

3. Methods

3.1. Dataset

All experiments use the public Iglesias *feat_analysis* corpus [6], released by the Communication Networks group at TU Wien. To our knowledge it is the only public CTC dataset that combines a full statistical feature set with several distinct covert techniques, which is why the field's standard published baseline is reported on it. The corpus comes as six comma-separated files that form three partitions, each pairing overt and covert flows: 30,000 overt and 1,022 covert flows, 30,000 and 1,036, and 46,193 and 1,232. The first two cover the same eight covert techniques (CAB, BER, SHA, JIN, GAS, LUO, ZAN and GIF). The third adds two more, ED1 and ED2, which appear in no other file.

The data are imbalanced in two ways. Overt flows outnumber covert flows by about thirty to one in every partition, and within the covert class JIN is a minority, with 42 training flows against 140 for each of the other seven techniques. A detector that called every flow overt would already pass 96% accuracy, so the evaluation relies on metrics that show how the minority class is handled (the Evaluation metrics subsection).

Each flow is described by the thirteen features of the Descriptive Analysis of Traffic (DAT) framework [6,30], computed from the inter-arrival times (IATs) of its packets rather than from their contents. Five describe the shape of the IAT distribution: the number of unique IAT values, the probability of the most common value, two multimodality scores and the mean spread of the detected peaks. Four test for regularity: runs-test and sign-test coefficients, approximate entropy and Kolmogorov complexity. Two measure dependence over time, the autocorrelation and the Hurst exponent, and two measure volume, the packet count and an estimate of the covert capacity. Several are correlated, most strongly the covert-capacity estimate and the sign test with the packet count (0.82 and 0.77 on the training split; Supplementary Fig. S1). Every model is trained and tested on all thirteen.

3.2. Validation and Test Protocol

The deep models are trained by stochastic gradient descent, so they need a held-out signal during training for early stopping [49], learning-rate scheduling [50] and model selection [51]. The study therefore follows the standard train, validation and test split [52]. The first partition is used for training. The second, with the same eight techniques, is used for validation: it drives early stopping and the choice of base learners. The third, the only one that contains ED1 and ED2, is the test set and was used once, after every design decision had been fixed. No flow was altered or moved between partitions, so ED1 and ED2 are never seen during

training or model selection. Comparisons with the published baseline are made only between matching technique sets, eight techniques against eight and ten against ten, as recommended for security evaluations [21].

3.3. Preprocessing

Preprocessing is kept minimal. The autocorrelation feature is undefined for eight flows that have too few packets, and these values are filled with the training median. The deep models receive z-score normalisation fitted on the training partition and applied unchanged to validation and test, while the tree ensembles, which are insensitive to scale, use the raw features. Binary labels map overt to 0 and covert to 1, and the multi-class labels are the eight known techniques.

For the augmentation study only, the covert training flows are over-sampled with SMOTE [20] to three overt-to-covert ratios: 3:1, 3:2 and the fully balanced 3:3. An explicit sampling schedule keeps the original share of each technique instead of equalising them. Because the features differ widely in scale, the nearest-neighbour interpolation is done in standardised space and the synthetic flows are mapped back to raw units [53]. The validation and test partitions are never augmented, so every evaluation keeps the natural covert share of about three per cent.

3.4. Detection Approaches and Base Learners

Four model families are trained on the same data, splits and metrics. Classical machine learning, covering linear and probabilistic models, tree ensembles, kernel and nearest-neighbour methods and a small multilayer perceptron, is used only as a comparison baseline. The three deep families form the pool from which the cascade draws its base learners: tabular deep networks (a multilayer perceptron and ResNet-tabular [17], a one-dimensional convolutional network and TabNet [18]), Transformer encoders (the FT-Transformer [17] and a plain encoder over the feature tokens [11]), and the Mamba S6 selective state-space model [13] in three sizes.

Mamba expects a sequence, so each flow's thirteen features are treated as a sequence of thirteen tokens, each projected to the model width. The Mamba block is written in PyTorch because the authors' CUDA kernel does not build on the Windows setup used here. It follows the layout of the open-source \texttt{mamba-minimal} implementation and the selective-scan recurrence, zero-order-hold discretisation and initialisation of [13], with three simplifications: a full-rank step-size projection instead of a low-rank one, LayerNorm instead of RMSNorm, and a loop over the thirteen positions instead of a fused kernel, which slows inference but does not change the computation.

All deep models except TabNet use the same training recipe, so that differences between them come from the architecture: class-weighted cross-entropy, batches of 512, up to 100 epochs, a learning rate halved when the validation loss stops improving, and early stopping after 15 epochs without improvement in the validation loss, keeping the best weights (Supplementary Fig. S2 shows the loss curves of the three Mamba base learners; TabNet's per-epoch losses were not recorded). The tabular networks use Adam and the Transformer and Mamba models use AdamW, both with weight decay 10^{-4}. TabNet is trained

through its own library, with step learning-rate decay, early stopping on validation accuracy with the same patience, and no class weighting. Hyperparameters are fixed in advance from the architecture references, with learning rates of 10^{-3} for the tabular and Mamba models, 10^{-4} for the Transformers and 2\times10^{-2} for TabNet. No hyperparameter search is run; a per-model search could narrow the gaps between families and is left to future work.

3.5. Cascaded Deep Stacking Network

The cascade is the core of the study. It joins two stacked ensembles [16] so that the second runs only on flows the first has judged covert (Fig. 2). Layer 1 is a binary ensemble of the two deep models with the highest validation F1, combined by a class-balanced logistic-regression meta-learner that outputs a covert probability p1. Flows with p1 below the threshold are labelled overt. The rest go to Layer 2, an ensemble over the eight known techniques built the same way from the two deep models with the highest validation macro-F1.

Both meta-learners are trained by blending, the held-out form of stacking [16,42]. The base learners are fitted once on the training partition, and each meta-learner is then fitted on their predicted probabilities for the validation partition. Since the base learners never saw that partition, the meta-learner learns from out-of-sample predictions, which avoids the leakage that stacking is designed to prevent. Blending was chosen over k-fold out-of-fold stacking because the latter would mean refitting every deep base learner k times, and the validation partition is large enough to fit a logistic regression reliably. One consequence is that the validation partition is used twice, to choose the base learners and to fit the meta-learners, so validation scores are optimistic. They are never compared with test scores, and the test partition plays no part in either step.

Unlike the flat, single-level deep stacking network of Tang et al. [19], stacking here happens inside each layer, and the layers are joined by a gate. Each layer therefore learns a simpler decision than one flat classifier over covert and technique labels together, the costlier attribution stage runs only on the small share of traffic flagged covert, and the second layer can later be replaced, for example by an open-set version, without retraining the first. Classical models are kept out of the pool so that the cascade measures what stacked deep learners add over the published classical result.

Layer 2 is closed-set, so it cannot output ED1 or ED2. When Layer 1 correctly flags an unseen flow as covert, Layer 2 has to assign it one of the eight known techniques: the detection is right but the technique is wrong. Results are therefore reported in three tiers. Layer 1 alone is scored on the binary label, which is what the baseline reports. Layer 2 is scored only on flows that Layer 1 forwarded correctly, which separates attribution from detection errors. The end-to-end cascade credits a covert flow only when both layers are right. ED1 and ED2 are counted separately as detected but misattributed, rather than folded into the eight-technique macro-F1.

3.6. Evaluation Metrics

Because of the imbalance, no single number is relied on. Binary detection is reported with accuracy, precision, recall, F1, false-positive rate, ROC-AUC, PR-AUC and the Matthews correlation coefficient (MCC), with covert as the positive class. PR-AUC and MCC carry the most weight because both stay informative when the negative class dominates [54,55]. Supplementary Note 1 gives the definition of every metric.

The published baseline reports accuracy, precision, recall and ROC-AUC but not MCC [6]. MCC is added for two reasons. At a covert share near three per cent, accuracy is dominated by the overt class and ROC-AUC looks optimistic, because its false-positive rate is computed over the large overt set [54]; MCC uses all four cells of the confusion matrix, so a detector cannot score well on it just by getting the majority class right [55]. MCC also stays defined when a class never appears among the predictions, which matters for a closed-set second layer that cannot output the unseen techniques. The baseline's own metrics are still reported in full, so every comparison in Results is like for like.

Technique attribution is reported with per-class precision, recall and F1, and with macro-F1, which weights every technique equally and so keeps the minority JIN class visible. For the attribution headline, MCC is preferred to macro-F1, because the closed-set limit on ED1 and ED2 caps a ten-class macro-F1 without reflecting any real attribution error. Detection time per flow is recorded as a deployment cost. All figures come from a single run at seed 42, a limitation discussed in the Discussion.

4. Experimental Results and Discussion

All results below are measured on the held-out test partition (1,232 covert and 46,193 overt flows across ten techniques, including the unseen ED1 and ED2) and are organised around the three contributions: binary detection, technique attribution and the effect of SMOTE. Following the validation-driven selection protocol of the cascade description, Layer 1 of the cascade retained Mamba-Slim and Mamba-Deep as its two base learners (validation F1 0.9786 and 0.9746), and Layer 2 retained Mamba-S6 and TabNet (validation Macro-F1 0.9165 and 0.9138). At the fully balanced 3:3 SMOTE ratio the same protocol re-ranks the pool to a Vanilla Transformer paired with Mamba-S6 at Layer 1, and Mamba-Deep paired with Mamba-S6 at Layer 2.

4.1. Binary Detection across Approaches

Table 2 compares the strongest model of each family against the Iglesias 2017 decision-tree baseline [6] and against the cascade's Layer 1, all on the identical ten-technique test partition. Two patterns stand out. The binary task is close to saturated: the classical, deep-learning, and Mamba families all clear an F1 of 0.93, and the best single detector is Mamba-Slim (F1 0.9530, MCC 0.9517), which is what earns the Mamba family both Layer 1 base-learner slots. The exception is the transformer family, whose recall stays competitive (0.9578) but whose precision collapses to 0.7753 under the extreme class imbalance, dragging its F1 to 0.8569; the attention mechanism over-flags when covert flows are outnumbered thirty to one. Every deep model, and the cascade's Layer 1, clears the Iglesias baseline on F1, recall, and MCC. Supplementary Fig. S3 plots the same values.

Table 2

Binary covert-versus-overt detection on the test partition (ten techniques, covert as the positive class). Each row is the strongest model of its family by F1; the cascade row is Layer 1 in isolation. The Iglesias row reproduces Set A, Table 3 of [6] on the matched partition, with F1 and MCC derived from the published confusion matrix. Best value per column in **bold**.

Approach	Top model	Acc	Prec	Recall	F1	MCC	FPR	ROC-AUC
Iglesias 2017 (baseline)	Decision Tree	0.9938	0.9152	0.8409	0.876	0.874	0.0021	0.903
Classical ML (baseline)	Gradient Boosting	0.9973	**0.9662**	0.9269	0.9461	0.9450	**0.0009**	**0.9995**
Deep Learning	CNN-1D	0.9967	0.9100	**0.9683**	0.9383	0.9370	0.0026	0.9993
Transformers	FT-Transformer	0.9917	0.7753	0.9578	0.8569	0.8578	0.0074	0.9979
Mamba S6	Mamba-Slim	**0.9975**	0.9442	0.9619	**0.9530**	**0.9517**	0.0015	**0.9995**
Cascaded DSN	Layer 1	0.9971	0.9356	0.9545	0.9450	0.9435	0.0018	**0.9995**

4.2. Binary Detection versus the Published Baseline

The first contribution concerns binary detection: whether the cascade's first layer improves on the published decision-tree baseline when both are scored on the same held-out partition.

The central comparison matches the cascade's Layer 1 against the Iglesias decision tree on the identical 47,425-flow partition (Set A, Table 3 of [6]). Layer 1 improves on the baseline across every metric the baseline reports: accuracy 99.71% against 99.38%, precision 93.56% against 91.52%, recall 95.45% against 84.09%, F1 0.945 against 0.876, MCC 0.944 against 0.874, and ROC-AUC 0.9995 against 0.903 (Supplementary Fig. S4). The dominant signal is recall: the 11.36-point gain corresponds to 140 additional covert flows caught (1,176 of 1,232 against the baseline's 1,036), and it is achieved while *reducing* the false-positive count from 96 to 81. Supplementary Table S2 sets the two confusion matrices side by side in the format of the original study, and Supplementary Fig. S5 shows the cascade's matrices for both layers. The parallel gains in MCC and ROC-AUC confirm that the improvement is not an artefact of the imbalance-inflated accuracy metric. This establishes the first contribution.

4.3. Generalisation to Unseen Techniques

For deployment, the most telling part of the binary result is how the detector behaves on the two techniques it never saw during training.

The sharpest evidence for that improvement concerns the two techniques that appear in no training file. Figure 4 compares Layer 1's binary recall on ED1 and ED2 with the Iglesias per-technique recall from Table 6 of [6]. Layer 1 detects ED2 at 96.43% recall (54 of 56 flows) against the baseline's 1.79% (1 of 56),

and raises ED1 recall from 29.29% to 62.86%. The gain should not be credited to the cascade. Re-scoring every saved single binary model on the same unseen flows shows that it is shared across the pipeline: the study's own Gradient Boosting model reaches 71.43% on ED1 and 100% on ED2, and even a single decision tree trained here reaches 60.71% and 96.43%, matching Layer 1 on ED2. What differs from the published figure is therefore the input rather than the architecture. The published per-technique recalls come from the original study's relevant-feature configuration, whereas every model here sees all thirteen DAT features, and the evidence points to that fuller representation as the reason unseen techniques become detectable. Layer 1 sits in the middle of the pack on ED1 and ED2; its claim rests on the aggregate binary result, not on the unseen techniques.

That generalisation does not reach the attribution stage, and the study reports the cost rather than hiding it. Because Layer 2 is closed-set, the 142 correctly detected unseen flows must be routed to one of the eight known techniques: ED1 flows are absorbed predominantly into CAB and ED2 flows into GIF, which lowers those classes' precision (CAB to 0.60, GIF to 0.72). Computed over all ten technique labels, the end-to-end macro-F1 is 0.739, below the naive eight-tenths ceiling one might expect, because the misrouted flows are genuine false positives against the classes that absorb them. This is exactly why MCC and the conditional eight-known macro-F1 are foregrounded for the attribution claim, and why the ten-class figure is reported only to make the closed-set cost explicit.

4.4. Technique Attribution

The second contribution is technique attribution: whether the cascade's second layer can name the technique behind a detected covert flow, and how well it does so against the single multi-class models trained here.

The published baseline on this corpus performs binary covert-versus-overt detection only: its classification experiments are explicitly binary, and its per-technique results (Table 6 of [6]) tabulate the binary detector's false negatives for each technique rather than the output of a multi-class classifier. The companion decision-tree study by the same group [7] is likewise a binary detector, on a separate testbed. An earlier study by the same group did label seven techniques and overt traffic with one flat Bayesian-network classifier, with per-technique test recalls between 60.71% and 100% [31]; it used a smaller dataset without unseen techniques, so its figures are not comparable with those reported here. Layer 2's technique attribution therefore has no published baseline on this dataset, and the reference points are instead the single multi-class models trained here. Comparing the cascade with a single classifier requires care, because the two are not evaluated on the same footing by default. A single multi-class model labels every flow it is given, whereas Layer 2 only ever sees the covert flows Layer 1 forwards to it. Scoring Layer 2 on that forwarded subset alone, against a single model scored on a different partition or label set, would credit the cascade for Layer 1's screening and for an easier population at once. Every model is therefore placed on one basis: the held-out test partition, every covert flow belonging to the eight known techniques, nothing conditioned away, with a flow that receives no technique counted as an attribution error. The classical models are scored from the exact artefacts saved during training, each first checked to reproduce its recorded test score. On that basis the cascade reaches a macro-F1 of 0.9736, 3.3 points

above TabNet, the strongest single deep model, at 0.9407, and above every other deep model trained, while attributing the minority JIN technique perfectly against TabNet's 94.64% (Table 3). The Extra Trees ensemble, however, is stronger than both, at 0.9847, and it too attributes JIN perfectly. Classical machine learning was kept out of the base-learner pool by design, so the cascade is the best deep approach to attribution on this corpus but not the best approach overall. That a bagged tree ensemble leads is consistent with the known strength of tree ensembles on small tabular feature spaces [17,33], and it motivates admitting tree ensembles to the Layer 2 pool, which the modular design allows without touching Layer 1. Supplementary Fig. S6 plots the three models on the common basis. The end-to-end cascade classifies 99.35% of test flows correctly (47,118 of 47,425), and because only 2.65% of flows are forwarded to Layer 2, the expensive attribution stage runs on a small fraction of the stream. The second contribution follows, stated at its true size: the cascade adds a technique label the binary baseline cannot produce, and does so better than any single deep model, though not better than a classical tree ensemble.

Table 3

Technique attribution over the eight known techniques, on one common basis: the held-out test partition, all 1,036 covert flows of those techniques, unconditional. A flow that receives no technique is counted as an error, so no model is credited for another's screening. Extra Trees is the strongest single model of any family and TabNet the strongest single deep model; classical models are scored from their saved artefacts. There is no Iglesias entry because the published baselines on this corpus [6,7] perform binary detection only.

Metric (8 known techniques, TEST)	Extra Trees (classical)	TabNet (best deep)	Cascaded DSN
Accuracy	**0.9826**	0.9286	0.9710
Macro-F1	**0.9847**	0.9407	0.9736
Weighted-F1	**0.9835**	0.9396	0.9722
MCC	**0.9800**	0.9191	0.9669
Balanced accuracy	**0.9839**	0.9299	0.9732
Recall, Extra Trees / TabNet / cascade: CAB 0.957/0.821/0.943; BER 0.993/0.993/0.993; SHA 0.929/0.929/0.950;			
JIN 1.000/0.946/1.000; GAS 1.000/1.000/1.000; LUO 0.993/0.879/0.993; ZAN 1.000/0.914/0.914; GIF 1.000/0.957/0.993			

4.5. SMOTE Rebalancing

The third contribution asks whether rebalancing the covert training data with SMOTE improves detection. To answer it, the cascade was rebuilt at each augmentation ratio and compared with the unaugmented result.

The augmentation study rebuilds the cascade from scratch at each SMOTE ratio, leaving the validation and test partitions untouched, so every row of Table 4 is measured on the same held-out test set as Phase 1. The headline finding is negative and holds across the metric battery: no ratio improves on the unaugmented cascade. Phase 1 keeps the best Layer 1 F1 (0.9450) and MCC (0.9435), the best Layer 2

conditional macro-F1 (0.9745), and the best end-to-end accuracy (99.35%). The fully balanced 3:3 ratio is the closest competitor and raises Layer 1 recall to its highest value anywhere (0.9602), but it does so by trading precision away, so F1 and MCC still fall short of Phase 1. The effect on the unseen techniques is the defining pattern: augmentation never moves both ED recalls in the same direction. At 3:3, ED1 recall climbs to 85.0% while ED2 collapses from 96.4% to 51.8%, because interpolating synthetic samples among the known techniques stretches the learned covert boundary toward the region ED1 occupies and away from ED2's. The augmentation result is therefore negative: at the imbalance level of this corpus the minority class, though proportionally small, is absolutely large enough that synthetic oversampling is not the limiting factor. The unaugmented Phase 1 cascade is reported as the study's headline configuration. Supplementary Fig. S7 plots the ratios side by side.

Table 4
Cascaded DSN on the held-out test partition at each SMOTE ratio, against the unaugmented Phase 1 baseline. Layer 1 metrics are binary (covert positive); L2 mF1 is the eight-technique macro-F1 conditional on Layer 1 forwarding a flow, so the rows are comparable with one another but not with the unconditional figures of Table 3; ED1 and ED2 are Layer 1 binary recall (%) on the unseen techniques. Best value per column in **bold**.

Configuration	L1 Rec	L1 F1	L1 MCC	L2 mF1	End-to-end	ED1 rec	ED2 rec
Phase 1 (no aug.)	0.9545	**0.9450**	**0.9435**	**0.9745**	**0.9935**	62.9	**96.4**
SMOTE 3:1	0.9253	0.9321	0.9304	0.9525	0.9934	45.0	75.0
SMOTE 3:2	0.9286	0.9282	0.9263	0.9715	0.9933	50.7	75.0
SMOTE 3:3	**0.9602**	0.9426	0.9412	0.9420	0.9926	**85.0**	51.8

4.6. Discussion

The discussion below reads these results against the prior literature, starting with what the base-learner selection reveals and ending with the threats that limit the claims.

The validation-driven protocol chose the cascade's base learners without human intervention, and its choices are informative in themselves. In the unaugmented configuration both Layer 1 slots and one Layer 2 slot were filled by Mamba variants, with TabNet joining Mamba-S6 at Layer 2. That the state-space family should dominate is consistent with the design intuition behind treating the thirteen-feature vector as a short token sequence: Mamba's selective gating weights each position by its content [13], which suits a feature vector whose components differ sharply in how much they separate covert from overt timing. Mamba-class models have previously been applied to general traffic classification and intrusion detection [14,15]; to our knowledge this is the first evidence that one is competitive for covert timing channel detection, and here a single Mamba-Slim model was the strongest binary detector of every model trained, ahead of the best classical model (F1 0.9530 against 0.9461). The Transformer encoders, trained on the same recipe, ranked last of the ten deep candidates for the binary layer in this configuration. The ranking is configuration-dependent, though: under the fully balanced 3:3 SMOTE ratio a plain Transformer encoder topped the binary ranking and entered Layer 1. TabNet's retention alongside the state-space models is equally telling, since its attention-based feature selection captures a view of the feature space different

enough from Mamba's that the meta-learner benefits from having both, which is the diversity that motivates stacking.

The cascade contributes on two fronts and keeps them separate. Layer 1 alone is already a stronger binary detector than the published baseline, and the stability of its operating point, a false-positive rate of 0.16% on the during-training split and 0.18% on the held-out split, indicates the improvement is not a calibration artefact of the particular threshold chosen. Layer 2 then supplies something the Iglesias line [6,7] never attempted: attribution of a detected flow to its specific technique. Measured on the common basis, that attribution beats every single deep model but not the classical Extra Trees ensemble, so the cascade's case rests on doing detection and attribution in one gated system rather than on winning attribution outright. The practical value is concrete. An analyst who knows which timing technique is in use can reach for the matching countermeasure, such as channel-specific filtering, traffic shaping, or a targeted logging policy, where a binary alarm leaves the mechanism to be reverse-engineered first. The gated structure carries an operational argument as well: because Layer 2 runs on only the 2.65% of test flows that Layer 1 forwards, the expensive attribution stage touches a small fraction of the stream, which is the mode in which inline monitoring becomes tractable. The modularity is a contribution in its own right, since Layer 2 can be replaced, for instance by an open-set variant, without retraining Layer 1, a property no monolithic classifier offers.

A tree ensemble that attributes better raises a fair question: why build the detector from deep models at all? The aim was never to show that classical learners are weak. The original study had already tested shallow learners on this corpus; the open question was whether modern sequence architectures work for covert timing detection. The answer is mixed. On binary detection a Mamba model beat every model trained, classical ones included. On attribution the tree ensembles lead, as the tabular-learning literature predicts [17,33], and stacking closes about three quarters of the gap between the best single deep model and Extra Trees. Admitting tree ensembles to the Layer 2 pool is the natural next step.

The strongest single result and the sharpest limitation arise from the same source. At the binary level the unseen techniques are detected well, ED2 at 96.43% by Layer 1, although single models in the same pipeline match this; at the attribution level the cascade cannot handle them, because a closed-set Layer 2 has no label to give them. The 142 unseen flows that Layer 1 correctly forwards are therefore misrouted into CAB and GIF, contaminating those classes' precision and pulling the ten-class macro-F1 down to 0.739. The natural remedy is an open-set Layer 2 that can assign a low-confidence flow to a reject or “unknown” class rather than forcing it into a known technique. OpenMax-style recalibration of the penultimate layer [56] and the strong-closed-set-classifier approach of Vaze et al. [57] are the two most relevant starting points, and both fit the modular cascade because they touch Layer 2 alone; the broader relevance of unseen-attack handling to network anomaly detection is surveyed by Martins et al. [58]. Turning the binary-correct, technique-incorrect outcome on ED1 and ED2 into a “covert, technique unknown” signal is the principal direction for future work.

The negative augmentation result deserves interpretation rather than dismissal. The most likely explanation lies in the absolute rather than the relative size of the covert class. The training set is severely

imbalanced in proportion, but it still supplies 1,022 real covert flows spread across the eight techniques, which is enough for the deep base learners to characterise each technique's timing signature. SMOTE interpolates new covert points between existing ones of the same technique [20]. The synthetic samples therefore fall inside the region the real ones already occupy; where a classifier is genuinely starved of minority examples this adds useful density, but here it mostly adds weight near the existing boundary without supplying new information. The visible consequence is the precision-for-recall trade of the augmentation results, and the ED1/ED2 behaviour sharpens the same point: augmentation stretches the covert boundary in particular directions rather than uniformly, so the synthetic mass that helps ED1 hurts ED2. This does not contradict the substantial evidence that SMOTE benefits many intrusion-detection pipelines [53,59]; it marks the Iglesias corpus as one where the minority class, though proportionally small, is absolutely large enough that synthetic augmentation is not the limiting factor. Reporting Phase 1 as the headline configuration and the augmentation study as a controlled negative result is the honest reading of that result.

4.6.1. Threats to Validity

Four threats limit these claims. First, every figure comes from a single training run at seed 42. A genuine ranking therefore cannot be fully separated from stochastic-optimisation noise; this matters most for the base-learner selection, where adjacent candidates are close, and a multi-seed audit reporting mean and standard deviation is the appropriate confirmation. Second, the published per-technique recalls on ED1 and ED2 come from the original study's relevant-feature configuration, not from its full feature set, so the comparison on the unseen techniques crosses feature configurations; the single-model results reported with it show that the gap belongs to the representation rather than to the cascade. Third, all results come from a single corpus, so external validity beyond the Iglesias *feat_analysis* setting is untested, and a second dataset such as synthetic traffic from the CCgen generator [60] would probe it. For the same reason the head-to-head comparison is against the only results published on this corpus [6]; the more recent deep detectors reviewed in the Literature Review were evaluated on private or differently featured traffic [9,36,39] and would need reimplementation on this feature set before a like-for-like comparison is possible. Fourth, the closed-set restriction on Layer 2 is structural rather than a tuning choice, and it bounds the attribution claim to the eight known techniques. None of these overturns the reported findings, but each marks the boundary within which they hold.

5. Conclusions

This study developed a two-layer cascaded deep-stacking ensemble for detecting network covert timing channels and attributing each covert flow to its specific technique. The first layer screens covert from overt traffic and the second, gated behind it, attributes one of eight known timing techniques, each layer stacking the two deep base learners that validation selected from a pool of tabular deep, Transformer and state-space architectures, through a class-balanced meta-learner, with classical machine learning kept out of the pool as a baseline. On the held-out test partition the three contributions stand as follows. It beats the published Iglesias decision-tree baseline [6] on binary detection, raising covert-flow recall by 11.36

points to 95.45% and catching 140 more covert flows while reducing false positives, and the models trained in this pipeline, the cascade among them, detect the two unseen techniques far better than the published figures, ED2 at 96.43% against 1.79%, a gain the evidence ties to the full thirteen-feature representation rather than to the cascade. It adds a technique-attribution capability the binary baseline lacks, with a macro-F1 of 0.9736 over the eight known techniques, above every single deep model; a classical Extra Trees ensemble, kept out of the pool by design, attributes better still at 0.9847, which points to admitting tree ensembles to Layer 2 as the next step. And it shows, as a controlled negative result, that SMOTE rebalancing does not improve detection on this corpus, because the covert class is already absolutely large enough for the deep learners to fit.

The limitations are stated plainly: a closed-set second layer that cannot output unseen techniques, a single-seed evaluation, and a single-corpus scope. Each points to concrete future work. The most immediate is an open-set Layer 2 with a reject option [56,57], which would turn the binary-correct, technique-incorrect outcome on ED1 and ED2 into an actionable “covert, technique unknown” signal. A second is to admit tree ensembles to the Layer 2 pool, since the strongest single attributor found here is a classical Extra Trees model. Beyond that, a multi-seed stability audit would quantify the confidence of the base-learner selection, cross-dataset evaluation on a generator such as CCgen [60] would test external validity, and adapting the offline cascade to streaming traffic would close the gap to the operational monitoring setting named at the outset. Three commitments run through the work: disclosing every methodological choice where it is made, matching every comparison by distribution, and reporting every limitation rather than hiding it in an aggregate. Those commitments are what give the reported gains their weight.

Declarations

Competing interests

The authors declare no conflicts of interest.

Ethics approval

Not applicable. This study used a publicly available network-traffic dataset and involved no human participants or animals.

Additional information

Correspondence and requests for materials should be addressed to Mohammed Youssef (mo.ausf95@gmail.com) or Mohammed Al Khatib (mhalkhatib@imamu.edu.sa).

Funding

This research was supported by the College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia.

Author Contribution

Conceptualization, M.Y., M.A.K. and F.S.; methodology, M.Y.; software, M.Y.; validation, M.Y., M.A.K. and F.S.; formal analysis, M.Y.; investigation, M.Y.; data curation, M.Y.; writing (original draft preparation), M.Y.; writing (review and editing), M.A.K. and F.S.; visualization, M.Y.; supervision, M.A.K. and F.S. All authors have read and agreed to the published version of the manuscript.

Acknowledgement

The authors thank the College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University, for its support. During the preparation of this manuscript, the authors used a large language model (Anthropic Claude, Claude Opus 4 series) to assist with drafting and language editing of the manuscript text, and Google Gemini to assist with drafting the schematic illustration in Figure 3. The authors reviewed and edited all output and take full responsibility for the content of this publication. Figs. 1, 2 and 4 and Supplementary Figs. S1-S7 were drawn by the authors with Matplotlib 3.11 (https://matplotlib.org).

Data Availability

This study used the publicly available Iglesias feat_analysis covert-timing-channel dataset released by the Communication Networks group at TU Wien and analysed by Iglesias et al. (2017), available at https://www.cn.tuwien.ac.at/data-analysis/ctc/ (accessed on 12 May 2026). No new data were created in this study. The code that produces every result reported here, together with the per-run metric files the tables and figures are drawn from, is available at https://github.com/moausf95-cpu/ctc-cascaded-deep-stacking; the repository is private during review and will be made public on acceptance, and access can be granted to reviewers on request through the editorial office. Supplementary Table S1 lists the Iglesias datasets and the studies that used them. Supplementary Fig. S1 gives the feature correlations, Fig. S2 the training curves of the Mamba base learners and Supplementary Note 1 the metric definitions. Supplementary Figs. S3, S6 and S7 plot the values of Tables 2, 3 and 4; Fig. S4 gives the ROC and precision-recall curves of Layer 1, Supplementary Table S2 sets Layer 1's confusion matrix beside the published one, and Fig. S5 gives the confusion matrices of both layers. All are provided in the Supplementary Information file.

References

1. Lampson, B. W. A Note on the Confinement Problem. *Commun. ACM*. **16**, 613–615 (1973).
2. Zander, S., Armitage, G. & Branch, P. A. Survey of Covert Channels and Countermeasures in Computer Network Protocols. *IEEE Commun. Surv. Tutor.* **9**, 44–57 (2007).
3. Biswas, A. K., Ghosal, D. & Nagaraja, S. A Survey of Timing Channels and Countermeasures. *ACM Comput. Surv.* **50**, 1–39 (2017).

4. Cabuk, S., Brodley, C. E. & Shields, C. IP Covert Timing Channels: Design and Detection. In Proc. 11th ACM Conference on Computer and Communications Security (CCS) 178–187 (2004). (2004).
5. Gianvecchio, S. & Wang, H. An Entropy-Based Approach to Detecting Covert Timing Channels. *IEEE Trans. Dependable Secur. Comput.* **8**, 785–797 (2011).
6. Iglesias, F., Annessi, R. & Zseby, T. Analytic Study of Features for the Detection of Covert Timing Channels in Network Traffic. *J. Cyber Secur. Mobil.* **6**, 245–270 (2017).
7. Iglesias, F., Bernhardt, V., Annessi, R. & Zseby, T. Decision Tree Rule Induction for Detecting Covert Timing Channels in TCP/IP Traffic. In Machine Learning and Knowledge Extraction (CD-MAKE) 105–122 (2017). (2017).
8. Al-Eidi, S., Darwish, O., Chen, Y., Husari, G. & SnapCatch Automatic Detection of Covert Timing Channels Using Image Processing and Machine Learning. *IEEE Access.* **9**, 177–191 (2021).
9. Al-Eidi, S., Darwish, O., Chen, Y., Elkhodr, M. & Husari, G. A Deep Learning Approach for Detecting Covert Timing Channel Attacks Using Sequential Data. *Clust Comput.* **27**, 1655–1665 (2024).
10. Sun, C., Chen, Y., Tian, H. & Wu, S. Covert Timing Channels Detection Based on Auxiliary Classifier Generative Adversarial Network. *IEEE Open. J. Comput. Soc.* **2**, 407–418 (2021).
11. Vaswani, A. et al. Attention Is All You Need. *Adv. Neural Inform. Process. Syst. (NeurIPS)*. **2017**, 5998–6008 (2017).
12. Manocchio, L. D. et al. FlowTransformer: A Transformer Framework for Flow-Based Network Intrusion Detection Systems. *Expert Syst. Appl.* **241**, 122564 (2024).
13. Gu, A., Dao, T. & Mamba Linear-Time Sequence Modeling with Selective State Spaces. Preprint at (2023). https://arxiv.org/abs/2312.00752
14. Wang, T., Xie, X. & Wang, W. & others NetMamba: Efficient Network Traffic Classification via Pre-training Unidirectional Mamba. Preprint at (2024). https://arxiv.org/abs/2405.11449
15. He, J., Wang, G., Liu, N. & Kong, L. CGP-Mamba: An Intrusion Detection Method Based on State Space Model. In Cyberspace Simulation and EvaluationSmartCom 258–272 (2025). (2024).
16. Wolpert, D. H. Stacked Generalization. *Neural Netw.* **5**, 241–259 (1992).
17. Gorishniy, Y., Rubachev, I., Khrulkov, V. & Babenko, A. Revisiting Deep Learning Models for Tabular Data. In Advances in Neural Information Processing Systems (NeurIPS) 18932–18943 (2021). (2021).
18. Arik, S. O., Pfister, T. & TabNet Attentive Interpretable Tabular Learning. In Proc. AAAI Conference on Artificial Intelligence 6679–6687 (2021).
19. Tang, Y., Gu, L. & Wang, L. Deep Stacking Network for Intrusion Detection. *Sensors* **22**, 25 (2022).
20. Chawla, N. V., Bowyer, K. W., Hall, L. O. & Kegelmeyer, W. P. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Intell. Res.* **16**, 321–357 (2002).
21. Arp, D. et al. Dos and Don'ts of Machine Learning in Computer Security. In Proc. USENIX Security Symposium. 3971–3988 (2022). (2022).
22. Cabuk, S., Brodley, C. E. & Shields, C. IP Covert Channel Detection. *ACM Trans. Inform. Syst. Secur.* **12**, 1–29 (2009).

23. Berk, V., Giani, A. & Cybenko, G. Detection of Covert Channel Encoding in Network Packet Delays. (2005). Dartmouth College, TR2005-536.
24. Gianvecchio, S. & Wang, H. Detecting Covert Timing Channels: An Entropy-Based Approach. In Proc. 14th ACM Conference on Computer and Communications Security (CCS) 307–316 (2007). (2007).
25. Gianvecchio, S., Wang, H., Wijesekera, D. & Jajodia, S. Model-Based Covert Timing Channels: Automated Modeling and Evasion. In Recent Advances in Intrusion Detection (RAID) 211–230 (2008). (2008).
26. Luo, X., Chan, E. W. W. & Chang, R. K. C. TCP Covert Timing Channels: Design and Detection. In Proc. IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) 420–429 (2008). (2008).
27. Archibald, R. & Ghosal, D. A. Comparative Analysis of Detection Metrics for Covert Timing Channels. *Comput. Secur.* **45**, 284–292 (2014).
28. Shrestha, P. L., Hempel, M., Rezaei, F. & Sharif, H. A. Support Vector Machine-Based Framework for Detection of Covert Timing Channels. *IEEE Trans. Dependable Secur. Comput.* **13**, 274–283 (2016).
29. Rezaei, F., Hempel, M. & Sharif, H. Towards a Reliable Detection of Covert Timing Channels over Real-Time Network Traffic. *IEEE Trans. Dependable Secur. Comput.* **14**, 249–264 (2017).
30. Iglesias, F., Annessi, R., Zseby, T. D. A. T. & Detectors Uncovering TCP/IP Covert Channels by Descriptive Analytics. *Secur. Commun. Netw.* **9**, 3011–3029 (2016).
31. Iglesias, F. & Zseby, T. Are Network Covert Timing Channels Statistical Anomalies? In Proc. International Conference on Availability, Reliability and Security (ARES) 2017 1–9 (2017).
32. Mao, J., Lan, X., Chen, Q. & Attention-Enhanced, L. S. T. M. for Covert Timing Channel Detection. In Proc. 11th International Conference on Computer and Communication Systems (ICCCS) 1043–1049 (2026).
33. Shwartz-Ziv, R. & Armon, A. Tabular Data: Deep Learning Is Not All You Need. *Inf. Fusion*. **81**, 84–90 (2022).
34. Han, J., Huang, C., Shi, F. & Liu, J. Covert Timing Channel Detection Method Based on Time Interval and Payload Length Analysis. *Comput. Secur.* **97**, 101952 (2020).
35. Elsadig, M. A. & Gafar, A. Covert Channel Detection: Machine Learning Approaches. *IEEE Access.* **10**, 38391–38405 (2022).
36. Darwish, O., Al-Fuqaha, A., Ben Brahim, G., Jenhani, I. & Vasilakos, A. Using Hierarchical Statistical Analysis and Deep Neural Networks to Detect Covert Timing Channels. *Appl. Soft Comput.* **82**, 105546 (2019).
37. Al-Eidi, S., Darwish, O., Chen, Y. & Elkhodr, M. Covert Timing Channels Detection Based on Image Processing Using Deep Learning. In Advanced Information Networking and Applications (AINA) 546–555 (2022). (2022).
38. Chen, S., Lang, B., Liu, H., Li, D. & Gao, C. DNS Covert Channel Detection Method Using the LSTM Model. *Comput. Secur.* **104**, 102095 (2021).

39. Li, H., Song, T. & Yang, Y. Generic and Sensitive Anomaly Detection of Network Covert Timing Channels. *IEEE Trans. Dependable Secur. Comput.* **20**, 4085–4100 (2023).
40. Wang, Y. et al. FF-MR: A DoH-Encrypted DNS Covert Channel Detection Method Based on Feature Fusion. *Appl. Sci.* **12**, 12644 (2022).
41. Zhang, L. et al. A Self-Attention Mechanism-Based Model to Detect IPv6 Multi-Field Covert Channels. *IEEE Trans. Cogn. Commun. Netw.*, 258–273 (2025).
42. Sagi, O. & Rokach, L. Ensemble Learning: A Survey. *WIREs Data Min. Knowl. Discov.* **8**, e1249 (2018).
43. Lazzarini, R., Tianfield, H. & Charissis, V. A. Stacking Ensemble of Deep Learning Models for IoT Intrusion Detection. *Knowl. Based Syst.* **279**, 110941 (2023).
44. Alalwany, E. & Mahgoub, I. Stacking Ensemble Deep Learning for Real-Time Intrusion Detection in IoMT Environments. *Sensors* **25**, 624 (2025).
45. Elsadig, M. A. & Gafar, A. Packet Length Covert Channel Detection: An Ensemble Machine Learning Approach. *J. Theoretical Appl. Inform. Technol.* **100**, 7035–7043 (2022).
46. Elsadig, M. A. & Gafar, A. An Ensemble Model to Detect Packet Length Covert Channels. *Int. J. Electr. Comput. Eng.* **13**, 5296–5304 (2023).
47. Li, H. & Chasaki, D. Detecting Covert Channel Attacks on Cyber-Physical Systems. *IET Cyber-Physical Systems: Theory Appl.*, 228–237 (2023).
48. Rahman, M. W. U. et al. AI/ML Based Detection and Categorization of Covert Communication in IPv6 Network. *Cybersecurity* **9**, 33 (2026).
49. Prechelt, L. Early Stopping - But When? In *Neural Networks: Tricks Trade* 55–69 (1998).
50. Goodfellow, I., Bengio, Y. & Courville, A. Deep Learning (MIT, (2016).
51. Bergstra, J. & Bengio, Y. Random Search for Hyper-Parameter Optimization. *J. Mach. Learn. Res.* **13**, 281–305 (2012).
52. Hastie, T., Tibshirani, R. & Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* 2nd edn (Springer, 2009).
53. Fernandez, A., Garcia, S., Herrera, F. & Chawla, N. V. SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary. *J. Artif. Intell. Res.* **61**, 863–905 (2018).
54. Saito, T. & Rehmsmeier, M. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. *PLoS ONE*. **10**, e0118432 (2015).
55. Chicco, D. & Jurman, G. The Advantages of the Matthews Correlation Coefficient (MCC) over F1 Score and Accuracy in Binary Classification Evaluation. *BMC Genom.* **21**, 6 (2020).
56. Bendale, A. & Boult, T. E. Towards Open Set Deep Networks. In Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 1563–1572 (2016). (2016).
57. Vaze, S., Han, K., Vedaldi, A. & Zisserman, A. Open-Set Recognition: A Good Closed-Set Classifier is All You Need. Preprint at (2021). https://arxiv.org/abs/2110.06207
58. Martins, D. M., Salles, R. M. & Goldschmidt, R. R. Adaptive Online Learning Approaches for Detecting Unknown Attacks in Network Traffic: A Systematic Literature Review. *Secur. Priv.* **8**, e70084 (2025).

59. Buczak, A. L. & Guven, E. A. Survey of Data Mining and Machine Learning Methods for Cybersecurity Intrusion Detection. *IEEE Commun. Surv. Tutor.* **18**, 1153–1176 (2016).
60. Iglesias, F., Meghdouri, F., Annessi, R., Zseby, T. & CCgen Injecting Covert Channels into Network Traffic. Secur. Commun. Netw. 1–11 (2022). (2022).

Figures

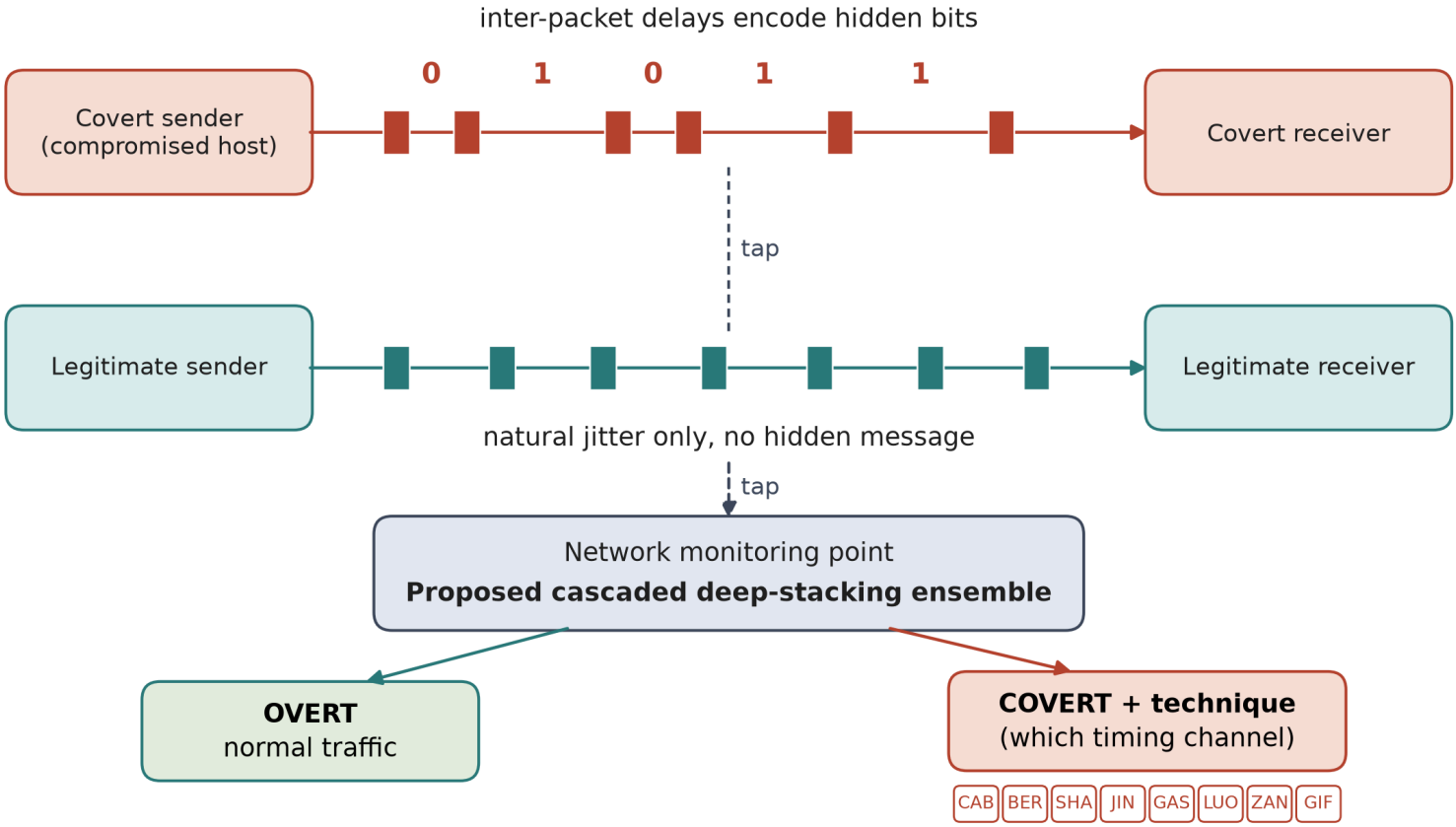
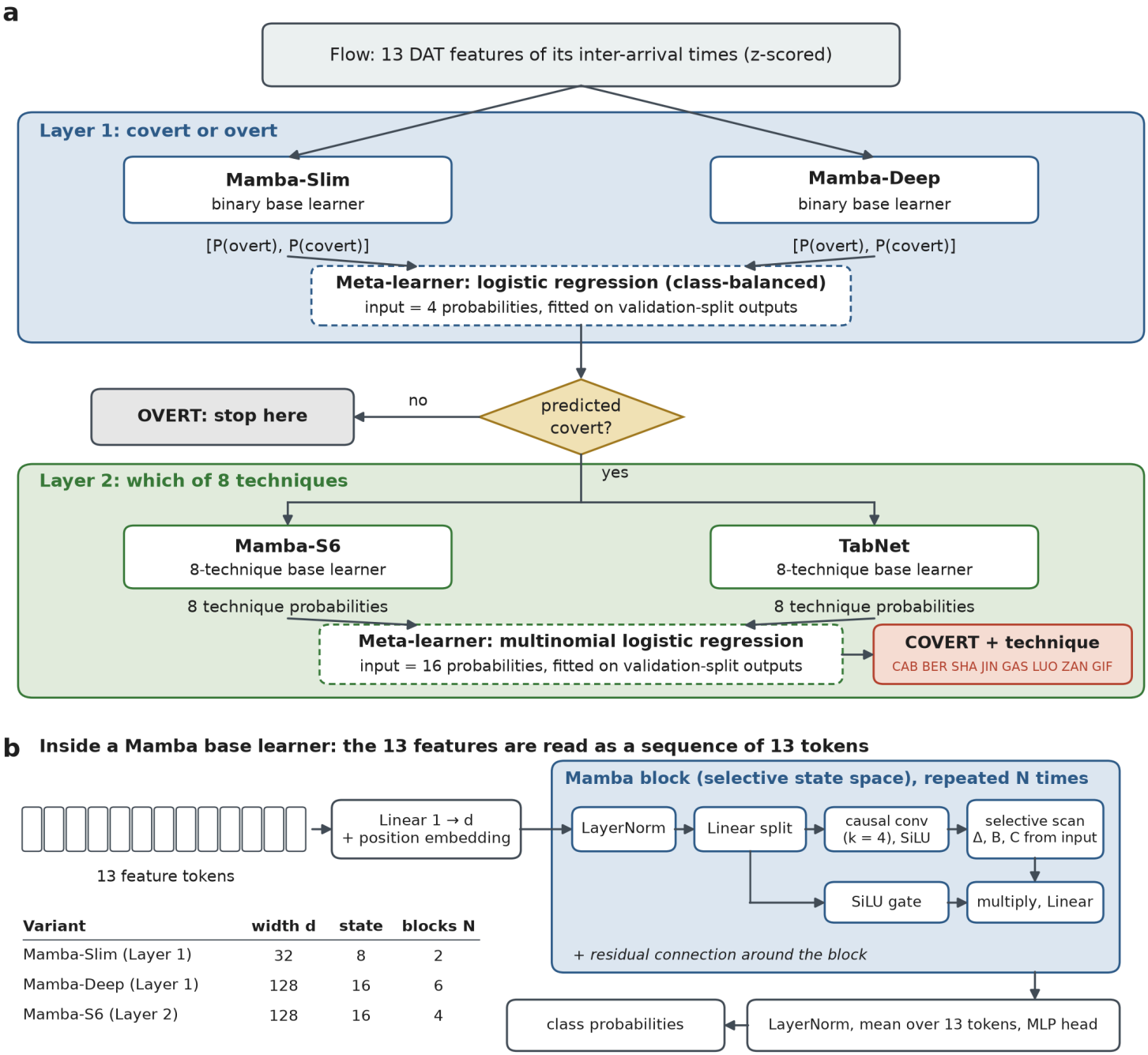


Figure 1

The detection problem addressed in this study, following the covert-timing-channel model of Cabuk et al. [4] as surveyed in [2,3]. A compromised host leaks data by modulating the delays between otherwise legitimate packets (top), so the hidden message lives in the timing itself and never appears in any payload. To a passive observer the result is hard to tell apart from the natural jitter of normal traffic (bottom). The proposed system watches flow timing statistics at a monitoring point and decides whether each flow is overt or covert, and, when covert, which timing technique produced it.



Variant	width d	state	blocks N
Mamba-Slim (Layer 1)	32	8	2
Mamba-Deep (Layer 1)	128	16	6
Mamba-S6 (Layer 2)	128	16	4

All three: conv kernel 4, expansion 2, dropout 0.1.

TabNet (Layer 2): n_d = n_a = 64, 5 decision steps, gamma 1.3.

Base learners and meta-learners use the training and validation splits only; the test split is used once, for the final evaluation.

Figure 2

The cascaded deep-stacking network as implemented. (**a**) Layer 1 stacks Mamba-Slim and Mamba-Deep, the two strongest binary detectors on the validation partition; a class-balanced logistic-regression meta-learner combines their four output probabilities, and only flows it predicts covert reach Layer 2. Layer 2 stacks Mamba-S6 and TabNet, the two strongest technique classifiers on validation, through a multinomial logistic-regression meta-learner over their sixteen technique probabilities. Both meta-learners are fitted on the base learners' outputs for the validation split, and every base learner was chosen by validation

performance alone. (**b**) Inside a Mamba base learner: each of the thirteen features becomes one token, the token sequence passes through N selective state-space blocks, and the pooled output feeds a small classification head. The table gives the width, state size and depth of each variant as set in the code.

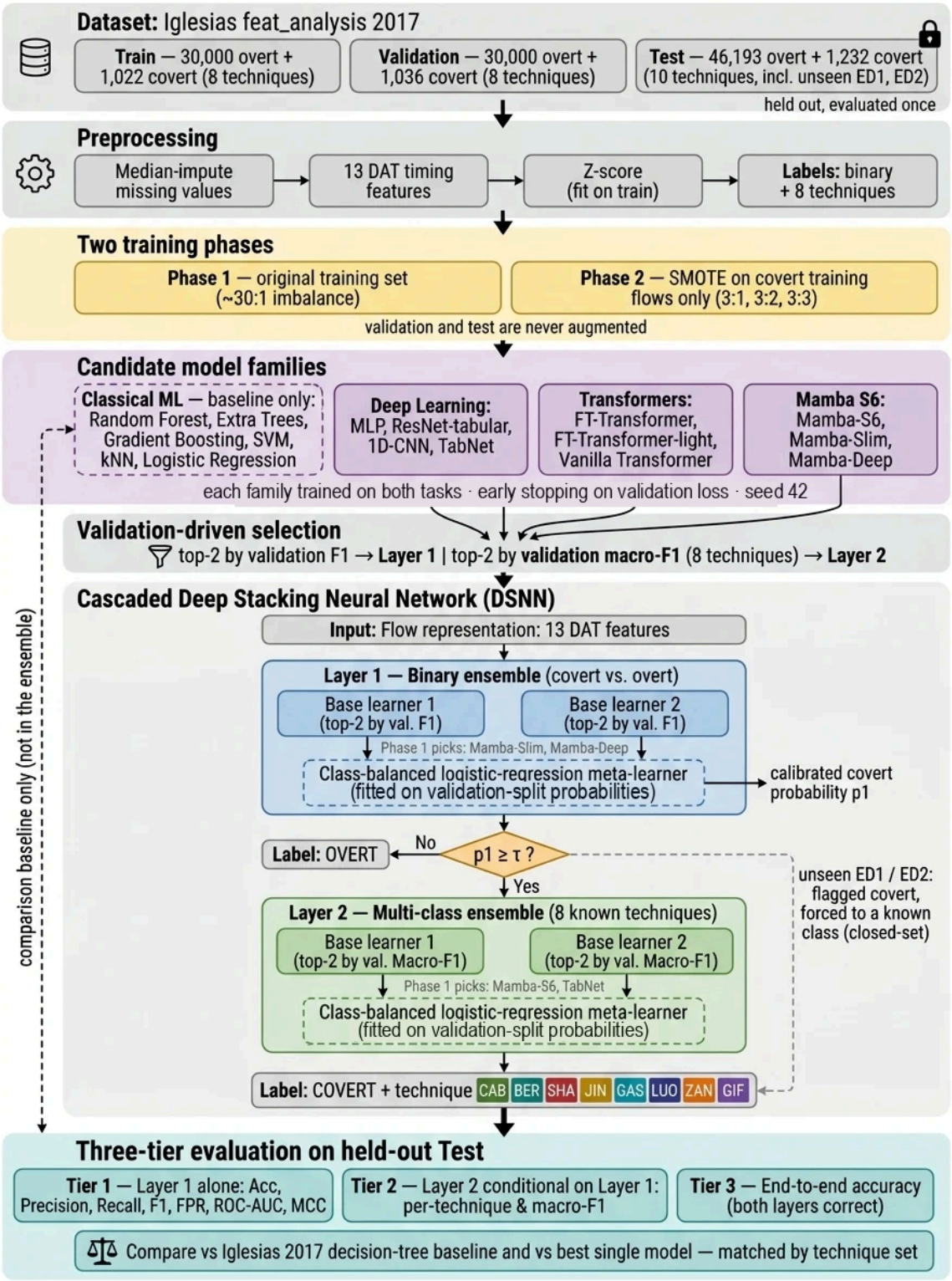


Figure 3

End-to-end experimental pipeline. Flows from the Iglesias *feat_analysis* corpus [6] are preprocessed into the thirteen DAT features, trained under two phases (original and SMOTE-rebalanced [20] training sets), and evaluated across four model families. A validation-driven selection picks the two strongest deep learners per layer for the cascaded deep-stacking network, which is then scored on the held-out test partition under the three-tier protocol and compared with the published decision-tree baseline. Classical machine learning is a comparison baseline only and does not enter the ensemble.

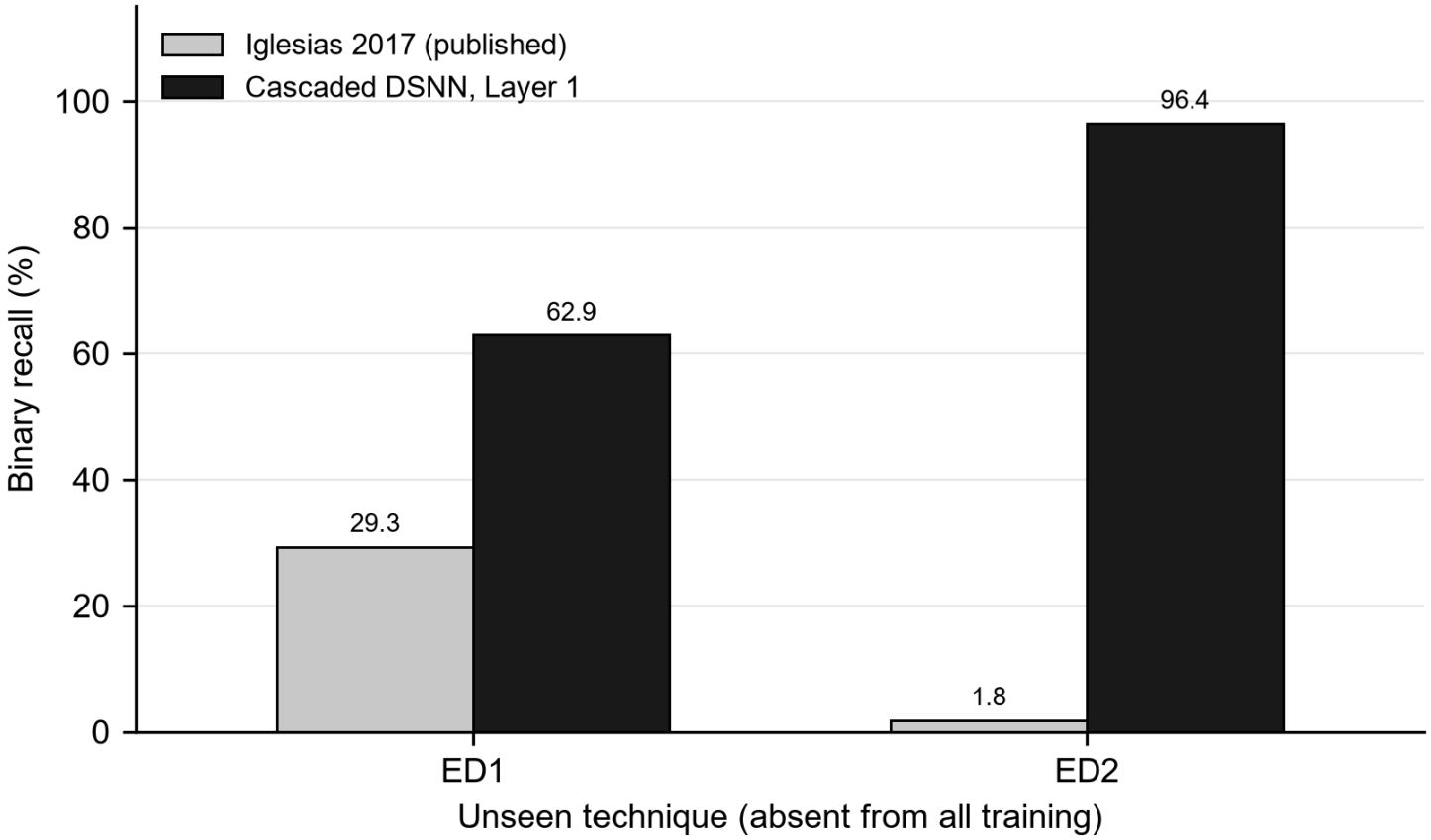


Figure 4

Binary detection recall on the two unseen techniques ED1 and ED2, neither present during training or model selection. Layer 1 reaches 62.86% on ED1 and 96.43% on ED2, against published figures of 29.29% and 1.79%. Iglesias figures are derived from the per-technique false-negative counts in Table 6 of [6] (ED1: 99 of 140 missed; ED2: 55 of 56 missed), which that paper reports for its relevant-feature configuration; they are the only per-technique baseline figures published for this corpus. Single models trained in the same pipeline reach comparable recall (a decision tree 60.71% and 96.43%, Gradient Boosting 71.43% and 100%), so the gain reflects the thirteen-feature representation rather than the cascade.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- SupplementaryInformationYoussefetal.pdf