FlowPicClip: Improving Network Traffic Classification Using Language Supervision

Daniel Shalev
Tel Aviv University
dans96@gmail.com

Tal Shapira
The Hebrew University
talshapirala@gmail.com

Yuval Shavitt
Tel Aviv University
shavitt@eng.tau.ac.il

***Abstract*—Traffic classification has gained much attention in the past decade, and deep learning proved to exhibit good classification performance. However, the lack of large labeled datasets pushed research to explore few-shots learning approaches, where only a few labeled samples per class are available. Augmentation techniques tailored to the domain of network traffic were proved as a viable solution.**

In this paper, we demonstrate a new approach to obtain better classification accuracy. Our solution simplifies preprocessing and reduces training time, while effectively utilizing small amounts of training data. Furthermore, it proves to be highly effective in few-shot scenarios, demonstrating robust results when tested on disjoint datasets, specifically the UCDAVIS19 and ISCX datasets.

Inspired by the recent breakthroughs in integrating image and text data, particularly the OpenAI CLIP model, we introduce FlowPicClip. This model harnesses the power of contrastive learning with FlowPics and their labels as text sentences. By leveraging Large Language Model (LLM) encoders, FlowPicClip aligns network traffic representations with their textual descriptions. We demonstrate 2.75% and 1.4% improvements over the best published results on the UCDavis19-Human and ISCX datasets for classification tasks, along with 1.5% and 6.2% improvements in few-shot classification achieved in a disjoint dataset scenario.

***Index Terms*—Traffic Classification, LLM, CLIP.**

I. Introduction

Accurately classifying network traffic, especially encrypted traffic, has become a major research focus in recent years. Traffic classification is critical for ensuring both Quality of Service (QoS) and security. It enables the prioritization of latency-sensitive applications like VoIP and video conferencing, and plays a crucial role in detecting anomalies and potential security threats by identifying unusual traffic patterns.

However, with the growing use of Internet traffic encryption and the widespread adoption of VPNs and Tor, traditional classification methods, such as Deep Packet Inspection (DPI), which rely on analyzing packet contents, have become less effective. As a result, newer approaches have shifted toward techniques that focus on extracting statistical features from network traffic flows [1], [2], [3]. These methods involve feature selection to discard irrelevant information, followed by the use of machine learning or deep learning models to classify the refined features.

An effective strategy for traffic classification involves transforming flow features into visual formats, such as 2D matrices or pseudo-images. These representations enable the application of deep learning models, particularly Convolutional Neural Networks (CNNs), which have been widely adopted from the field of image classification [4], [5], [6], [7], [8], [9], [10]. For our classification task we have chosen to use the FlowPic method, since it has been consistently demonstrated to achieve high classification accuracy [9], [10], [11], [12], [13], [14], and its straightforward implementation makes it an ideal choice for our purposes. Building on this foundation, Horowicz et al. [11], [12] introduced key improvements, including mini-FlowPics, a more compact and efficient variant of the original FlowPics.

Recently, a number of works have explored the use of data augmentation techniques to further improve traffic classification accuracy. Horowicz et al. [11], [12] introduced augmentations into the FlowPic framework. FlowPic had already shown high accuracy in classifying encrypted traffic in a supervised setting, but when there are only a few labeled sampled per class the addition of networking-inspired augmentations, such as changes in Round-Trip Time (RTT) and regular image augmentations, such as rotation, showed good performance. These augmentations were beneficial even for small labeled datasets requiring few training samples. Overall, [11], [12] approach is divided into two phases: first, they leverage augmentations to apply a self-supervised pre-training method, such as SimCLR. This allows them to obtain a pre-trained feature extraction network that is optimized to recognize similarities between similar FlowPics and differentiate between FlowPics with distinct labels. The second phase involves fine-tuning the pre-trained network from the first phase in a supervised manner, using only a small number of labeled examples. This approach, building upon the knowledge acquired during self-supervised pre-training, has demonstrated strong performance, even with limited training data, yielding promising classification results.

Following the introduction of the miniFlowPic approach [11], a subsequent study [13] was conducted to replicate and validate its results, specifically focusing on the UCDavis19 dataset. Unlike Horowicz el al. [11], which combined both script and human test samples in the UCDavis19 dataset, Finamore et al. [13] exclusively targeted the challenging human samples. To ensure fair comparison, we used the same training samples used in [13] in this work evaluation. Recently, Wang et al. [15] demonstrated that certain augmentations, such as sequence and masking, can significantly improve classification results across multiple datasets and are the better suited for traffic classification tasks.

979-8-3503-9281-4/25/$31.00 ©2025 IEEE

Building on these insights, our paper proposes a new approach that eliminates the need for augmentations entirely. Instead, we leverage language supervision by utilizing large language models (LLMs) text encoders to enhance traffic classification and enables network traffic and language to exist within the same latent space. Our model achieves impressive classification results on the UCDavis19 human test dataset, with an accuracy of 83.2%, reflecting a 2.75% improvement over the results reported by Finamore et al. [13]. Additionally, with only 10 samples, our model achieves 75.5% accuracy, matching the performance of the more complex unsupervised, augmentation-based training and tuning approach presented in [13]. Furthermore, we achieve a 1.5% improvement over [12] in the disjoint few-shot scenario on UCDavis19, along with a 6.2% improvement on the ISCX dataset, highlighting the strength of our approach.

II. Datasets

To compare our work with previous results established in [11], [13], we also utilize the UCDavis19 QUIC dataset as well as the ISCX dataset.

1) UC-Davis, QUIC Google Services: This dataset [3] comprises five distinct classes, each representing a different Google service: Google Drive, Google Music, Google Docs, Google Search, and YouTube. It includes a large number of labeled traffic flows in its training set. However, previous works, particularly [11], [12], [13], had demonstrated that it is possible to achieve strong classification performance by training on only 100 samples per class. This approach not only simplifies the training process but also ensures a more balanced distribution of labels, as opposed to the imbalanced distribution found in the full training dataset. The smaller, balanced training subsets are partitioned into five sets to conduct multiple experiments, with performance measured by the mean accuracy and confidence intervals across these partitions. Finamore et al. [13] extended their experiments beyond the few-shot learning setup by also training their models on the entire remaining training set. This approach allowed for a comprehensive evaluation, demonstrating how model performance scales with increased training data.

The UCDavis19 is comprise of two test datasets: one generated by automatic **scripts**, and thus easier to classify; and **human** generated database

2) ISCX, VoIP, and Video Application Identification: We utilize a combination of datasets from the University of New Brunswick (UNB): the 'ISCX VPN-nonVPN Traffic Dataset' (ISCX-VPN) [16] and the 'ISCX Tor-nonTor Dataset' (ISCX-Tor) [17]. These datasets contain packet capture (pcap) files that are labeled according to various encryption techniques (VPN, Tor, Regular), traffic types (e.g., VoIP, Video), and applications (e.g., WhatsApp, Facebook). To maintain consistency with previous research and enable direct comparisons, we follow the experimental settings established in previous studies [10], [12], [13]. During preprocessing, we filtered the ISCX dataset to include only sessions that contain at least 100 packets. In line with prior work, we constructed a dataset aimed at VoIP and Video Application Identification. This dataset includes 10 distinct classes that represent the use of VoIP applications (Facebook, Hangouts, Skype, Buster) and video applications (Facebook, Hangouts, Netflix, Skype, Vimeo, YouTube) under non-VPN encryption. For our experiments, we split the sessions into separate training and testing groups, ensuring there is no overlap between them. This methodology, consistent with the approach used for the UCDavis19 dataset, allows us to rigorously evaluate the performance of our models on both VoIP and video applications, providing a comprehensive understanding of their capabilities in handling encrypted traffic.

III. Methods

A. Leveraging Pre-trained Models for Image-Text Alignment

The CLIP model [18] has spurred extensive research in aligning image and text representations, leading to notable progress in multimodal learning. Building on this, Locked-image Tuning (LiT) [19] achieved strong zero-shot transfer performance by freezing a pre-trained image encoder and tuning the text encoder through contrastive learning, thereby utilizing robust visual features. Inspired by this, our approach takes the reverse path: we freeze a powerful text encoder and train a Convolutional Neural Network (CNN) to align its image representations with the text outputs. This design leverages the contextual strength of modern Large Language Models to supervise image classification without the need for data augmentations, reducing computational cost and improving accuracy in encrypted traffic scenarios.

B. Language Supervision Training

This section outlines the approach and settings for contrastive learning with text supervision. The primary goal is to evaluate how many training samples are required to achieve strong results compared to previous studies, while building a robust FlowPic feature extraction backbone, which will later be tested in a few-shot scenario across different datasets.

*1) **Model Overview**:* The model architecture we propose consists of two distinct components: a text encoder and an image encoder, each selected and designed with specific motivations and requirements in mind. The overall structure of this model is illustrated in Figure 1.

2) Image Encoder: The image encoder employed in our model is a Convolutional Neural Network (CNN) based on the LeNet5 [20] architecture. This choice was made for two primary reasons: (1) Consistency with Previous Work [11], [13], [12], and (2) Suitability for Sparse FlowPics: As discussed by Horowicz et al. [11].

3) Text Encoder: We used two text encoders for training and testing our model. The first encoder, "clip-vit-base-patch32" by OpenAI [18] with pretrained weights from CLIP. The variant of our model that integrates this encoder is referred to as "FlowPicClip-CLIP". We selected a more recent and efficient text encoder—OpenAI's "text-embedding-3-large". This model offers strong performance making it a compelling choice for modern NLP tasks. The variant of our model using

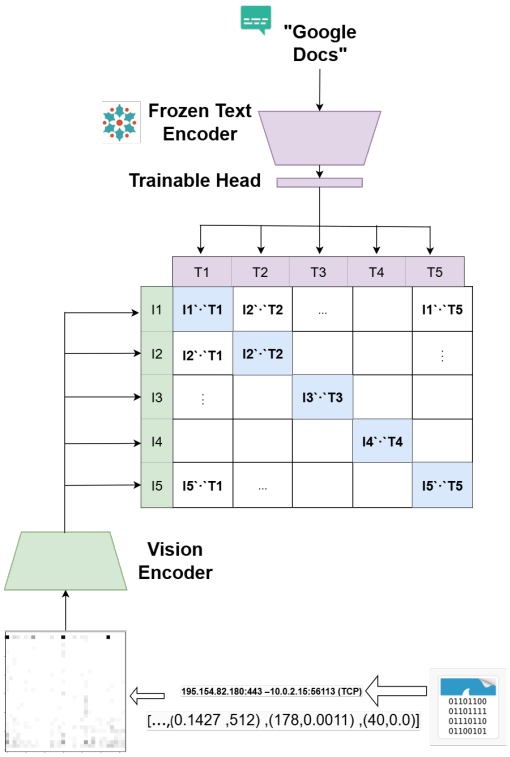

Figure 1: FlowPicClip model overview

this encoder is called "FlowPicClip-Large". Its efficiency and ease of use, combined with robust embedding capabilities, provide a valuable contrast to the CLIP-based encoder.

Additionally, using these models is straightforward once the necessary API access is obtained. This allows us to precompute and store the text label embeddings offline, significantly simplifying the training process. Once the embeddings are obtained, they are passed through a pair of linear layers to align them with the image embeddings. Specifically:

- The first linear layer reduces the dimensionality from the original text encoder embeddings features (3072 for text-embedding-3-large and 512 for clip-vit-base-patch32) down to 360 features.
- The second linear layer maps the 360 features to 120 features, matching the output size of the image encoder.

Using a frozen text encoder and precomputed embeddings offers several advantages: **Faster Training:** since the text embeddings are computed offline, the only overhead during training is the quick pass through the linear layers, making the process efficient. **Simplified Workflow:** By handling the language supervision component offline, the overall training procedure is streamlined, reducing complexity and computational load.

The final image and text embeddings are then fed into a contrastive loss function to effectively align the visual and textual representations.

*4) **Approach**:* Our training approach consists of several key aspects:

Text Label Selection: In our experiments, we explored strategies for text label selection to evaluate their impact on model performance. The first approach used only the application name as the text label, e.g., in the UC Davis dataset, the label for a YouTube session was simply "YouTube." This was the only feasible approach for the UC Davis dataset due to the limited available traffic-session information.

In contrast, the ISCX dataset provided additional details such as the protocol (TCP or UDP) and category (video or VoIP). This facilitated more descriptive text label options, such as: "An internet session of using *application* for *category* over *protocol*." This approach aimed to provide richer context to the FlowPic, potentially improving clustering by adding meaningful information, but may potential increase noise during training.

Batch Composition Constraints: Due to the contrastive loss metric, it is imperative that each batch contains distinct examples. This ensures that related texts and images achieve high similarity scores and are positioned close in the latent space, while non-related texts and images are kept far apart. Therefore, each batch must include only one example per class from the dataset. This constraint results in smaller batch sizes, e.g., in the UCDavis19 dataset, the batch size is limited to 5 due to the presence of only 5 distinct classes.

Batch Combinations and Sampling Strategy: Let C represent the total number of classes ($C = 5$ for UCDavis19), and let X denote the total number of samples per class in the dataset. Each batch includes exactly one sample from each class, thus the maximum number of unique batches is X^C. E.g., if we assume $X = 100$ samples per class in UCDavis19, and $c = 5$ the total number of possible distinct batch combinations would be $100^5 = 10,000,000,000$.

In practice, we do not sample all possible combinations before updating the model. Instead, we choose B batch combinations to sample before performing a backward update. Increasing B introduces more diversity in the sampled batches but reduces the frequency of updates, while a lower B have the opposite effect. We aim to find an optimal trade-off between training diversity and convergence speed.

IV. Experiments and Results

A. Metrics Evaluations

We evaluate the model using three key metrics: training loss, validation accuracy, and test accuracy. The training loss is assessed using the contrastive loss function. During validation and testing, accuracy is measured by the cosine similarity between the image embeddings and all text label embeddings generated by the pretrained model. The testing accuracy is computed by first extracting a row from the similarity matrix S, where each element $S[i, j]$ represents the similarity between the i-th image and the j-th text label. Then, a softmax function is applied to the extracted row to convert the similarities into probabilities. The predicted label $\hat{y}$ is chosen as the text label with the highest probability, as described in equation 1:

$$\hat{y} = \arg\max_j \left(\text{softmax}(S[i, :])_j\right) \tag{1}$$

where:

- $S[i,:]$ is the i-th row of the similarity matrix corresponding to the i-th image,
- $\text{softmax}(S[i,:])_j$ is the probability assigned to the j-th text label after applying the softmax function to the similarity scores,
- $\hat{y}$ is the predicted text label for the i-th image, determined by selecting the label j with the highest probability.

The overall accuracy is computed as the ratio of correct predictions to the total number of samples. Unlike the training phase, where the batch size imposes constraints, this evaluation is performed without such limitations.

B. Activation Function

We explored two activation functions for the image encoder in our LeNet5 CNN architecture. Following the recommendations in previous works [11], [13] we used ReLU, and implemented Kaiming initialization as suggested in [13] to enhance the network's performance. We term this "FlowPicClip-ReLU". Additionally, we experimented with the Tanh activation function, which we term "FlowPicClip-tanh".

For the training phase of the language supervision approach, we fine-tuned the hyperparameters separately for the UCDavis19 and ISCX datasets to optimize performance. The FlowPicClip model was trained for 16 epochs with a batch size of 5 (as discussed in III-B4).

C. Language Supervised Results

Each type of model was run with five different random seeds, both during the language supervision training phase and the disjoint dataset few-shot training. All the presented results are the mean accuracies across these runs.

We begin by examining the results on the most challenging dataset: the human test dataset from the *UCDavis19* dataset. By analyzing the performance of our models on this dataset, we select the most promising model configuration to apply to the remaining datasets for further training and testing.

Figure 2 presents the accuracy of our models *flowpicClip-clip* (referred here as "clip" and uses tanh activation), *flowpicClip-tanh* ("large-tanh"), and *flowpicClip-relu* ("large-relu"). Throughout the remainder of this section, 'large' refers to the usage of the "text-embedding-3-large" text encoder described in III-B3). The number of batch combinations is coded by color for $B = 2, 20, 50$.

Figure 2 clearly shows that as the number of samples per class increases, the models consistently achieve higher accuracy. At very small sample sizes (e.g., 10 samples per class), the performance dependency on the number of batch combinations is quite large for the *large-tanh* and *clip* models. This is expected, as exposing the model to a larger variety of sample combinations compensates for the lack of training examples, enhancing the diversity of training data. However, as the number of samples per class increases (e.g., 70 and 100), the performance dependence on batch combination diminishes.

The *large-tanh* model consistently demonstrates strong accuracy and stability across various batch combinations and

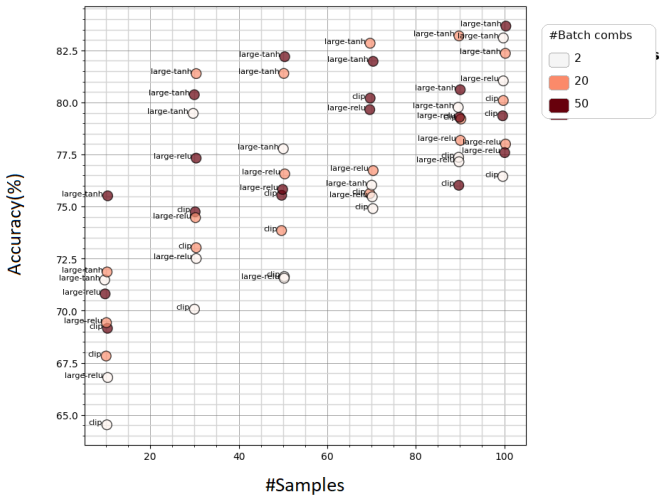

Figure 2: UCDavis19 Human test dataset results

sample sizes, making it the best-performing candidate compared to the *clip* and *large-relu* models.

Since the *large-tanh* model shows the best performance, we made additional experiments (graphs are omitted). When the number of samples per class is large, #BC has little effect on accuracy. However, for 10 samples per class, increasing the #BC improves accuracy and reduces the confidence intervals. Fig. 3 shows that most of the gain in accuracy is achieved for 30 samples, and as we increase the number of samples beyond 30 the confidence intervalk decreases.

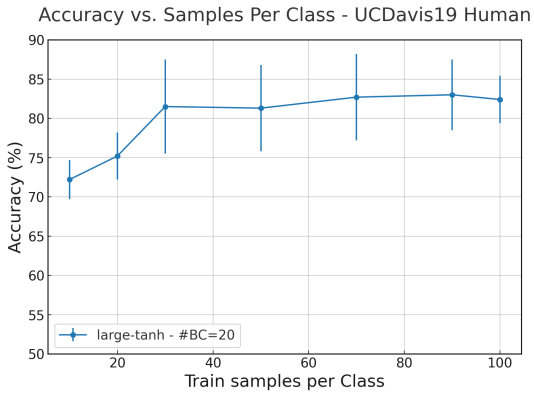

Figure 3: Accuracy vs. Samples Per Class for large-tanh variant with #BC=20 on UCDavis19-Human dataset.

(2) UCDavis19 Script results As was already shown in previous studies [11], [12], [13] this dataset is not challenging, and we achieved significantly higher accuracy than for the Human dataset (Table omitted). The best results are generally achieved with #BC = 50 across all models, with very minor differences in variance.

Based on the script and human results, we now focus on the *large-tanh* model, referring to this configuration as "FlowPicClip" for the remainder of this paper.

(3) UCDavis19 Comparison With Previous Work Using the best configuration of our model, we compare our results with previous papers [11], [12], [13] that evaluated the classification performance on the UCDavis dataset (both human and script) using several distinct approaches.

Finamore et al. [13] used as baseline for comparison the XGBoost machine learning technique with FlowPic images as input, performing supervised training and testing on the dataset without applying any augmentations.

All the three papers [11], [12], [13] used the LeNet5 CNN architecture. They [11], [12], [13] reported the results of the supervised method both with and without augmentations. In the augmented version, the training dataset was significantly increased by applying various augmentation techniques. Specifically, Finamore et al. [13] used 100 samples per class for each partition (identical to the partitions we used) and expanded the training set tenfold using these augmentations, aiming to enhance the model's performance.

Finally, Finamore et al. [13] replicated the unsupervised learning approach introduced by [11], [12] using the SimCLR model with augmentations. After completing the unsupervised training, Finamore et al. [13] fine-tuned the model with only a few labeled samples to evaluate whether this method could outperform their previous supervised approaches. Specifically, [11], [12], [13] presented results of fine-tuning the pre-trained model in a supervised setting, using 10 samples per class.

Table I: Comparison of Accuracy on UCDavis19 Dataset

Method	Script (%)	Human (%)
FlowpicClip-10 (Ours)	93.1 ± 1.2	75.5 ± 2.5
FlowpicClip-30 (Ours)	97.1 ± 0.5	80.7 ± 2.2
FlowpicClip-100 (Ours)	**98.5 ± 0.6**	**83.2 ± 3.7**
Supervised (from [13])		
XGBoost	96.37 ± 0.31	**73.65 ± 2.14**
CNN	95.64 ± 0.37	68.84 ± 1.45
CNN w/ Augmentations:		
Rotate	96.31 ± 0.44	71.65 ± 1.98
Change RTT	**97.29 ± 0.35**	70.76 ± 1.99
simclr & fine tune (from [13])		
No aug.	92.18 ± 0.31	74.69 ± 1.13
w/ Augmentations:		
Change RTT, Time Shift*	92.18 ± 0.31	74.69 ± 1.13
Packet Loss, Color Jitter	90.17 ± 0.41	73.67 ± 1.24
Change RTT, Color Jitter*	91.72 ± 0.36	75.56 ± 1.23
Change RTT, Rotate	92.38 ± 0.32	74.33 ± 1.26
"leftover" pretrain, simclr+fine tune	**93.90 ± 0.74**	**80.45 ± 2.37**

The results presented in this table are aggregated from 5 different seeds across 5 distinct data splits/partitions, following the same methodology as in [13], resulting in a total of 25 experiments for each set of results presented here.

Table I provides a comprehensive comparison between our language-supervised FlowPicClip results and the various approaches used by [13], specifically on the UCDavis19 dataset.

We show that the FlowPicClip model trained with 100 samples per class ("FlowPicClip-100") achieves the highest accuracy among all the experiments reported in [13], outperforms the best results from [13] by nearly 3%.

Our results are particularly impressive given that we achieve similarly high accuracy, nearly matching the top score from [13], using only 30 samples per class with language supervision training ("FlowPicClip-30" in Table I). Furthermore, our results for language supervision with just 10 samples per class, show that even with this limited data, the accuracy remains the second highest among all the "simclr + fine tune" results in [13]. The only method that surpasses FlowPicClip-10 is their "leftover" experiment, where Finamore et al. [13] used nearly the entire training dataset in a contrastive learning approach with augmentations. While Finamore et al. [13] method yields slightly better performance, it requires extensive data augmentation and a more complex training process. In contrast, our approach with just 10 samples per class, without any augmentations, offers a simpler and faster training setup.

D. Few-Shot Learning Results

After pretraining our model using language supervision, we fine-tuned the FlowPicClip model using only a few samples from a disjoint dataset. Specifically, we fine-tuned the UCDavis-pretrained model with limited examples from the ISCX dataset, and vice versa, as reported below. Our fine-tuning process involves using both pre-trained image and text encoders and training them on a new, unseen dataset (new flowpics and new text labels). Testing our model on a disjoint dataset allows us to evaluate its generalization ability, as it is exposed to a completely different distribution of text labels from those seen during training. We then compared our few-shot results with previous results [11], [12].

Table II: Comparison of Accuracies on Disjoint ISCX Test Dataset

Method	#Shots	Accuracy (%)
flowpicClip (Ours)	5	79.23 ± 1.66
flowpicClip (Ours)	7	82.6 ± 2.5
flowpicClip (Ours)	10	**86.05 ± 0.52**
Horowicz et al. [12]	10	79.83

The results are obtained by averaging over 5 different seeds, fine-tuning the FlowPicClip models. The text encoder used, "text-embedding-3-large", remains frozen during all experiments.

Table II shows that our FlowPicClip model with 10-shots achieves the highest accuracy at 86.05%, significantly outperforming the baseline method [12], which reported 79.83% accuracy. FlowPicClip with 7-shots also achieves strong performance, recording an accuracy of 82.6%, while the 5-shot model delivers competitive results at 79.23%.

Figure 4 shows performance of FlowPicClip pre-trained on UCDavis19 and tested on the disjoint ISCX dataset. Accuracy steadily improves from 56% with 1 sample to 86.05% with 10 samples. The accuracy on the Disjoint UCDavis19 dataset for the human-based data reaches 72.6% with 7-shots and does not improve for 9 and 10 shots. For UCDavis19-script accuracy reaches 93.2% for 10-shots (figure omitted).

V. Implications and Future Work

By incorporating multimodality into the domain of network traffic analysis, we unlock a wide range of applications and

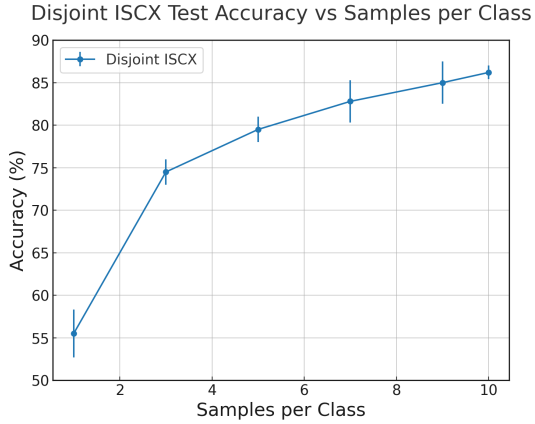

Figure 4: Video and VoIP applications - Accuracy as a function of the training set size

avenues for future improvement. One promising direction is to explore zero-shot learning on different datasets, beyond the few-shot scenarios we've tested. Our initial experiments with zero-shot learning on the UCDavis and ISCX datasets yielded poor results (not detailed here), since zero-shot learning excels when rich, descriptive text labels are available, acting as valuable metadata. However, the UCDavis and ISCX datasets, aside from YouTube, contain distinct and non-overlapping applications, with UCDavis focusing on QUIC and ISCX on TCP/UDP traffic. We believe that leveraging multiple datasets with common descriptors, such as network protocols or session types (e.g., Tor, VPN), could yield more successful results.

Aligning text with network traffic opens up powerful new capabilities. For instance, we successfully implemented (not reported here) a text-based search engine for network traffic. By storing FlowPic embeddings in a vector database and searching via cosine similarity, users can query network traffic with natural language. For example, a query like "users listening to Taylor Swift" could return the most relevant embeddings from YouTube and Google Music categories. This approach could be extended to implement Visual Question Answering (VQA) systems for network traffic.

VI. Conclusions

We presented FlowPicClip, an approach to network traffic classification that incorporates language supervision through LLMs text encoders. By aligning FlowPics with their textual descriptions, the model improves classification performance. Experiments on the UCDavis19 and ISCX datasets show competitive or superior performance compared to augmentation-based methods. By eliminating the need for traditional data augmentation techniques, our method also simplifies preprocessing steps and reduces computational demands.

Ethical Considerations: The databases used in this paper have been used by many previous papers and do not raise ethical concerns.

References

[1] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, "Traffic classification through simple statistical fingerprinting," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 1, pp. 5–16, 2007.

[2] A. Fahad, Z. Tari, I. Khalil, I. Habib, and H. Alnuweiri, "Toward an efficient and scalable feature selection approach for internet traffic classification," *Computer Networks*, vol. 57, no. 9, pp. 2040 – 2057, 2013.

[3] S. Rezaei and X. Liu, "How to achieve high classification accuracy with just a few labels: A semi-supervised approach using sampled packets," in *Industrial Conference on Data Mining (ICDM)*, 2019.

[4] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges," *IEEE Transactions on Network and Service Management*, vol. 16, Jun. 2019.

[5] H. Huang, H. Deng, J. Chen, L. Han, and W. Wang, "Automatic multi-task learning system for abnormal network traffic detection." *International Journal of Emerging Technologies in Learning*, vol. 13, no. 4, 2018.

[6] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access*, vol. 5, pp. 18 042–18 050, 2017.

[7] S. Rezaei and X. Liu, "Multitask learning for network traffic classification," in *2020 29th International Conference on Computer Communications and Networks (ICCCN)*. IEEE, 2020, pp. 1–9.

[8] S. Roy, T. Shapira, and Y. Shavitt, "Fast and lean encrypted internet traffic classification," *Computer Communications*, vol. 186, pp. 166–173, 2022.

[9] T. Shapira and Y. Shavitt, "FlowPic: Encrypted internet traffic classification is as easy as image recognition," in *IEEE Workshop on Network Intelligence: Machine Learning for Networking (NI)*, 2019, pp. 680–687.

[10] ——, "FlowPic: A generic representation for encrypted traffic classification and applications identification," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1218 – 1232, 2021.

[11] E. Horowicz, T. Shapira, and Y. Shavitt, "A few shots traffic classification with mini-flowpic augmentations," in *The 22nd ACM Internet Measurement Conference (IMC)*, 2022, pp. 647–654.

[12] ——, "Self-supervised traffic classification: Flow embedding and few-shot solutions," *IEEE Transactions on Network and Service Management*, vol. 21, pp. 3054 – 3067, 2024.

[13] A. Finamore, C. Wang, J. Krolikowski, J. M. Navarro, F. Chen, and D. Rossi, "Replication: Contrastive learning and data augmentation in traffic classification using a flowpic input representation," in *Proceedings of the 2023 ACM on Internet Measurement Conference*, 2023, pp. 36–51.

[14] L. Yang, A. Finamore, F. Jun, and D. Rossi, "Deep learning and zero-day traffic classification: Lessons learned from a commercial-grade dataset," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4103–4118, 2021.

[15] C. Wang, A. Finamore, P. Michiardi, M. Gallo, and D. Rossi, "Data augmentation for traffic classification," in *Passive and Active Network Measurement (PAM)*, 2024, pp. 159–186.

[16] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related features," in *Proceedings of the 2nd International Conference on Information Systems Security and Privacy - Volume 1: ICISSP*, INSTICC. SciTePress, 2016, pp. 407–414.

[17] A. H. Lashkari, G. D. Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of tor traffic using time based features," in *Proceedings of the 3rd International Conference on Information Systems Security and Privacy - Volume 1: ICISSP*, INSTICC. SciTePress, 2017, pp. 253–262.

[18] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[19] X. Zhai, X. Wang, B. Mustafa, A. Steiner, D. Keysers, A. Kolesnikov, and L. Beyer, "Lit: Zero-shot transfer with locked-image text tuning," in *The IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 18 123–18 133.

[20] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.