PACC: PROTOCOL-AWARE CROSS-LAYER COMPRESSION FOR COMPACT NETWORK TRAFFIC REPRESENTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Network traffic classification underpins network security and management but faces growing challenges from pervasive encryption and evolving protocols. Existing representations involve clear trade-offs: hand-crafted flow statistics are efficient but lossy, raw-bit encodings can be accurate but costly, and pre-trained embeddings support transfer but often flatten the protocol stack and entangle cross-layer signals. We observe substantial *redundancy* both *across* and *within* network layers, which existing paradigms do not explicitly address, wasting capacity and encouraging shortcut learning that degrades generalization. To address this, we propose PACC, a redundancy-aware cross-layer representation framework. PACC models protocol layers as multiview inputs and learns faithful, compact projections factorized into *shared* (cross-layer) and *private* (layer-specific) components. It incorporates an information-preserving sparse segment representation to reduce structural repetition and supervised intra-layer filtering to suppress task-irrelevant semantic redundancy. A behavior-guided shared/private decomposition aligns cross-layer consensus while retaining layer-specific information, supported by information-theoretic analysis. Across encrypted application classification, IoT device identification, benign/malicious traffic classification, and Tor traffic detection, PACC achieves the best overall performance. On encrypted traffic, it achieves an F1 score of 0.9685, substantially outperforming existing baselines while using approximately $15\times$ fewer computes and less GPU memory than the latest pre-trained traffic model. An anonymized codebase is provided for review at https://anonymous.4open.science/r/PACC-D5E5.

1 INTRODUCTION

Network traffic classification underpins network security (e.g., intrusion detection (Mirsky et al., 2018; Gupta et al., 2025), malware analysis (Shibahara et al., 2016), anomaly detection (Radford et al., 2018; Li et al., 2025)), network management (e.g., application identification (Jiang et al., 2025; Shaowang et al., 2025; Chu et al., 2025), QoE inference (Kulkarni et al., 2025; Liu et al., 2024; Mangla et al., 2018) and capacity planning (Liu et al., 2023a; Zhu et al., 2021)). As most traffic payload is now encrypted (Google LLC, 2026), classification must infer application and security semantics from partially observable traces, with useful evidence distributed across protocol layers.

Existing representations range from coarse-grained flow statistics (Jiang et al., 2025; Liu et al., 2023b) and packet-sequence features(Liu et al., 2019) to raw-bit/byte encodings(Zhang et al., 2023) and pre-trained embeddings(Lin et al., 2022). Although effective in different settings, they may face a granularity-cost tradeoffs, or entangle signals across protocol layers. Fundamentally, they are not designed to reflect a key insight built upon the layered principles of the Internet: *network traffic exhibits substantial redundancy both across layers and within layers*.

Based on this insight, we distinguish redundancy by its role. *Intra-layer redundancy* includes repetitive structure (e.g., recurring header fields, header options padding, and predictable field states), and task-irrelevant semantic noise (e.g., session-specific artifacts). These inflate representations and consume computation without necessarily improving classification. In contrast, *inter-layer redundancy* expresses related semantics across layers, forming cross-layer consensus that can provide robust ev-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

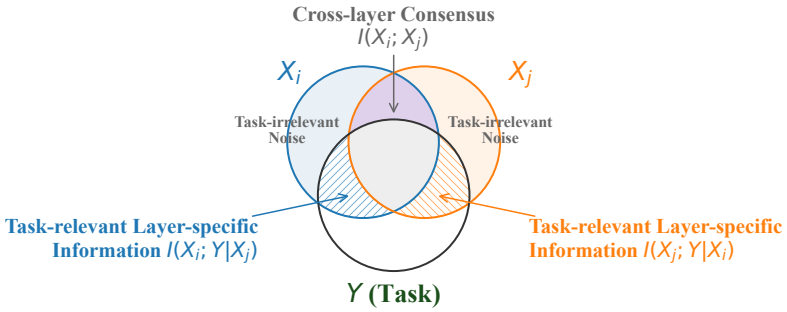


Figure 1: The shared–private principle for cross-layer traffic: capturing cross-layer consensus $I(X_i; X_j)$ and layer-specific, task-relevant evidence such as $I(X_i; Y \mid X_j)$.

idence. Each layer may also contain *non-redundant* task-relevant information unexplained by this consensus. Our goal is therefore to compress repetitive intra-layer structure, suppress task-irrelevant semantic noise, align inter-layer shared evidence, and preserve complementary layer-specific information. Fig. 1 illustrates this shared–private principle through mutual information, $I(\cdot;\cdot)$.

To examine this motivation empirically, we analyze raw layer representations on CipherSpectrum (Wickramasinghe et al., 2025), whose application payload is encrypted. We measure inter-layer consensus using $\mathrm{NMI}(X_i; X_j)$ between discretized layer representations and task relevance using $\mathrm{NMI}(X_i; Y)$ between each layer and the class label. Both are normalized to $[0, 1]$. We also use the compression ratio $\mathrm{CR}(X_i)$ as a proxy for intra-layer structural redundancy: larger values indicate greater compressibility, while values near 1 indicate near-incompressibility. Metric definitions appear in Appendix D.

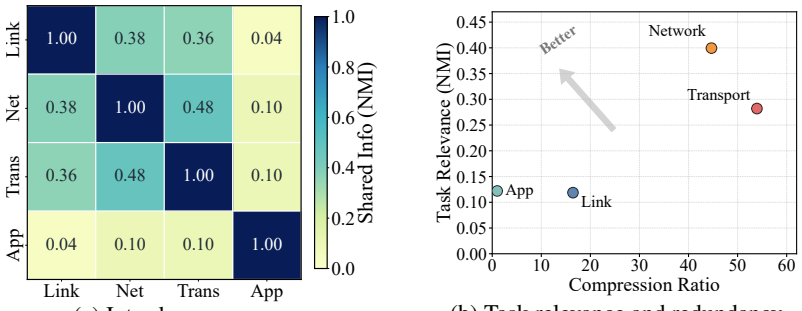
Fig. 2 reveals a clear layer-wise mismatch. The protocol layers exhibit moderate inter-layer consensus (NMI 0.36–0.48), while the Network and Transport layers are highly compressible ($44.7\times$ and $53.9\times$) yet retain relatively high task relevance, especially at the Network layer. In contrast, the encrypted Application layer shows weak cross-layer consensus (NMI 0.04–0.10), near-random compressibility ($\mathrm{CR} \approx 1$), and low task relevance. These non-perfect cross-layer dependencies further indicate private layer-specific evidence, motivating the use of flow behavior to reduce intra-layer nuisance redundancy while preserving shared consensus and complementary task-relevant cues.

Motivated by the above observations, we propose PACC, a redundancy-aware cross-layer traffic representation framework. PACC first applies lossless chunk compression and task-aware filtering to preserve ternary protocol semantics while reducing repetitive structure and task-irrelevant intra-layer noise. It then uses a compact flow-level representation of observable encrypted-flow behavior as a *behavior anchor*, guiding the *shared* components toward inter-layer consensus while preserving *private* components as non-redundant layer-specific residual evidence. Finally, an information-theoretically grounded adaptive aggregation strategy fuses representations according to their sample-specific predictive contribution. Together, these theoretically grounded components produce compact and generalizable cross-layer representations for efficient training, low-latency inference, and diverse downstream tasks. We summarize our main contributions as follows:

- We formalize network traffic as *cross-layer "multiview" data*, identifying redundancy as a first-class property affecting efficiency and generalization.
- Information-theoretic and compression-based analyses demonstrate substantial redundancy across and within layers in raw cross-layer traffic.
- We propose PACC, using global flow behavior as an anchor to learn shared consensus and private non-redundant information across protocol layers.
- Extensive experiments show consistently strong performance across diverse traffic classification tasks, especially emerging encrypted application classification.

2 Problem Formulation

Notation. We study network traffic analysis under a *cross-layer, multiview* abstraction. Given N bidirectional flows with labels $Y \in \{1, \ldots, C\}$, we use M protocol-layer views $X = \{X_1, \ldots, X_M\}$, where $X_i \in \mathbb{R}^{N \times d_{f_i}}$ corresponds to a specific network layer, such as L2/L3/L4



(a) Inter-layer consensus (b) Task relevance and redundancy

Figure 2: Layer-wise analysis on CipherSpectrum. (a) Inter-layer consensus measured by $\mathrm{NMI}(X_i; X_j)$. (b) Task relevance measured by $\mathrm{NMI}(X_i; Y)$, with intra-layer redundancy proxied by the compression ratio. Higher task relevance and lower compression ratio are preferred.

headers and L7 payload content.[1] The views may differ in dimensionality and semantic granularity. We learn a compact fused representation $Z = h_\phi(X_1, \ldots, X_M)$ that remains accurate, efficient, and robust under encryption.

Multiview data non-redundancy. Task-relevant information may reside in both shared and unique view information. Cross-layer consensus reinforces related protocol evidence, but relying on it alone may omit complementary information available only from particular layers.

Definition 2.1. *X_i is considered non-redundant with X_j for Y if and only if there exists $\epsilon > 0$ such that $I(X_i; Y \mid X_j) > \epsilon$ or $I(X_j; Y \mid X_i) > \epsilon$.*

Representation objectives. The learned representation should satisfy three requirements: *layer fidelity*, preserving essential layer-specific information while reducing repetitive structure and task-irrelevant noise; *redundancy-aware complementarity*, combining shared consensus $I(X_i; X_j)$ with non-redundant task information $I(X_i; Y \mid X_j)$; and *compactness and efficiency*, enabling scalable training and low-latency inference with substantially smaller representations than raw-bit encodings. Our goal is to satisfy these requirements jointly under encrypted and heterogeneous traffic settings.

3 Methodology

To operationalize these objectives, PACC follows the pipeline illustrated in Fig. 3. It constructs compact behavioral and protocol-header representations, filters intra-layer semantic noise, separates cross-layer consensus from private residual evidence under behavioral guidance, and adaptively aggregates their sample-specific task contributions.

3.1 Compact Behavioral and Structural Representations

Compact Encrypted-Flow Behavioral Representation. Encryption hides application content, but behavioral fingerprints, such as packet length, direction, and inter-arrival time, remain observable. We summarize these side channels as $pkt_t = (l_t, s_t, \delta_t)$, where l_t is the TCP/UDP payload length, $s_t \in \{+1, -1\}$ is the direction, and δ_t is the inter-arrival time. The first observed source IP defines $+1$, and the reverse direction defines -1.

For each flow, we concatenate compact statistical length, direction, temporal, burst, distribution, and frequency features, and encode them as

$$x_{\mathrm{src}} = [\phi_l, \phi_d, \phi_\delta, \phi_{\mathrm{run}}, \phi_{\mathrm{hist}}, \phi_{\mathrm{fft}}], \qquad X_P = E_P(X_{\mathrm{src}}) \in \mathbb{R}^{N \times d_e}. \tag{1}$$

Here, E_P is jointly optimized with the downstream multi-layer model, and X_P provides the compact flow-level behavioral representation later used to construct the consensus anchor. Detailed feature

[1] Commercial packet capture tools primarily provide access to L2–L4. In encrypted settings, L7 content is generally unavailable and is often ignored or approximated through statistical flow patterns or deep sequence embeddings of observable encrypted traffic behavior.

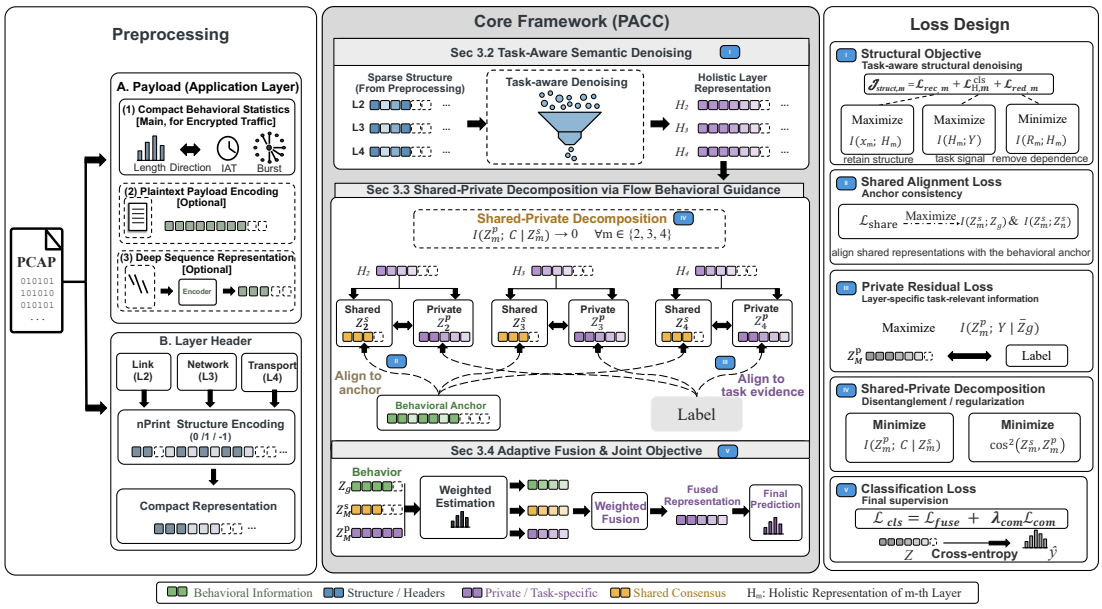


Figure 3: The overall framework of PACC. Compact behavioral features and lossless structural chunks are processed through task-aware semantic denoising, behavior-guided shared–private decomposition, and adaptive fusion. The right panel summarizes the optimization objectives.

construction and preprocessing appear in Appendix B.1, Table 4. The framework can also incorporate deep packet-sequence representations like Lotfollahi et al. (2020) or Liu et al. (2019); we use compact statistics for efficiency and to avoid an additional heavy pretrained backbone.

Lossless Chunk Compression. The link, network, and transport layers provide complementary field presence, values, and protocol-layout patterns. However, dense packet encodings such as nPrint (Holland et al., 2021) contain long repetitive absent-field and padding regions, increasing computation without necessarily adding structural information. We reduce this representation-level redundancy before learning which patterns are useful for classification.

For structural layer $m \in \mathcal{M}$, let $x_m \in \{-1, 0, 1\}^{D_m}$ denote its dense ternary sequence. The value 0 is a valid bit state, whereas -1 denotes an absent or padded field; removing zero-valued entries would therefore destroy protocol semantics. Instead, run-length encoding (Golomb, 1966) merges consecutive identical states:

$$U_m = T_m(x_m) = \{(a_{m,r}, \ell_{m,r}, c_{m,r}, g_{m,r})\}_{r=1}^{K_m}. \tag{2}$$

Each chunk retains its start position $a_{m,r}$, length $\ell_{m,r}$, ternary state $c_{m,r}$, and protocol field-block identifier $g_{m,r}$. Geometry permits exact reconstruction, and field-block identity preserves protocol-layout membership rather than treating runs as unordered entries.

The distinction between absent states and valid protocol bits is essential because dense encodings are aligned to protocol layouts. A repetitive region may occupy many positions yet express only one recurring state. Recording the run and its location condenses this repetition while retaining both its span and semantic identity, reducing input length before neural representation learning.

Proposition 3.1. *Information-Lossless Compression. For any structural layer m, $U_m = T_m(x_m)$ is information-lossless with respect to x_m. Consequently, $I(H_m; x_m) = I(H_m; U_m)$ for any downstream representation H_m.*

The proof appears in Appendix C.2. Thus, chunks reduce physical encoding cost while retaining all information available to subsequent representations.

3.2 Task-Aware Semantic Denoising

The compressed chunks may still include semantic task-irrelevant redundancy, such as checksums or option-related structure weakly related to the label. We therefore introduce intra-layer denoising

to reduce representation dimension and suppress semantic noise while preserving useful protocol evidence.

Chunk encoding and gating. Each chunk combines geometry and protocol semantics through location, length, state, and field-block embeddings. Since identical consecutive states have been merged, a lightweight neighborhood encoder captures the remaining local transitions:

$$\begin{aligned} u_{m,r} &= \mathrm{LN}\big(e_a(a_{m,r}) + e_\ell(\ell_{m,r}) + e_c(c_{m,r}) + e_g(g_{m,r})\big), \\ h_{m,r} &= F_m(u_{m,r-1}, u_{m,r}, u_{m,r+1}). \end{aligned} \tag{3}$$

Encoding cost is linear in the number of chunks; boundary positions and padding are handled by the token mask. Let $R_m = \{h_{m,r}\}_{r=1}^{K_m}$ be the unfiltered structural reference. A learnable gate produces the holistic representation:

$$\alpha_{m,r} = \sigma(f_m(h_{m,r})), \qquad H_m = \mathrm{Pool}(\{\alpha_{m,r} h_{m,r}\}_{r=1}^{K_m}). \tag{4}$$

The gate controls retained chunk contributions, and pooling aggregates their contextual evidence. We train H_m to preserve ternary structure and task information while avoiding replication of R_m.

The unfiltered reference R_m contains both discriminative patterns and semantic noise, whereas H_m is a compact representation for subsequent stages. Since gating alone does not establish which information to preserve, we combine it with structural reconstruction and task supervision, balancing protocol fidelity against label-irrelevant dependence.

Theorem 3.1. *Cross-Entropy Surrogate for Mutual Information (Cover & Thomas, 2006; Poole et al., 2019). Cross-entropy provides the variational lower bound $I(H;Y) \geq H(Y) - \mathcal{L}_{\mathrm{cls}}$. Since $H(Y)$ is fixed for a given dataset, reducing $\mathcal{L}_{\mathrm{cls}}$ increases this lower bound.*

Structural-Fidelity Regularization. Proposition 3.1 establishes information equivalence before filtering. To encourage recoverable ternary semantics after gating, a decoder maps H_m to logits $\mathcal{D}_m(H_m) \in \mathbb{R}^{D_m \times 3}$ for states $\{-1, 0, 1\}$. Its reconstruction loss is

$$\mathcal{L}_{\mathrm{rec},m} = -\frac{1}{n} \sum_{i=1}^{n} \sum_{d=1}^{D_m} \log q_\phi(x_{m,d}^{(i)} \mid H_m^{(i)}). \tag{5}$$

By Theorem 3.1, this maximizes a lower-bound surrogate of $I(x_m; H_m)$, grounding the filtered representation in the original structure. Ternary classification retains the distinction between valid zero-valued bits and absent fields.

Reconstruction complements the lossless transform at a different stage. Chunk compression guarantees recoverability of the structural input, while reconstruction encourages the representation to remain faithful after the gate has reweighted and pooled that input. It therefore protects layer-specific structural evidence during semantic filtering rather than determining task relevance by itself.

Task-relevance Redundancy-Free Filtering. Structural fidelity alone does not remove task-irrelevant dependence. We therefore preserve label information while reducing residual dependence on the unfiltered reference.

Definition 3.1. *Task-relevance Redundancy-Free Filter. H_m is a task-sufficient redundancy-free filtered representation of R_m , if $I(H_m; Y) = I(R_m; Y)$ and $I(R_m; H_m \mid Y) = 0$.*

Proposition 3.2. *Surrogate Criterion for Task-relevance Filtering. Under $Y \leftrightarrow R_m \leftrightarrow H_m$, maximizing $I(H_m; Y)$ encourages the preservation of task-relevant information, while minimizing $I(R_m; H_m)$ toward $I(H_m; Y)$ suppresses dependence on R_m unexplained by the label. Together, these objectives approach the task-sufficient redundancy-free filter in Definition 3.1.*

The precise bounds and proof are given in Appendices B.2 and C.3. This proposition specifies an optimization target rather than a property already achieved by the learned representation.

According to Theorem 3.1, auxiliary classification increases a lower-bound surrogate of $I(H_m; Y)$, preserving discriminative evidence. A CLUB-based penalty (Liang et al., 2023) reduces an estimate of $I(R_m; H_m)$, discouraging dependence beyond that needed for the task:

$$\mathcal{L}_{H,m}^{\mathrm{cls}} = -\frac{1}{n} \sum_{i=1}^{n} \log q_{H,m}\big(y^{(i)} \mid H_m^{(i)}\big), \quad \mathcal{J}_{\mathrm{struct},m} = \mathcal{L}_{\mathrm{rec},m} + \mathcal{L}_{H,m}^{\mathrm{cls}} + \mathcal{L}_{\mathrm{red},m}. \tag{6}$$

Here, $\mathcal{L}_{\text{red},m} = I_{\text{CLUB}}(R_m; H_m)$. The reconstruction term encourages structural fidelity, while the classification and redundancy terms approximate the two information objectives in Proposition 3.2. The CLUB estimator is detailed in Appendix B.3.

3.3 Shared-Private Decomposition via Flow Behavioral Guidance

Filtered layers still contain heterogeneous evidence: some reflects cross-layer behavioral consistency, while other patterns arise from layer-specific protocol mechanisms. We separate these into shared consensus and private residual information so that alignment does not discard complementary task evidence.

Definition 3.2. *Flow-behavior-guided Conditional Latent Decomposition. Two independent encoders map H_m to (Z_m^s, Z_m^p). The shared component captures evidence related to the cross-layer shared task factor C, while the private component preserves task information unexplained by the behavioral anchor. The desired split satisfies $I(Z_m^p; C \mid Z_m^s) = 0$.*

Since C is latent, we map the global packet-sequence embedding to $Z_g = E_g(X_P)$ and use it as an observable anchor. This compact representation summarizes overall encrypted-flow behavior, providing a common reference for structurally different protocol layers.

Proposition 3.3. *Anchor-Induced Shared-Private Decomposition. Assume $H(C \mid Z_g) \le \varepsilon_g$ and $Z_m^s \leftrightarrow C \leftrightarrow Z_g$. Maximizing $I(Z_m^s; Z_g)$ serves as an observable objective for capturing $I(Z_m^s; C)$ and promotes agreement among shared components. It also constrains shared-factor information that may remain in Z_m^p after conditioning on Z_m^s. Complementarily, maximizing $I(Z_m^p; Y \mid Z_g)$ preserves task-relevant evidence beyond the behavioral anchor.*

The precise bounds and proof appear in Appendices B.4 and C.4. These conclusions depend on the stated proxy and mediation assumptions and do not imply that the learned branches satisfy exact conditional independence.

Shared consensus alignment. We approximate the shared objective in Proposition 3.3 with the InfoNCE lower bound (Liang et al., 2023; Oord et al., 2018):

$$\mathcal{L}_{\text{share},m} = -I_{\text{lb}}(Z_m^s; Z_g), \qquad \mathcal{L}_{\text{share}} = \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \mathcal{L}_{\text{share},m}. \tag{7}$$

Minimizing this loss encourages each shared branch to retain structural evidence consistent with the same global behavioral anchor.

Private residual preservation. Let $\bar{Z}_g = \text{sg}(Z_g)$ be the detached anchor. We approximate the conditional task-information objective using label prediction:

$$\mathcal{L}_{\text{priv},m} = -\frac{1}{n} \sum_{i=1}^{n} \log q_{p,m}\big(y^{(i)} \mid \bar{Z}_g^{(i)}, Z_m^{p,(i)}\big). \tag{8}$$

By the conditional form of Theorem 3.1, minimizing this loss increases a lower-bound surrogate of $I(Z_m^p; Y \mid \bar{Z}_g)$, encouraging private evidence that remains useful after behavioral information is given. The decomposition objective is

$$\mathcal{J}_{\text{dec},m} = \mathcal{L}_{\text{share},m} + \mathcal{L}_{\text{priv},m}. \tag{9}$$

Together, these objectives encourage anchor-consistent consensus and complementary task-relevant residual evidence.

3.4 Adaptive Fusion and Joint Objective

Behavioral, shared, and private components provide complementary evidence whose reliability varies across samples. We therefore perform sample-specific fusion to emphasize discriminative components and suppress uncertain predictions.

Index the components by b, with representation Z_b and classifier $q_b(Y \mid Z_b)$. Its posterior provides predictive statistics r_b, including confidence, prediction margin, and normalized certainty. The fusion is computed as

$$\hat{Z}_b = P_b(Z_b), \qquad \alpha_b = \frac{\exp(g_f(\hat{Z}_b, r_b))}{\sum_{b'} \exp(g_f(\hat{Z}_{b'}, r_{b'}))}, \qquad Z_F = \sum_b \alpha_b \hat{Z}_b, \tag{10}$$

Table 1: Classification Results Across Various Domain Traffic Datasets.

Dataset	Metric	Flow Statistics		Raw Packets		Pretrained Embedding		PACC
		AppScanner	FlowPrint	TFE-GNN	nPrint	YaTC	NetMamba	
Task: Encrypted Application Classification								
CipherSpectrum	ACC	0.5356	0.0244	0.2149	0.3037	0.6678	0.5517	**0.9685**
	PRE	0.9278	0.0006	0.2147	0.3933	0.6642	0.5462	**0.9694**
	REC	0.5300	0.0244	0.2149	0.3037	0.6678	0.5517	**0.9685**
	F1	0.6737	0.0012	0.1642	0.2851	0.6633	0.5370	**0.9685**
Task: IoT Device Identification								
UNSW-IoT	ACC	0.8014	0.2438	0.9648	0.5161	**0.9956**	0.9949	0.9937
	PRE	0.8297	0.0114	0.7527	0.7279	0.8960	0.9479	**0.9902**
	REC	0.5296	0.0498	0.7272	0.5057	0.8983	0.8988	**0.9807**
	F1	0.5998	0.0178	0.7356	0.5148	0.8964	0.9166	**0.9852**
Task: Benign/Malicious Traffic Classification								
USTC-TFC	ACC	0.3240	0.2014	0.9428	0.9927	0.9937	0.9927	**0.9948**
	PRE	0.7794	0.0107	0.9290	0.9877	0.9902	0.9870	**0.9912**
	REC	0.3396	0.0502	0.9171	0.9893	0.9807	0.9859	**0.9902**
	F1	0.1632	0.0168	0.9207	0.9884	0.9852	0.9861	**0.9906**
Task: Tor Traffic Detection								
ISCX Tor/NonTor	ACC	0.9905	0.9788	0.9969	0.7285	0.9999	0.9999	**1.0000**
	PRE	0.6360	0.4894	0.9661	0.4685	0.9999	0.9999	**1.0000**
	REC	0.5403	0.5000	0.9628	0.3966	0.9966	0.9966	**1.0000**
	F1	0.5613	0.4946	0.9644	0.4261	0.9982	0.9982	**1.0000**

where P_b maps heterogeneous components into a shared latent space and g_f estimates their sample-specific quality. The resulting Z_F adaptively combines overall behavior, cross-layer consensus, and layer-specific residual evidence.

Optimization. We jointly optimize the tractable surrogate objectives introduced above; their definitions and theoretical bounds appear in Appendix B.3 and the corresponding proofs. Because the adaptive gate relies on component posteriors, cross-entropy supervises both the fused and component classifiers. The overall objective is

$$\mathcal{L} = \mathcal{L}_{\text{fuse}} + \mathcal{L}_{\text{com}} + \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \left(\lambda_{\text{str}} \mathcal{J}_{\text{struct},m} + \lambda_{\text{dec}} \mathcal{J}_{\text{dec},m} \right). \tag{11}$$

This objective jointly trains structural filtering, shared–private decomposition, and sample-specific fusion.

4 Experiments

We evaluate the effectiveness, efficiency, and component contributions of PACC in reducing intra-layer redundancy, exploiting inter-layer consensus, and preserving layer-specific evidence for traffic classification.

4.1 Experimental Setups

Datasets. We evaluate PACC on four real-world datasets covering distinct traffic classification objectives: CipherSpectrum (Wickramasinghe et al., 2025) for encrypted application classification, USTC-TFC (Wang et al., 2017b) for benign/malware traffic classification, UNSW-IoT (Sivanathan et al., 2018) for IoT device classification, and ISCXTor/NonTor (Habibi Lashkari et al., 2017) for Tor/non-Tor recognition. **Baselines** include flow-statistics, raw-bit/byte, and pre-trained traffic representation methods. **Implementation Details** cover data preparation, artifact masking, feature construction, model training, and hyperparameter settings. Detailed dataset statistics, baseline descriptions, evaluation metrics, and implementation configurations are provided in Appendix E.

Efficiency evaluation. Following prior studies (Wang et al., 2024; 2026; Chehade et al., 2025), we profile prepared CipherSpectrum batches on an NVIDIA RTX PRO 6000 Blackwell Server Edition GPU. We report Input/flow, Params, MACs/flow, Peak GPU, Latency/flow, Throughput, and the measured preprocessing and training time. Detailed inference and timing protocols are provided in Appendix E.3.

Table 2: Efficiency comparison on CipherSpectrum. Input size is reported as model-consumed elements for PACC, pixels for packet-image baselines, and graph size for TFE-GNN. Lower is better for Params, MACs, Peak GPU, and Latency, while higher is better for Throughput. Pipeline time reports preprocessing and training; a dash denotes an unreported prestage cost rather than zero.

Method	Input / flow	Params	MACs / flow	Peak GPU	Latency / flow	Throughput	Prestage + training
YaTC	1600 px	1.866M	971.9M	158 MB	0.1482 ms	6,749/s	$— + 1420.8s$
NetMamba	1600 px	1.867M	683.3M	456 MB	0.0481 ms	20,774/s	$— + 1169.5s$
TFE-GNN	903 nodes / 5768 edges	44.364M	157.4M	1297 MB	6.2624 ms	160/s	$1991.7s$
PACC	**608 elems**	**1.273M**	**47.2M**	**106 MB**	**0.0168 ms**	**59,660/s**	$134.2s + 743.0s$

4.2 Results Analysis

Classification results. Table 1 shows consistently strong performance across the four tasks, with the clearest gains on strictly masked CipherSpectrum. PACC achieves ACC and macro F1 of 0.9685, compared with YaTC's ACC of 0.6678 and the strongest baseline macro F1 of 0.6737. Artifact masking removes many shortcut features, making this setting particularly challenging for existing representations.

Flow statistics may smooth out fine-grained cross-layer agreement and rare discriminative events. FlowPrint drops sharply because its fingerprints rely heavily on address and port-related statistics, while AppScanner's behavioral features retain comparatively better performance but lack explicit cross-layer structural modeling. Raw-packet methods face encrypted-byte noise and structural redundancy; graph expansion can further increase cost. YaTC and NetMamba remain competitive, although their upstream distributions do not fully match the strictly masked setting and flattened representations may mix useful headers with encrypted-byte noise.

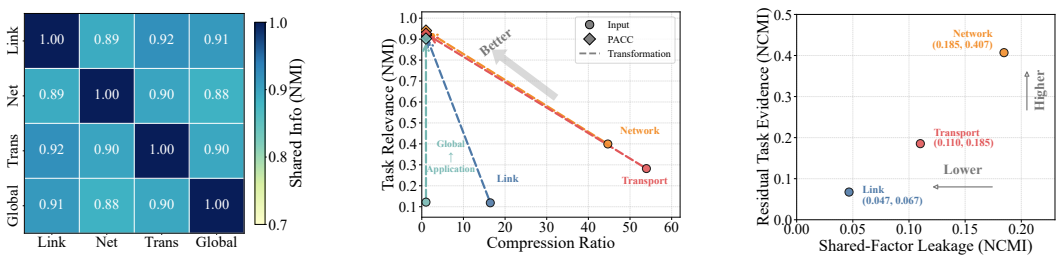
On UNSW-IoT, YaTC and NetMamba achieve slightly higher ACC, but PACC obtains the highest macro PRE, REC, and F1, with F1 of 0.9852. It also achieves the highest reported metrics on USTC-TFC and ISCX Tor/NonTor, with macro F1 of 0.9906 and 1.0000, respectively. These results show competitive or superior performance without expensive upstream pre-training. UNSW-IoT further suggests that useful payload-level protocol patterns can complement structural evidence when strict encryption is absent.

Efficiency results. Table 2 shows that PACC achieves low deployment cost. It uses 608 sparse elements and 47.2M MACs per flow, with 1.273M parameters and 106 MB peak GPU memory. Under the same fixed-batch, logits-only protocol, it reaches 0.0168 ms/flow and 59,660 flows/s, outperforming the compared methods in computation, memory, and inference speed. Including preprocessing, PACC requires 877.2 seconds, which is lower than the reported training or fine-tuning time alone for YaTC (1420.8 seconds), NetMamba (1169.5 seconds), and TFE-GNN (1991.7 seconds).

Overall, the effectiveness and efficiency results support handling redundancy according to its role: compressing intra-layer structural repetition and filtering task-irrelevant noise reduce representation and computational costs, while aligning inter-layer consensus and retaining private residual cues preserve discriminative evidence. Consequently, PACC achieves strong classification performance with compact inputs, low inference overhead, and efficient training.

Redundancy analysis. We inspect learned layer representations on CipherSpectrum. Since encrypted application bytes are nearly random, their learned counterpart is the flow-level behavioral representation from Section 3.1. Shared components follow the empirical protocol in Fig. 2. For private components, we additionally measure shared-factor leakage $I(Z_m^p; Z_g \mid Z_m^s)$ and residual task evidence $I(Z_m^p; Y \mid Z_g)$; normalized estimates are defined in Appendix D.

Fig. 4 shows stronger and more balanced shared consensus than raw views, consistent with extracting common task-relevant factors rather than merely retaining protocol co-occurrence. Shared embeddings move toward higher task relevance and lower redundancy; replacing encrypted application content with compact behavior particularly improves task relevance. Private components exhibit low to moderate anchor leakage after conditioning on Z_m^s, while retaining residual task evidence, especially in the Network and Transport layers. Thus, they provide complementary information rather than simply duplicating shared components.



(a) Shared consensus (b) Shared transformation (c) Private residuals

Figure 4: Layer-wise analysis of PACC embeddings on CipherSpectrum. (a) Shared consensus measured by $\mathrm{NMI}(Z_i^s; Z_j^s)$. (b) Shared-component task relevance measured by $\mathrm{NMI}(Z_m^s; Y)$, with redundancy proxied by the compression ratio. (c) Private-component shared-factor leakage measured by $I(Z_m^p; Z_g \mid Z_m^s)$ and residual task evidence measured by $I(Z_m^p; Y \mid Z_g)$.

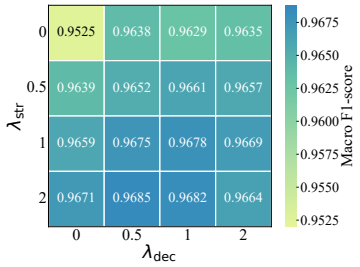


Figure 5: CipherSpectrum macro-F1 sensitivity to structural/decomposition group weights.

Variant	Cipher	UNSW	USTC	Tor
w/o H_m	0.9488	0.9731	0.9892	0.9982
w/o Z_m^s	0.9037	0.9694	0.9737	0.9652
w/o Z_m^p	0.9348	0.9532	0.9720	0.9869
w/o Fusion	0.9645	0.9802	0.9774	0.9991
PACC	**0.9685**	**0.9852**	**0.9906**	**1.0000**

Table 3: Macro-F1 architectural ablation on CipherSpectrum, UNSW-IoT, USTC-TFC, and ISCX Tor/NonTor.

4.3 Ablation Study

We evaluate the architectural contributions of PACC by removing the filtered holistic representation H_m, shared component Z_m^s, private component Z_m^p, and adaptive fusion module. All variants follow the same data splits and evaluation protocol as the complete model. Objective and additional representation ablations are provided in Appendix F.1. Table 3 confirms contributions from all four modules. Removing H_m degrades CipherSpectrum and UNSW-IoT, supporting intra-layer filtering before cross-layer decomposition; removing Z_m^s causes the largest CipherSpectrum drop (0.9685 to 0.9037), confirming the importance of cross-layer consensus; removing Z_m^p lowers CipherSpectrum and UNSW-IoT to 0.9348 and 0.9532, showing that shared information cannot replace non-redundant layer-specific evidence; adaptive fusion provides consistent gains.

4.4 Hyperparameter Sensitivity Analysis

We analyze the balance between structural filtering and shared/private decomposition on CipherSpectrum using macro F1-score. After tuning the complete objective on the validation set, we vary the group-level weights $(\lambda_{\mathrm{str}}, \lambda_{\mathrm{dec}})$ over $\{0, 0.5, 1, 2\} \times \{0, 0.5, 1, 2\}$ while keeping the other settings fixed. Fig. 5 shows the lowest performance when both objective groups are disabled. Moderate-to-large structural weights combined with a moderate decomposition weight perform better, suggesting that intra-layer filtering provides a foundation for organizing shared and private evidence. Excessive group weights may weaken the contribution of the main classification objective. Sensitivity to individual loss weights is analyzed in Appendix F.2.

5 Conclusion

Across four traffic-classification tasks, PACC's protocol-aware design achieves strong performance at low per-flow cost by suppressing repetitive or task-irrelevant intra-layer information through compact chunks and supervised filtering, while preserving cross-layer consensus and complementary layer-specific evidence through shared/private decomposition and adaptive fusion. Appendix F.3 reports further sensitivity and generalization analyses. Beyond the current bidirectional-flow setting, future work will extend PACC to session- and host-level context, adaptive payload representations, and robustness to traffic manipulation.

STATEMENTS

AI USE STATEMENT

We used large language models for English-language polishing, translation, literature search, and assistance with experimental analysis. We also used general-purpose coding agents to assist with writing and debugging research code and experiment scripts. The authors reviewed AI-assisted text, code, and analyses, verified the reported results, and take full responsibility for the scientific claims and content of this paper.

ETHICS STATEMENT

This work aims to support network security and management. Our experiments primarily use publicly available datasets. For data collected from an authorized network, we removed network identifiers and other information that could identify users or devices before analysis. Traffic classification could be misused for surveillance; we therefore advocate its use only in authorized settings with appropriate privacy safeguards.

REPRODUCIBILITY STATEMENT

We provide an anonymized code repository containing the implementation and experimental details to facilitate reproduction of this work. If the paper is accepted, we will publicly release the complete source code and full dataset.

REFERENCES

Adel Chehade, Edoardo Ragusa, Paolo Gastaldo, and Rodolfo Zunino. Energy-efficient deep learning for traffic classification on microcontrollers. *arXiv preprint arXiv:2506.10851*, 2025.

Andrew Chu, Xi Jiang, Shinan Liu, Arjun Bhagoji, Francesco Bronzino, Paul Schmitt, and Nick Feamster. Netssm: Multi-flow and state-aware network trace generation using state-space models. *arXiv preprint arXiv:2503.22663*, 2025.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 2 edition, 2006.

Solomon Golomb. Run-length encodings (corresp.). *IEEE transactions on information theory*, 12 (3):399–401, 1966.

Google LLC. Https encryption on the web. https://transparencyreport.google.com/https/overview?hl=en, 2026. Accessed: 2026-02-01.

Zhaochen Guo, Zhixiang Shen, Xuanting Xie, Liangjian Wen, and Zhao Kang. Disentangling homophily and heterophily in multimodal graph clustering. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 2044–2053, 2025.

Ragini Gupta, Shinan Liu, Ruixiao Zhang, Xinyue Hu, Xiaoyang Wang, Hadjer Benkraouda, Pranav Kommaraju, Phuong Cao, Nick Feamster, and Klara Nahrstedt. Generative active adaptation for drifting and imbalanced network intrusion detection. *arXiv preprint arXiv:2503.03022*, 2025.

Satyandra Guthula, Roman Beltiukov, Navya Battula, Wenbo Guo, and Arpit Gupta. netfound: Foundation model for network security. *arXiv preprint arXiv:2310.17025*, 2023.

Arash Habibi Lashkari, Gerard Draper Gil, Mohammad Mamun, and Ali Ghorbani. Characterization of tor traffic using time based features. pp. 253–262, 01 2017. doi: 10.5220/0006105602530262.

Jordan Holland, Paul Schmitt, Nick Feamster, and Prateek Mittal. New directions in automated traffic analysis. In *Proceedings of the 2021 ACM SIGSAC conference on computer and communications security*, pp. 3366–3383, 2021.

Ting-Li Huoh, Yan Luo, Peilong Li, and Tong Zhang. Flow-based encrypted network traffic classification with graph neural networks. *IEEE Transactions on Network and Service Management*, 20(2):1224–1237, 2022.

Xi Jiang, Shinan Liu, Aaron Gember-Jacobson, Arjun Nitin Bhagoji, Paul Schmitt, Francesco Bronzino, and Nick Feamster. Netdiffusion: Network data augmentation through protocol-constrained traffic generation. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 8(1):1–32, 2024.

Xi Jiang, Shinan Liu, Saloua Naama, Francesco Bronzino, Paul Schmitt, and Nick Feamster. Jiti: Dynamic model serving for just-in-time traffic inference. *Proceedings of the ACM on Networking*, 3(CoNEXT4):1–24, 2025.

Umakant Kulkarni, Khaled Diab, Lianjie Cao, Faraz Ahmed, Shivang Aggarwal, Puneet Sharma, and Sonia Fahmy. Maestro: Qoe-aware dynamic resource allocation in wi-fi networks. *Proc. ACM Netw.*, 3(CoNEXT1), March 2025. doi: 10.1145/3709371. URL https://doi.org/10.1145/3709371.

Ronghua Li, Shinan Liu, Haibo Hu, Qingqing Ye, and Nick Feamster. Wifinger: Fingerprinting noisy iot event traffic using packet-level sequence matching. *arXiv preprint arXiv:2508.03151*, 2025.

Paul Pu Liang, Zihao Deng, Martin Ma, James Zou, Louis-Philippe Morency, and Ruslan Salakhutdinov. Factorized contrastive learning: Going beyond multi-view redundancy. In *Advances in Neural Information Processing Systems*, 2023.

Xinjie Lin, Gang Xiong, Gaopeng Gou, Zhen Li, Junzheng Shi, and Jing Yu. Et-bert: A contextualized datagram representation with pre-training transformers for encrypted traffic classification. In *Proceedings of the ACM Web Conference 2022*, pp. 633–642, 2022.

Chang Liu, Longtao He, Gang Xiong, Zigang Cao, and Zhen Li. Fs-net: A flow sequence network for encrypted traffic classification. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, pp. 1171–1179, 2019. doi: 10.1109/INFOCOM.2019.8737507.

Lingbo Liu, Jiajie Zhen, Guanbin Li, Geng Zhan, Zhaocheng He, Bowen Du, and Liang Lin. Dynamic spatial-temporal representation learning for traffic flow prediction. *IEEE Transactions on Intelligent Transportation Systems*, 22(11):7169–7183, 2021. doi: 10.1109/TITS.2020.3002718.

Shinan Liu, Francesco Bronzino, Paul Schmitt, Arjun Nitin Bhagoji, Nick Feamster, Hector Garcia Crespo, Timothy Coyle, and Brian Ward. Leaf: Navigating concept drift in cellular networks. *Proceedings of the ACM on Networking*, 1(CoNEXT2):1–24, 2023a.

Shinan Liu, Tarun Mangla, Ted Shaowang, Jinjin Zhao, John Paparrizos, Sanjay Krishnan, and Nick Feamster. Amir: Active multimodal interaction recognition from video and network traffic in connected environments. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 7(1):1–26, 2023b.

Shinan Liu, Ted Shaowang, Gerry Wan, Jeewon Chae, Jonatas Marques, Sanjay Krishnan, and Nick Feamster. Serveflow: A fast-slow model architecture for network traffic analysis. *arXiv preprint arXiv:2402.03694*, 2024.

Mohammad Lotfollahi, Mahdi Jafari Siavoshani, Ramin Shirali Hossein Zade, and Mohammdsadegh Saberian. Deep packet: A novel approach for encrypted traffic classification using deep learning. *Soft Computing*, 24(3):1999–2012, 2020.

Tarun Mangla, Emir Halepovic, Mostafa Ammar, and Ellen Zegura. emimic: Estimating http-based video qoe metrics from encrypted network traffic. In *2018 Network Traffic Measurement and Analysis Conference (TMA)*, pp. 1–8. IEEE, 2018.

Xuying Meng, Yequan Wang, Runxin Ma, Haitong Luo, Xiang Li, and Yujun Zhang. Packet representation learning for traffic classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3546–3554, 2022.

Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. Kitsune: An ensemble of autoencoders for online network intrusion detection, 2018. URL https://arxiv.org/abs/1802.09089.

Basil Mustafa, Carlos Riquelme, Joan Puigcerver, Rodolphe Jenatton, and Neil Houlsby. Multimodal contrastive learning with limoe: the language-image mixture of experts. *Advances in Neural Information Processing Systems*, 35:9564–9576, 2022.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Lingfeng Peng, Xiaohui Xie, Sijiang Huang, Ziyi Wang, and Yong Cui. Ptu: Pre-trained model for network traffic understanding. In *32nd IEEE International Conference on Network Protocols, ICNP 2024, Charleroi, Belgium, October 28-31, 2024*, pp. 1–12. IEEE, 2024. doi: 10.1109/ICNP61940.2024.10858503. URL https://doi.org/10.1109/ICNP61940.2024.10858503.

Ben Poole, Sherjil Ozair, Aaron van den Oord, Alexander A. Alemi, and George Tucker. On variational bounds of mutual information. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5171–5180. PMLR, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Benjamin J Radford, Leonardo M Apolonio, Antonio J Trias, and Jim A Simpson. Network traffic anomaly detection using recurrent neural networks. *arXiv preprint arXiv:1803.10769*, 2018.

Eric Rescorla. Transport layer security (tls) protocol version 1.3. RFC 8446, Internet Engineering Task Force (IETF), 2018. URL https://datatracker.ietf.org/doc/html/rfc8446. IETF Standards Track Document.

Vera Rimmer, Davy Preuveneers, Marc Juarez, Tom Van Goethem, and Wouter Joosen. Automated website fingerprinting through deep learning. In *Proceedings of the 25th Network and Distributed System Security Symposium (NDSS 2018)*. Internet Society, 2018.

Ted Shaowang, Shinan Liu, Jonatas Marques, Nick Feamster, and Sanjay Krishnan. Algorithmic data minimization for machine learning over internet-of-things data streams. *arXiv preprint arXiv:2503.05675*, 2025.

Toshiki Shibahara, Takeshi Yagi, Mitsuaki Akiyama, Daiki Chiba, and Takeshi Yada. Efficient dynamic malware analysis based on network behavior using deep learning. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7. IEEE, 2016.

Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, pp. 1928–1943, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356930. doi: 10.1145/3243734.3243768. URL https://doi.org/10.1145/3243734.3243768.

Arunan Sivanathan, Hassan Habibi Gharakheili, Franco Loi, Adam Radford, Chamith Wijenayake, Arun Vishwanath, and Vijay Sivaraman. Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics. *IEEE Transactions on Mobile Computing (TMC)*, 18(8):1745–1759, Aug 2018. doi: 10.1109/TMC.2018.2866249. URL https://www2.ee.unsw.edu.au/~hhabibi/publications.html#18tmc.

Vincent F Taylor, Riccardo Spolaor, Mauro Conti, and Ivan Martinovic. Robust smartphone app identification via encrypted network traffic analysis. *IEEE Transactions on Information Forensics and Security*, 13(1):63–78, 2017.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European conference on computer vision*, pp. 776–794. Springer, 2020.

Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J. Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Florence, Italy, 7 2019. Association for Computational Linguistics.

Thijs Van Ede, Riccardo Bortolameotti, Andrea Continella, Jingjing Ren, Daniel J Dubois, Martina Lindorfer, David Choffnes, Maarten Van Steen, and Andreas Peter. Flowprint: Semi-supervised mobile-app fingerprinting on encrypted network traffic. In *Network and distributed system security symposium (NDSS)*, volume 27, 2020.

Gerry Wan, Shinan Liu, Francesco Bronzino, Nick Feamster, and Zakir Durumeric. Cato: End-to-end optimization of {ML-Based} traffic analysis pipelines. In *22nd USENIX Symposium on Networked Systems Design and Implementation (NSDI 25)*, pp. 1523–1540, 2025.

Tongze Wang, Xiaohui Xie, Wenduo Wang, Chuyi Wang, Youjian Zhao, and Yong Cui. Netmamba: Efficient network traffic classification via pre-training unidirectional mamba. In *2024 IEEE 32nd International Conference on Network Protocols (ICNP)*, pp. 1–11. IEEE, 2024.

Tongze Wang, Xiaohui Xie, Wenduo Wang, Chuyi Wang, Jinzhou Liu, Boyan Huang, Yannan Hu, Youjian Zhao, and Yong Cui. Netmamba+: A framework of pre-trained models for efficient and accurate network traffic classification. *arXiv preprint arXiv:2601.21792*, 2026.

Wei Wang, Ming Zhu, Jinlin Wang, Xuewen Zeng, and Zhongzhen Yang. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In *2017 IEEE International Conference on Intelligence and Security Informatics, ISI 2017, Beijing, China, July 22-24, 2017*, pp. 43–48. IEEE, 2017a. doi: 10.1109/ISI.2017.8004872. URL https://doi.org/10.1109/ISI.2017.8004872.

Wei Wang, Ming Zhu, Xuewen Zeng, Xiaozhou Ye, and Yiqiang Sheng. Malware traffic classification using convolutional neural network for representation learning. In *2017 International Conference on Information Networking (ICOIN)*, pp. 712–717, 2017b. doi: 10.1109/ICOIN.2017.7899588.

Nimesha Wickramasinghe, Arash Shaghaghi, Gene Tsudik, and Sanjay Jha. SoK: Decoding the Enigma of Encrypted Network Traffic Classifiers . In *2025 IEEE Symposium on Security and Privacy (SP)*, pp. 1825–1843, Los Alamitos, CA, USA, May 2025. IEEE Computer Society. doi: 10.1109/SP61157.2025.00165. URL https://doi.ieeecomputersociety.org/10.1109/SP61157.2025.00165.

Xi Xiao, Wentao Xiao, Rui Li, Xiapu Luo, Haitao Zheng, and Shutao Xia. Ebsnn: Extended byte segment neural network for network traffic classification. *IEEE Transactions on Dependable and Secure Computing*, 19(5):3521–3538, 2022. doi: 10.1109/TDSC.2021.3101311.

Yucheng Yin, Zinan Lin, Minhao Jin, Giulia Fanti, and Vyas Sekar. Practical gan-based synthetic ip header trace generation using netshare. In *Proceedings of the ACM SIGCOMM 2022 Conference*, pp. 458–472, 2022.

Xin Yuan, Zhe Lin, Jason Kuen, Jianming Zhang, Yilin Wang, Michael Maire, Ajinkya Kale, and Baldo Faieta. Multimodal contrastive training for visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6995–7004, 2021.

Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. Tensor fusion network for multimodal sentiment analysis. *arXiv preprint arXiv:1707.07250*, 2017.

Haozhen Zhang, Le Yu, Xi Xiao, Qing Li, Francesco Mercaldo, Xiapu Luo, and Qixu Liu. Tfe-gnn: A temporal fusion encoder using graph neural networks for fine-grained encrypted traffic classification. In *Proceedings of the ACM Web Conference 2023*, WWW '23, pp. 2066–2075, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9781450394161. doi: 10.1145/3543507.3583227. URL https://doi.org/10.1145/3543507.3583227.

Ruijie Zhao, Mingwei Zhan, Xianwen Deng, Yanhao Wang, Yijun Wang, Guan Gui, and Zhi Xue. Yet another traffic classifier: A masked autoencoder based traffic transformer with multi-level flow representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 5420–5427, 2023.

Guangmeng Zhou, Xiongwen Guo, Zhuotao Liu, Tong Li, Qi Li, and Ke Xu. Trafficformer: an efficient pre-trained model for traffic data. In *2025 IEEE Symposium on Security and Privacy (SP)*, pp. 1844–1860. IEEE, 2025.

Hang Zhu, Varun Gupta, Satyajeet Singh Ahuja, Yuandong Tian, Ying Zhang, and Xin Jin. Network planning with deep reinforcement learning. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, pp. 258–271, 2021.

A Related Work

Network traffic representations. A network flow is a temporally ordered packet sequence exchanged between two endpoints, with each packet exposing evidence across the protocol stack. Link, network, and transport headers provide metadata and control semantics; the application layer, often encrypted (Rescorla, 2018), still exposes behavior through record sizes, burst patterns, and timing. A **representation** abstracts these flows into a feature space capturing communication patterns (Wang et al., 2017b; Liu et al., 2021). Existing methods organize this layered evidence with different trade-offs.

Flow statistics summarize packet counts, size distributions, inter-arrival times, and burst metrics (Jiang et al., 2025; Liu et al., 2023b; Yin et al., 2022; Wan et al., 2025; Taylor et al., 2017; Van Ede et al., 2020). They are lightweight and condense repetition, but may smooth out fine-grained cross-layer agreement and rare discriminative events, making them fragile across environments as encryption and protocol evolution reduce stable plaintext information. Neural approaches use application-layer packet-sequence features, including sizes, directions, and timing, to model encrypted traffic behavior (Liu et al., 2019; Rimmer et al., 2018; Sirinam et al., 2018). Their performance often depends on feature granularity: coarse features may miss discriminative behavior, while detailed sequences increase computational cost and still largely overlook protocol header-field semantics across layers.

Raw-bit/byte encodings, including Deep Packet (Lotfollahi et al., 2020) and nPrint-style (Holland et al., 2021; Jiang et al., 2024) representations, directly encode headers (Wang et al., 2017a; Zhang et al., 2023; Xiao et al., 2022; Meng et al., 2022) and sometimes payload bytes (Huoh et al., 2022). Although potentially accurate, these sparsely high-dimensional sequences can contain thousands of bits/tokens per flow. Predictable header fields, padding, and session-specific artifacts increase training difficulty and inference cost, as well as the risk of learning incidental correlations or task-irrelevant noise.

Pre-trained embeddings transfer knowledge from large backbones and demonstrate superior performance on downstream tasks (Zhao et al., 2023; Lin et al., 2022; Zhou et al., 2025; Peng et al., 2024; Guthula et al., 2023). However, many pipelines flatten protocol stacks into a single stream, entangling cross-layer signals and underutilizing protocol hierarchy. Their dependence on upstream data distributions can also lead to degradation under downstream domain shifts, requiring costly re-pre-training or adaptation. These limitations motivate a redundancy-aware *cross-layer multiview abstraction* that combines compact layer-wise representations with shared consensus and complementary private information.

Representation optimization in other domains. Multiview and multimodal representation learning offer two relevant directions. *Contrastive alignment* uses cross-modal agreement as supervision (Radford et al., 2021; Tian et al., 2020; Yuan et al., 2021; Mustafa et al., 2022), tending to emphasize stable shared semantics while reducing dependence on syntactic repetition and environment-dependent noise that do not align across views. *Multimodal fusion* integrates modality-specific features into joint embeddings, preserving complementary signals and often improving over uni-modal baselines (Zadeh et al., 2017; Tsai et al., 2019; Guo et al., 2025). These findings suggest treating protocol layers as separate but correlated modalities: learning cross-layer consensus alongside layer-specific information can help mitigate the brittleness of single-layer features. `PACC` follows this perspective while using compact global flow behavior as an anchor and explicitly distinguishing intra-layer nuisance redundancy from inter-layer consensus.

B Supplementary Method Details

B.1 Compact Packet-Sequence Feature Construction

For each flow, we compute six groups of statistics from the observable packet attributes defined in Section 3.1. All statistics are extracted independently per flow; raw payload bytes are not inspected. Table 4 lists the features in each group.

The length-scale group summarizes the amount of traffic and variation in packet sizes. The zero-payload ratio distinguishes packets carrying no TCP/UDP payload from those carrying data. Di-

Table 4: Compact packet-sequence feature construction.

View	Extracted statistics	Captured behavior
Length scale	Packet count, total bytes, mean/std/median/90th-percentile/max packet length, and zero-payload ratio	Traffic volume and packet-size variation.
Direction	Forward/backward packet ratios, forward/backward byte ratios, and direction-switching rate	Endpoint dominance and request-response interaction.
Temporal	Summary statistics of log-compressed inter-arrival times	Transmission pace, timing fluctuation, and delayed responses.
Run-level bursts	Number of same-direction runs, average/max run length, and average/max run bytes	Alternating exchanges and directional bursts.
Distribution	Normalized histograms of packet lengths and log-compressed inter-arrival times	Distributional shape beyond low-order moments.
Frequency	Low-frequency Fourier magnitudes of signed packet-length and timing sequences	Burst periodicity and request-response rhythm.

rectional packet and byte ratios characterize how traffic is distributed between endpoints, while the direction-switching rate captures how often the flow alternates between them.

Temporal statistics summarize inter-arrival times after log compression, reducing the influence of extreme timing values. Consecutive packets traveling in the same direction form a run; run counts, lengths, and bytes distinguish short alternating exchanges from longer directional bursts. Normalized length and timing histograms retain coarse distributional shape that may not be captured by summary statistics alone.

For frequency features, packet lengths are log-scaled and multiplied by packet direction before the Fourier transform so that the sequence retains bidirectional structure. We retain only leading low-frequency magnitudes of the packet-length and timing sequences. These components summarize coarse burst periodicity and request-response rhythm while suppressing high-frequency packet-level noise.

The resulting per-flow features are concatenated as specified in Eq. 1. Their preparation pipeline is

$$\mathrm{PCAP} \to X_{\mathrm{src}} \in \mathbb{R}^{N \times d_p} \to E_P \to X_P \in \mathbb{R}^{N \times d_e}. \tag{12}$$

B.2 Complete Bounds for Task-relevance Filtering

Let $\underline{I}(H_m; Y)$ be a lower bound of $I(H_m; Y)$ and $\overline{I}(R_m, H_m)$ an upper bound of $I(R_m; H_m)$. Under the Markov relation $Y \leftrightarrow R_m \leftrightarrow H_m$, Proposition 3.2 gives

$$0 \le I(R_m; Y) - I(H_m; Y) \le I(R_m; Y) - \underline{I}(H_m; Y), \tag{13}$$

and

$$0 \le I(R_m; H_m \mid Y) \le \overline{I}(R_m, H_m) - \underline{I}(H_m; Y). \tag{14}$$

The first bound controls task-information loss, while the second controls dependence on R_m unexplained by the label. Maximazing the task-information lower bound toward $I(R_m; Y)$ and reducing the retained-dependence upper bound toward that lower bound provide the filtering surrogate. The proof appears in Appendix C.3.

B.3 CLUB Estimation for Semantic Denoising

To implement the redundancy-reduction objective, we use the CLUB estimator (Liang et al., 2023) for the dependence between the unfiltered representation R_m and filtered representation H_m:

$$\begin{aligned} I_{\mathrm{CLUB}}(R_m; H_m) = \mathbb{E}_{(R_m, H_m^+) \sim p(R_m, H_m)} \big[f^*(R_m, H_m^+) \big] \\ - \mathbb{E}_{R_m \sim p(R_m),\, H_m^- \sim p(H_m)} \big[f^*(R_m, H_m^-) \big]. \end{aligned} \tag{15}$$

Here, H_m^+ is paired with R_m from the same flow, whereas H_m^- is sampled from the marginal distribution of H_m. The critic is $f^*(R_m, H_m) = \log q_\psi(H_m \mid R_m)$; in our implementation, R_m parameterizes a Gaussian conditional distribution over H_m, and the critic evaluates its log-likelihood. We use $I_{\mathrm{CLUB}}(R_m; H_m)$ as the redundancy penalty $\mathcal{L}_{\mathrm{red},m}$ in the structural objective.

B.4 Complete Bounds for Anchor-Induced Decomposition

Proposition 3.3 uses $I(Z_m^s; Z_g)$ as the observable shared objective. Under $H(C \mid Z_g) \leq \varepsilon_g$ and $Z_m^s \leftrightarrow C \leftrightarrow Z_g$, this objective approximates information about the latent shared factor:

$$I(Z_m^s; Z_g) \leq I(Z_m^s; C) \leq I(Z_m^s; Z_g) + \varepsilon_g. \tag{16}$$

For two structural layers m and n, alignment with the same anchor gives

$$I(Z_m^s; Z_n^s) \geq I(Z_m^s; Z_g) + I(Z_n^s; Z_g) - H(Z_g). \tag{17}$$

Increasing both anchor-dependence terms therefore raises a lower bound on pairwise shared agreement. Moreover,

$$I(Z_m^p; C \mid Z_m^s) \leq H(C) - I(Z_m^s; Z_g). \tag{18}$$

Increasing shared-branch dependence on Z_g reduces this upper bound on additional shared-factor information in the private component. For residual task information, the detached anchor $\bar{Z}_g = \mathrm{sg}(Z_g)$ has the same observed values as Z_g but blocks gradients through the anchor in the private objective. Conditional cross-entropy yields

$$I(Z_m^p; Y \mid \bar{Z}_g) \geq H(Y \mid \bar{Z}_g) - \mathcal{L}_{\mathrm{priv},m}. \tag{19}$$

The InfoNCE objective similarly supplies a tractable lower-bound surrogate for $I(Z_m^s; Z_g)$. Together, these bounds explain why shared-branch alignment targets cross-layer consensus while conditional private supervision retains task evidence beyond the behavioral anchor. The proof of the first three bounds appears in Appendix C.4.

C Theoretical Analysis

C.1 Properties of Mutual Information and Representations

In this section, we enumerate basic properties of mutual information used in the proofs. For any random variables x, y, and z, we have:

(P_1) Non-negativity:

$$I(x; y) \geq 0, \qquad I(x; y \mid z) \geq 0, \qquad H(x \mid y) \geq 0. \tag{20}$$

(P_2) Chain rule:

$$I(x, y; z) = I(y; z) + I(x; z \mid y). \tag{21}$$

(P_3) Entropy expansion of conditional mutual information:

$$I(x; y \mid z) = H(y \mid z) - H(y \mid x, z). \tag{22}$$

(P_4) Interaction-information identity:

$$I(x; y; z) = I(x; y) - I(x; y \mid z) = I(y; z) - I(y; z \mid x) = I(x; z) - I(x; z \mid y). \tag{23}$$

(P_5) Data processing inequality: if $x \leftrightarrow y \leftrightarrow z$, then

$$I(x; z) \leq I(x; y). \tag{24}$$

C.2 Proof of Proposition 3.1

By construction, the chunk transform T_m maps the dense ternary sequence x_m to U_m by merging consecutive positions with identical states and recording the start position, length, state, and semantic field-block identifier of each chunk. Thus,

$$U_m = T_m(x_m), \qquad H(U_m \mid x_m) = 0. \tag{25}$$

Conversely, the recorded chunk information reconstructs the original dense sequence exactly:

$$x_m = T_m^{-1}(U_m), \qquad H(x_m \mid U_m) = 0. \tag{26}$$

We first expand the joint mutual information by the chain rule:

$$I(H_m; x_m, U_m) \overset{(P_2)}{=} I(H_m; x_m) + I(H_m; U_m \mid x_m). \tag{27}$$

The conditional term satisfies

$$\begin{aligned} 0 &\overset{(P_1)}{\leq} I(H_m; U_m \mid x_m) \\ &\overset{(P_3)}{=} H(U_m \mid x_m) - H(U_m \mid H_m, x_m) \\ &\leq H(U_m \mid x_m) \overset{25}{=} 0. \end{aligned} \tag{28}$$

Therefore,

$$I(H_m; U_m \mid x_m) = 0. \tag{29}$$

Then,

$$\begin{aligned} I(H_m; x_m, U_m) &\overset{27}{=} I(H_m; x_m) + I(H_m; U_m \mid x_m) \\ &\overset{29}{=} I(H_m; x_m). \end{aligned} \tag{30}$$

Similarly, using the inverse deterministic reconstruction in Eq. 26,

$$I(H_m; x_m \mid U_m) = 0. \tag{31}$$

Applying the chain rule again,

$$\begin{aligned} I(H_m; x_m, U_m) &\overset{(P_2)}{=} I(H_m; U_m) + I(H_m; x_m \mid U_m) \\ &\overset{31}{=} I(H_m; U_m). \end{aligned} \tag{32}$$

Thus,

$$I(H_m; x_m) \overset{30}{=} I(H_m; x_m, U_m) \overset{32}{=} I(H_m; U_m). \tag{33}$$

This proves that the chunk representation is sparse in length but information-lossless with respect to the dense structural input.

C.3 Proof of Proposition 3.2

Fix the learned filtering map. Since the filter takes the unfiltered structural representation R_m as input and does not directly observe the label Y, its conditional distribution satisfies

$$p(H_m \mid R_m, Y) = p(H_m \mid R_m). \tag{34}$$

Therefore,

$$\begin{aligned} I(H_m; Y \mid R_m) &\overset{(P_3)}{=} H(H_m \mid R_m) - H(H_m \mid R_m, Y) \\ &\overset{34}{=} 0. \end{aligned} \tag{35}$$

Hence, the filtering architecture induces the Markov relation $Y \leftrightarrow R_m \leftrightarrow H_m$. We first derive the task-information gap. By the chain rule,

$$\begin{aligned} I(R_m, H_m; Y) &\overset{(P_2)}{=} I(R_m; Y) + I(H_m; Y \mid R_m) \\ &\overset{35}{=} I(R_m; Y). \end{aligned} \tag{36}$$

Similarly,

$$\begin{aligned} I(R_m, H_m; Y) &\overset{(P_2)}{=} I(H_m; Y) + I(R_m; Y \mid H_m) \\ &\overset{(P_1)}{\geq} I(H_m; Y). \end{aligned} \tag{37}$$

Thus,

$$I(H_m; Y) \leq I(R_m; Y). \tag{38}$$

Since $\underline{I}(H_m; Y)$ is a lower bound of $I(H_m; Y)$, we obtain

$$0 \overset{38}{\leq} I(R_m; Y) - I(H_m; Y) \leq I(R_m; Y) - \underline{I}(H_m; Y). \tag{39}$$

This proves the first bound.

Next, using the interaction-information identity in Eq. 23, we have

$$I(R_m; H_m) - I(R_m; H_m \mid Y) \overset{(P_4)}{=} I(H_m; Y) - I(H_m; Y \mid R_m). \tag{40}$$

Therefore,

$$\begin{aligned} I(R_m; H_m) &\overset{40}{=} I(H_m; Y) - I(H_m; Y \mid R_m) + I(R_m; H_m \mid Y) \\ &\overset{35}{=} I(H_m; Y) + I(R_m; H_m \mid Y). \end{aligned} \tag{41}$$

Equivalently,

$$I(R_m; H_m \mid Y) = I(R_m; H_m) - I(H_m; Y). \tag{42}$$

Because $\overline{I}(R_m, H_m)$ upper-bounds $I(R_m; H_m)$ and $\underline{I}(H_m; Y)$ lower-bounds $I(H_m; Y)$,

$$0 \overset{(P_1)}{\leq} I(R_m; H_m \mid Y) \overset{42}{=} I(R_m; H_m) - I(H_m; Y) \leq \overline{I}(R_m, H_m) - \underline{I}(H_m; Y). \tag{43}$$

This proves the second bound.

If $I(R_m; Y) - \underline{I}(H_m; Y)$ approaches zero, Eq. 39 forces

$$I(H_m; Y) = I(R_m; Y). \tag{44}$$

If $\overline{I}(R_m, H_m) - \underline{I}(H_m; Y)$ approaches zero, Eq. 43 forces

$$I(R_m; H_m \mid Y) = 0. \tag{45}$$

These two limiting conditions are exactly the two requirements in Definition 3.1. Therefore, maximizing the task-information lower bound and minimizing the retained-dependence upper bound provides a tractable surrogate for approaching the task-sufficient redundancy-free filter. This proves the proposition.

C.4 Proof of Proposition 3.3

(i) Proxy to the latent shared factor. Under the shared-factor relation $Z_m^s \leftrightarrow C \leftrightarrow Z_g$, the data processing inequality gives

$$I(Z_m^s; Z_g) \overset{(P_5)}{\leq} I(Z_m^s; C). \tag{46}$$

Next, by the interaction-information identity,

$$I(Z_m^s; C) - I(Z_m^s; C \mid Z_g) \overset{(P_4)}{=} I(Z_m^s; Z_g) - I(Z_m^s; Z_g \mid C). \tag{47}$$

Since $Z_m^s \leftrightarrow C \leftrightarrow Z_g$,

$$I(Z_m^s; Z_g \mid C) = 0. \tag{48}$$

Therefore,

$$\begin{aligned} I(Z_m^s; C) &\overset{47}{=} I(Z_m^s; Z_g) + I(Z_m^s; C \mid Z_g) - I(Z_m^s; Z_g \mid C) \\ &\overset{48}{=} I(Z_m^s; Z_g) + I(Z_m^s; C \mid Z_g) \\ &\overset{(P_3)}{=} I(Z_m^s; Z_g) + H(C \mid Z_g) - H(C \mid Z_m^s, Z_g) \\ &\overset{(P_1)}{\leq} I(Z_m^s; Z_g) + H(C \mid Z_g) \\ &\leq I(Z_m^s; Z_g) + \varepsilon_g. \end{aligned} \tag{49}$$

Combining Eq. 46 and Eq. 49, we obtain

$$I(Z_m^s; Z_g) \leq I(Z_m^s; C) \leq I(Z_m^s; Z_g) + \varepsilon_g. \tag{50}$$

Thus, $I(Z_m^s; Z_g)$ is an observable proxy for the latent shared-factor information $I(Z_m^s; C)$, with approximation error bounded by ε_g.

(ii) Anchor-induced pairwise shared agreement. For two structural modalities m and n, the interaction-information identity gives

$$\begin{aligned} I(Z_m^s; Z_n^s; Z_g) &\overset{(P_4)}{=} I(Z_m^s; Z_n^s) - I(Z_m^s; Z_n^s \mid Z_g), \\ I(Z_m^s; Z_n^s; Z_g) &\overset{(P_4)}{=} I(Z_n^s; Z_g) - I(Z_n^s; Z_g \mid Z_m^s). \end{aligned} \tag{51}$$

By the chain rule,

$$\begin{aligned} I(Z_m^s, Z_n^s; Z_g) &\overset{(P_2)}{=} I(Z_m^s; Z_g) + I(Z_n^s; Z_g \mid Z_m^s) \\ &\overset{51}{=} I(Z_m^s; Z_g) + I(Z_n^s; Z_g) - I(Z_m^s; Z_n^s; Z_g). \end{aligned} \tag{52}$$

Combining Eq. 51 and Eq. 52, we have

$$\begin{aligned} I(Z_m^s; Z_n^s) =& I(Z_m^s; Z_n^s \mid Z_g) + I(Z_m^s; Z_g) \\ &+ I(Z_n^s; Z_g) - I(Z_m^s, Z_n^s; Z_g). \end{aligned} \tag{53}$$

Then,

$$\begin{aligned} I(Z_m^s; Z_n^s) &\overset{53}{=} I(Z_m^s; Z_n^s \mid Z_g) + I(Z_m^s; Z_g) + I(Z_n^s; Z_g) - I(Z_m^s, Z_n^s; Z_g) \\ &\overset{(P_1)}{\geq} I(Z_m^s; Z_g) + I(Z_n^s; Z_g) - I(Z_m^s, Z_n^s; Z_g) \\ &\overset{(P_3)}{=} I(Z_m^s; Z_g) + I(Z_n^s; Z_g) - H(Z_g) + H(Z_g \mid Z_m^s, Z_n^s) \\ &\overset{(P_1)}{\geq} I(Z_m^s; Z_g) + I(Z_n^s; Z_g) - H(Z_g). \end{aligned} \tag{54}$$

Therefore, optimizing each structural shared branch toward the same anchor Z_g increases a lower bound of the pairwise shared agreement between Z_m^s and Z_n^s.

(iii) Connection to the decomposition target. Definition 3.2 requires the private branch to contain no additional shared-factor information once the shared branch is given. Since $I(Z_m^p; C \mid Z_m^s)$ is not directly optimized, we use its upper bound:

$$\begin{aligned} I(Z_m^p; C \mid Z_m^s) &\overset{(P_3)}{=} H(C \mid Z_m^s) - H(C \mid Z_m^p, Z_m^s) \\ &\overset{(P_1)}{\leq} H(C \mid Z_m^s) \\ &= H(C) - I(Z_m^s; C) \\ &\overset{46}{\leq} H(C) - I(Z_m^s; Z_g). \end{aligned} \tag{55}$$

Thus, increasing the observable anchor dependence $I(Z_m^s; Z_g)$ reduces an upper bound of the shared-factor information that can remain in Z_m^p after conditioning on Z_m^s. Together with Eq. 50, this shows that the anchor objective both approximates shared-factor extraction and supports the decomposition condition in Definition 3.2. This proves the proposition.

D Empirical Metrics for Redundancy Analysis

For each layer representation X_i, we first obtain a discrete assignment $A_i = q_K(X_i)$ by K-means vector quantization with $K = 32$, where q_K denotes the quantization map. For the n-th sample, let $A_i^{(n)} \in \{1, \ldots, K\}$ denote its quantized assignment. We use the same quantization procedure for all layers.

The mutual information terms are estimated from empirical histograms. For two layers X_i and X_j, we define

$$\hat{p}_{ij}(a, b) = \frac{1}{N} \sum_{n=1}^{N} \mathbf{1}\left[A_i^{(n)} = a,\ A_j^{(n)} = b\right], \tag{56}$$

with marginals $\hat{p}_i(a) = \sum_b \hat{p}_{ij}(a, b)$ and $\hat{p}_j(b) = \sum_a \hat{p}_{ij}(a, b)$. The plug-in mutual information estimator is

$$I(A_i; A_j) = \sum_{a=1}^{K} \sum_{b=1}^{K} \hat{p}_{ij}(a, b) \log \frac{\hat{p}_{ij}(a, b)}{\hat{p}_i(a)\hat{p}_j(b)}, \tag{57}$$

where zero-probability terms are omitted. The entropy is computed as $H(A_i) = -\sum_a \hat{p}_i(a) \log \hat{p}_i(a)$.

The cross-layer consensus between two layers X_i and X_j is measured by normalized mutual information:

$$\mathrm{NMI}(X_i; X_j) = \frac{2I(A_i; A_j)}{H(A_i) + H(A_j)}. \tag{58}$$

Similarly, the task relevance of layer X_i is measured by the normalized mutual information between its discrete assignment and the class label Y:

$$\mathrm{NMI}(X_i;Y) = \frac{2I(A_i;Y)}{H(A_i)+H(Y)}. \tag{59}$$

Both quantities lie in $[0,1]$, where larger values indicate stronger shared information.

To estimate the compressible structural part of intra-layer redundancy, we serialize each sample representation $x_i^{(n)}$ into bytes and compute its generic compression ratio:

$$\mathrm{CR}(X_i) = \frac{\sum_n |\mathrm{bytes}(x_i^{(n)})|}{\sum_n |\mathrm{zlib}(\mathrm{bytes}(x_i^{(n)}))|}. \tag{60}$$

Here, $|\cdot|$ denotes the byte length, and $\mathrm{zlib}(\cdot)$ denotes generic lossless compression using the zlib algorithm. A larger $\mathrm{CR}(X_i)$ indicates that the layer representation contains more compressible structural repetition, while $\mathrm{CR}(X_i) \approx 1$ indicates an almost incompressible representation.

For the private-branch analysis in Fig. 4c, we use the same K-means quantization protocol. For each structural modality m, let Z_m^s and Z_m^p denote the learned shared and private representations, and let Z_g denote the global flow-behavior anchor. We obtain discrete assignments

$$S_m = q_K(Z_m^s), \qquad P_m = q_K(Z_m^p), \qquad G = q_K(Z_g).$$

The additional shared-factor information retained by the private branch is estimated by

$$I(P_m;G \mid S_m) = \sum_{p,g,s} \hat{p}(p,g,s) \log \frac{\hat{p}(p,g,s)\hat{p}(s)}{\hat{p}(p,s)\hat{p}(g,s)}, \tag{61}$$

and reported as the normalized conditional mutual information

$$\mathrm{NCMI}(Z_m^p; Z_g \mid Z_m^s) = \frac{2I(P_m;G \mid S_m)}{H(P_m \mid S_m) + H(G \mid S_m)}. \tag{62}$$

A smaller value indicates less residual shared-factor leakage in the private branch after the shared representation is given.

Similarly, the residual task evidence preserved by the private branch is estimated by

$$I(P_m;Y \mid G) = \sum_{p,y,g} \hat{p}(p,y,g) \log \frac{\hat{p}(p,y,g)\hat{p}(g)}{\hat{p}(p,g)\hat{p}(y,g)}, \tag{63}$$

and reported as

$$\mathrm{NCMI}(Z_m^p; Y \mid Z_g) = \frac{2I(P_m;Y \mid G)}{H(P_m \mid G) + H(Y \mid G)}. \tag{64}$$

A larger value indicates that the private branch retains more task-relevant residual evidence after the global flow-behavior anchor is given.

E Experimental Settings and Protocols

E.1 Datasets, Baselines, and Evaluation Metrics

Datasets. CipherSpectrum (Wickramasinghe et al., 2025) is a high-strength encrypted traffic benchmark for application classification under strict artifact masking. USTC-TFC (Wang et al., 2017b) contains benign and malware traffic. UNSW-IoT (Sivanathan et al., 2018) supports IoT device classification and includes mixed application protocols that may retain useful packet-level payload patterns. ISCXTor/NonTor (Habibi Lashkari et al., 2017) supports Tor/non-Tor recognition, where compact global traffic behavior can provide discriminative evidence.

Table 5 reports the numbers of classes and flows, the final train/validation/test splits, and the corresponding data sizes. Flow-PCAP size is measured after organizing packets into bidirectional-flow samples and splitting them. Processed input size denotes the preprocessed representation size after reducing structurally repetitive information.

Table 5: Dataset statistics under the final evaluation protocol.

Dataset	Task	Classes	Flow Samples	Train/Val/Test	Flow-PCAP Size	Processed Input Size
CipherSpectrum	Encrypted Application	41	41,000	32,800 / 4,100 / 4,100	3.58 GB	224.39 MB
USTC-TFC	Benign/Malware Traffic	20	444,071	355,178 / 44,440 / 44,453	3.65 GB	1.60 GB
UNSW-IoT	IoT Device	22	41,176	32,980 / 4,098 / 4,098	671.14 MB	100.63 MB
ISCXTor/NonTor	Tor/Non-Tor	2	68,258	54,603 / 6,827 / 6,828	21.59 GB	178.44 MB

Baselines. We compare three representation paradigms. Flow-statistics methods include AppScanner (Taylor et al., 2017) and FlowPrint (Van Ede et al., 2020). Raw-bit/byte methods include TFE-GNN (Zhang et al., 2023) and nPrint (Holland et al., 2021). For nPrint, we reproduce its feature encoding and feed the resulting representations to an MLP classifier. Pre-trained methods include YaTC (Zhao et al., 2023) and NetMamba (Wang et al., 2024); we fine-tune their open-source pre-trained weights on the downstream tasks.

Classification metrics. We report Accuracy (ACC) and macro-averaged Precision (PRE), Recall (REC), and F1-score (F1). All reported classification metrics are computed on the test split.

E.2 Implementation Details

Data preparation. Each raw PCAP is partitioned into bidirectional flows. We standardize the packet-level input length through truncation or zero-padding and use a unified 8:1:1 training/validation/test split. To reduce shortcut learning from network configurations or temporal session states, we apply artifact masking across layers following the protocol associated with CipherSpectrum (Wickramasinghe et al., 2025).

Feature construction. We construct standardized bit-level structural views from Link (L2), Network (L3), and Transport (L4) protocol headers using nPrint. For strongly encrypted or anonymized traffic such as CipherSpectrum, the payload view uses compact behavioral statistics instead of raw payload bytes. The raw-view analysis in Fig. 2 motivates this choice: encrypted application content offers limited directly accessible plaintext semantics, leaving observable behavior as an important source of evidence. For non-strictly encrypted traffic, sparse payload-bit encodings can retain useful protocol-level evidence. The structural representations and redundancy-aware learning pipeline remain unchanged while the payload encoder adapts to the traffic setting.

Training configuration. We train PACC with AdamW. Across the four datasets, the learning rate is selected from 2.0×10^{-4} to 1.0×10^{-3}, with weight decay 1.0×10^{-4}. Batch sizes range from 16 to 64 according to dataset scale and memory footprint, and training lasts 30–160 epochs. The maximum numbers of structural chunks are capped at 32, 192, and 128 for the link, network, and transport layers, respectively. The hidden dimension is set to 96 or 128.

E.3 Efficiency Evaluation Protocol and Metrics

Inference profiling. Following prior traffic-model efficiency studies (Wang et al., 2024; 2026; Chehade et al., 2025), we profile prepared CipherSpectrum inference batches on an NVIDIA RTX PRO 6000 Blackwell Server Edition GPU. All methods use batch size 64, 80 warm-up iterations, and 600 timed repeats. Latency, throughput, parameter count, MACs, and peak GPU memory refer to the deployment inference graph that produces final prediction logits.

Inference profiling excludes data loading, disk I/O, CPU preprocessing, loss computation, back-propagation, optimizer updates, logging, and auxiliary modules used solely for training-time regularization. Pre-trained baselines are profiled after fine-tuning on the same strict split; their original pre-training cost is excluded.

Efficiency metrics. *Input/flow* denotes the logical input consumed by each model: sparse elements for PACC, pixels for packet-image baselines, and nodes/edges for TFE-GNN. *Params* counts trainable parameters in the deployment inference graph. *MACs/flow* measures per-flow multiply-accumulate operations, while *Peak GPU* records peak GPU memory during profiling. *Latency/flow* is the average logits-only forward time divided by batch size, and *Throughput* is the number of flows processed per second.

Table 6: Objective ablation across datasets. Values are macro F1-scores.

Variant	CipherSpectrum	UNSW-IoT	USTC-TFC	ISCX Tor/NonTor
w/o $\mathcal{L}^{\text{cls}}_{H,m}$	0.9425	0.9826	0.9892	0.9964
w/o $\mathcal{L}_{\text{rec},m}$	0.9568	0.9779	0.9887	0.9911
w/o $\mathcal{L}_{\text{red},m}$	0.9674	0.9849	0.9901	0.9998
w/o $\mathcal{L}_{\text{share}}$	0.9669	0.9750	0.9895	0.9988
w/o $\mathcal{L}_{\text{priv},m}$	0.9658	0.9837	0.9849	0.9986
PACC	**0.9685**	**0.9852**	**0.9906**	**1.0000**

Pipeline timing. We measure PACC from split CipherSpectrum flow PCAPs through preprocessing and training. Preprocessing all 41,000 flows with eight CPU workers takes 134.2 seconds, and training takes 743.0 seconds. YaTC, NetMamba, and TFE-GNN are trained on the same GPU with batch size 64 and learning rate 10^{-3}; their training or fine-tuning times through the selected best checkpoint are 1420.8, 1169.5, and 1991.7 seconds, respectively.

Baseline preprocessing and pre-training costs are unreported. Thus, the dash in Table 2 indicates missing prestage time, and the baseline entries should not be interpreted as complete pipeline totals.

F Supplementary Experiments

F.1 Additional Ablation Studies

Objective Ablation. We remove one training objective at a time while retaining the remaining architecture and other loss objectives. All variants follow the data splits and evaluation protocol of the main experiments. Table 6 reports macro F1-scores across the four datasets.

Removing $\mathcal{L}^{\text{cls}}_{H,m}$ noticeably degrades performance, indicating that the filtered holistic representation should remain task-discriminative. Removing $\mathcal{L}_{\text{rec},m}$ also causes clear degradation, showing the importance of preserving recoverable protocol structure during filtering. The effect of $\mathcal{L}_{\text{red},m}$ is milder, but its removal still slightly reduces performance, suggesting that redundancy suppression acts as a useful regularizer rather than the dominant training signal. Removing $\mathcal{L}_{\text{share}}$ weakens shared representation alignment, while removing $\mathcal{L}_{\text{priv},m}$ reduces the private representation's ability to retain residual task evidence. Together, these results support the complementary roles of intra-layer filtering, cross-layer consensus, and private residual preservation in PACC.

Representation and Fusion Ablation.

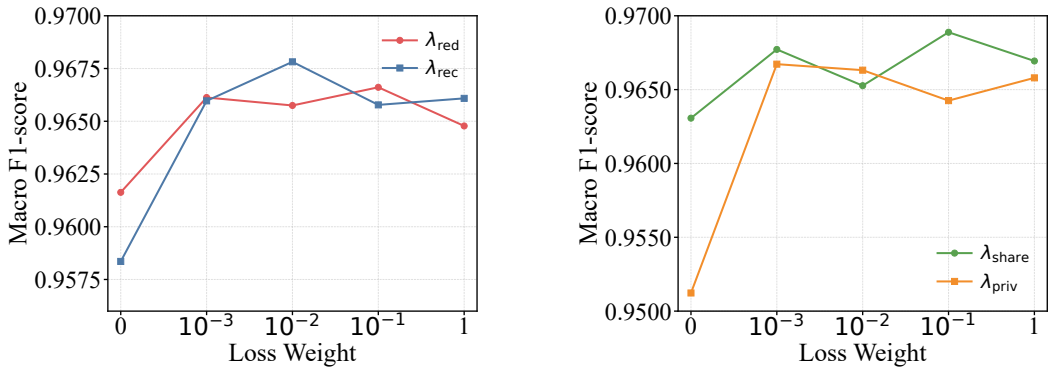
We compare behavioral-only, protocol-layer-only, and combined representations on CipherSpectrum to examine the contributions of input sources, learned representations, and fusion utility. The experiment uses 32,800 training and 4,100 test flows across 41 classes; the validation split is not used. Behavioral MLP and Behavioral LightGBM classify the same 256-dimensional behavioral representations input, while L2–L4 only concatenates the nPrint-based structural inputs from the Link, Network, and Transport layers without behavioral input. Raw features concatenation directly combines the Behavioral, Link, Network, and Transport inputs. Representations concatenation instead combines PACC's learned branch representations without adaptive fusion, corresponding to the w/o Fusion design in Table 3. PACC retains all learned representations and adaptive fusion.

Neural variants are trained for 30 epochs with AdamW, batch size 256, learning rate 10^{-3}, and weight decay 10^{-4}. Behavioral MLP, L2–L4 only, and Raw features concatenation are trained with classification cross-entropy. Representations concatenation and PACC use the full training objectives, they differ only in whether adaptive fusion is used. Behavioral LightGBM is selected by 40-trial randomized five-fold cross-validation on the training split. We run each variant with 10 random seeds, evaluate its final checkpoint once on the test split, and report the mean and sample standard deviation of ACC, PRE, REC, and F1 in Table 7.

The table shows that behavioral-only models outperform L2–L4 only, while combining all four raw inputs improves F1 further. Learned shared-private representations raise F1 from 0.9007 for Raw features concatenation to 0.9430 for Representations concatenation. Together, these results show

Table 7: Representation and fusion ablation on CipherSpectrum. Values are mean ± sample standard deviation over 10 seeds.

Variant	ACC	PRE	REC	F1
Behavioral MLP	0.8833 ± 0.0046	0.8890 ± 0.0021	0.8833 ± 0.0046	0.8832 ± 0.0042
Behavioral LightGBM	0.8818 ± 0.0035	0.8857 ± 0.0037	0.8818 ± 0.0035	0.8824 ± 0.0033
L2–L4 only	0.6997 ± 0.0604	0.7225 ± 0.0614	0.6997 ± 0.0604	0.6919 ± 0.0634
Raw features concatenation	0.9022 ± 0.0039	0.9203 ± 0.0036	0.9022 ± 0.0039	0.9007 ± 0.0052
Representations concatenation	0.9430 ± 0.0212	0.9441 ± 0.0225	0.9430 ± 0.0222	0.9430 ± 0.0215
PACC	$\mathbf{0.9571 \pm 0.0122}$	$\mathbf{0.9581 \pm 0.0125}$	$\mathbf{0.9571 \pm 0.0122}$	$\mathbf{0.9570 \pm 0.0122}$



(a) Filtering and reconstruction

(b) Shared and private objectives

Figure 6: Sensitivity of macro F1-score on CipherSpectrum to individual filtering, reconstruction, shared, and private objective weights.

the value of combining evidence across protocol layers. The further gain from learned representations is consistent with PACC's design: reducing intra-layer redundancy while retaining cross-layer consensus and layer-specific evidence. Adaptive fusion provides an additional improvement and interpretability.

F.2 Sensitivity to Individual Objectives

Starting from the configuration selected on the validation set, we vary one auxiliary loss weight at a time over $\{0, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$ while keeping all other weights fixed. This protocol isolates each objective's effect around the validated configuration rather than re-selecting all hyperparameters.

Fig. 6 reports the results. For structural filtering, Fig. 6a shows that both λ_{red} and λ_{rec} improve performance at moderate weights. Increasing the reconstruction weight from 0 to 10^{-2} is particularly beneficial, supporting preservation of recoverable ternary protocol structure; an excessive redundancy weight may instead suppress useful structural evidence.

For shared/private decomposition, Fig. 6b shows a clear drop when $\lambda_{\text{priv}} = 0$, consistent with the ablation results and supporting supervision of layer-specific residual evidence. Moderate shared alignment helps exploit cross-layer consensus, whereas an overly strong constraint may interfere with layer-specific cues. Overall, the auxiliary objectives are most useful as regularizers rather than dominating the main classification objective.

F.3 Generalization Experiments

Tasks and metrics. We examine three settings beyond the main evaluation: few-shot recognition of unseen applications on CipherSpectrum, future-day IoT device classification on UNSW-IoT, and encrypted application classification on a proprietary 50-class DoH dataset. We compare PACC with YaTC and NetMamba using Accuracy (ACC) and macro-F1.

Table 8: Few-shot classification on CipherSpectrum (%).

Shot	Method	ACC	Macro-F1
1	YaTC	50.92	48.15
	NetMamba	55.71	53.25
	PACC	**60.51**	**59.22**
5	YaTC	68.59	67.75
	NetMamba	72.05	70.54
	PACC	**79.68**	**79.28**
10	YaTC	72.84	72.34
	NetMamba	77.60	76.77
	PACC	**85.55**	**85.41**

Table 9: Temporal and 50-class DoH results (%).

UNSW-IoT: temporal generalization		
Method	ACC	Macro-F1
YaTC	77.26	88.50
NetMamba	80.91	85.45
PACC	**90.03**	**88.92**
50-class DoH classification		
Method	ACC	Macro-F1
YaTC	74.97	78.64
NetMamba	79.72	80.77
PACC	**83.17**	**84.52**

Few-shot setting. The 41 CipherSpectrum classes are partitioned into 26 base, five development, and ten novel classes. After training on the base classes, we freeze each encoder and L2-normalize its features. Each 5-way episode contains $K \in \{1, 5, 10\}$ support examples and 15 query examples per class. Support and query flows come from the original training and test partitions without overlap. All methods use the same 100 novel-class episodes, nested support sets, and query samples. A logistic-regression classifier is fitted to each support set; its regularization parameter is selected on 30 development episodes. Table 8 reports the mean over novel-class episodes.

Temporal setting. For UNSW-IoT, training and validation use captures from September 23–26, 2016, and testing uses captures from October 8, 2016. We apply the same bidirectional sessionization rules across dates, splitting TCP and UDP sessions after idle gaps exceeding 120 and 60 seconds, respectively. The resulting sets contain 38,730 training, 4,302 validation, and 8,374 test flows. The test set covers 13 of the 22 training classes; prediction remains over all 22 classes, while macro-F1 averages the 13 classes present in the test set. Results appear in Table 9.

Additional benchmark. We further evaluate 50-class encrypted application classification on a challenging DoH dataset containing strictly encrypted traffic. We use the same 8:1:1 training/validation/test split and training protocol as in the main experiments. Table 9 reports ACC and macro-F1.

PACC achieves the highest ACC and macro-F1 across all few-shot, temporal, and 50-class DoH evaluations. These results demonstrate strong generalization to unseen applications, future-day traffic, and a separate proprietary dataset.