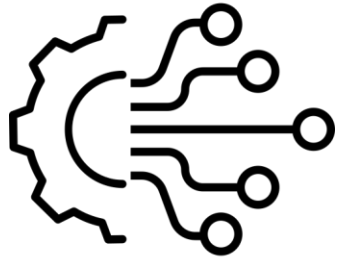
Revisiting Concept Drift in Windows Malware Detection: Adaptation to Real Drifted Malware with Minimal Samples

Adrian Shuai Li, Arun Iyengar, Ashish Kundu, Elisa Bertino

Classic ML Pipeline

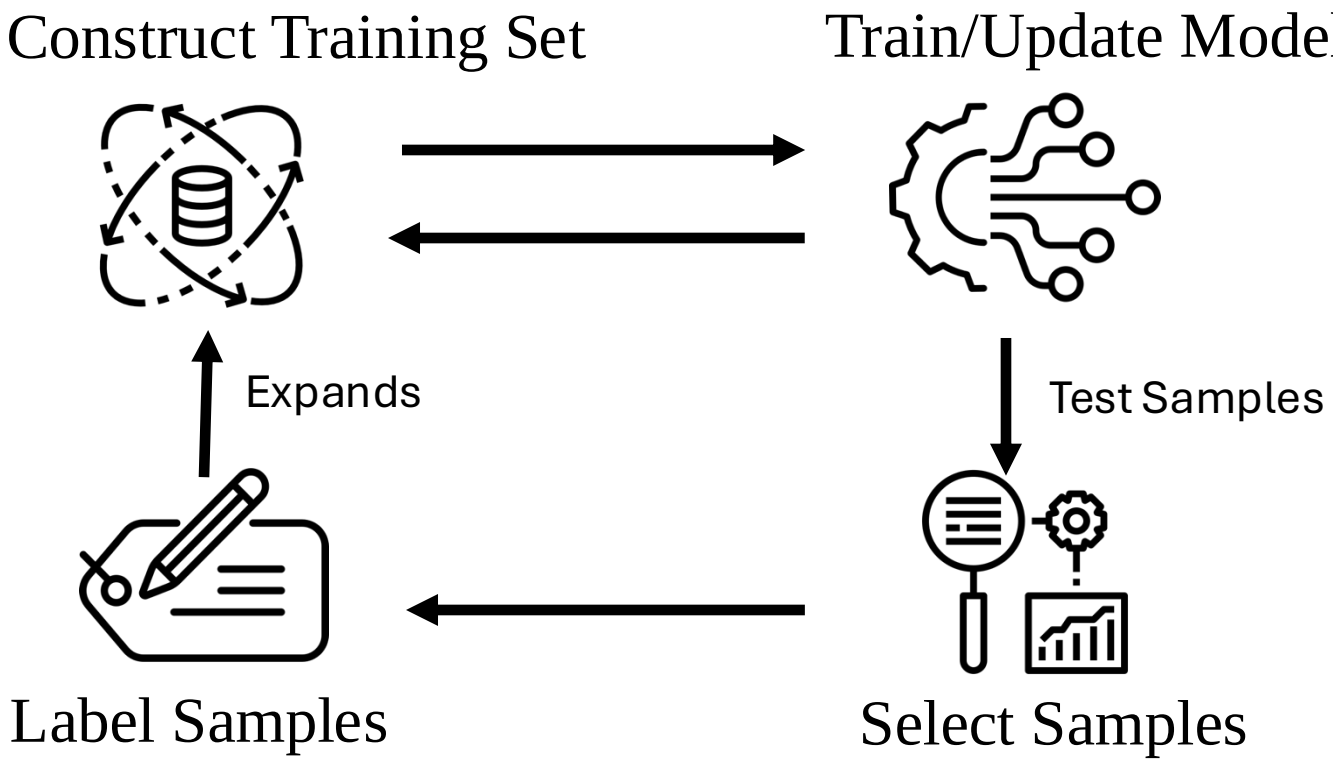
Construct Training Set

Train/Update Models



Expanding the training set is time-consuming, costly, and sometimes impractical due to analysts' limited capacity for daily malware sample labeling

Active Learning Pipeline



Selection methods: uncertainty sampling [1], utilizing rejection-thresholds [2], using contrastive learning [3]

[1] Y. Chen, Z. Ding, and D. Wagner, Continuous learning for android malware detection. USENIX Security 23.
[2] F. Barbero, F. Pendlebury, F. Pierazzi, and L. Cavallaro, Transcending transcend: Revisiting malware classification in the presence of concept drift. S & P. 2022.
[3] L. Yang et al. CADE: Detecting and explaining concept drift samples for security applications. USENIX Security 21.

Train/Update Models

- Past approaches have predominantly used basic retraining techniques for model update [1]
- **Cold-start learning,** involves training a fresh model each time new labels are introduced
- **Warm-start learning,** continue training an existing model with new samples
- Our study indicates that neither strategy yields optimal performance when only a few new samples are available
- Our Goal:

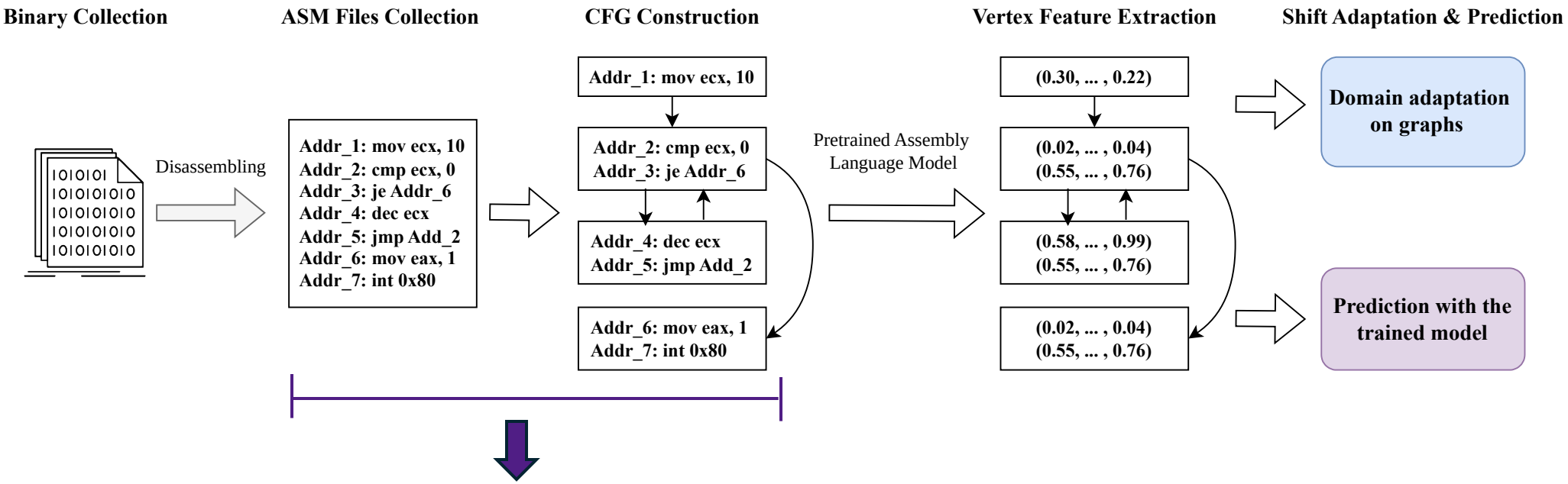
Propose a new solution for model retraining, that surpasses the above two model update strategies, when only a small number of new labels are available

- Core idea

An adaptive model should acquire knowledge of common characteristics shared by a broad range of malware

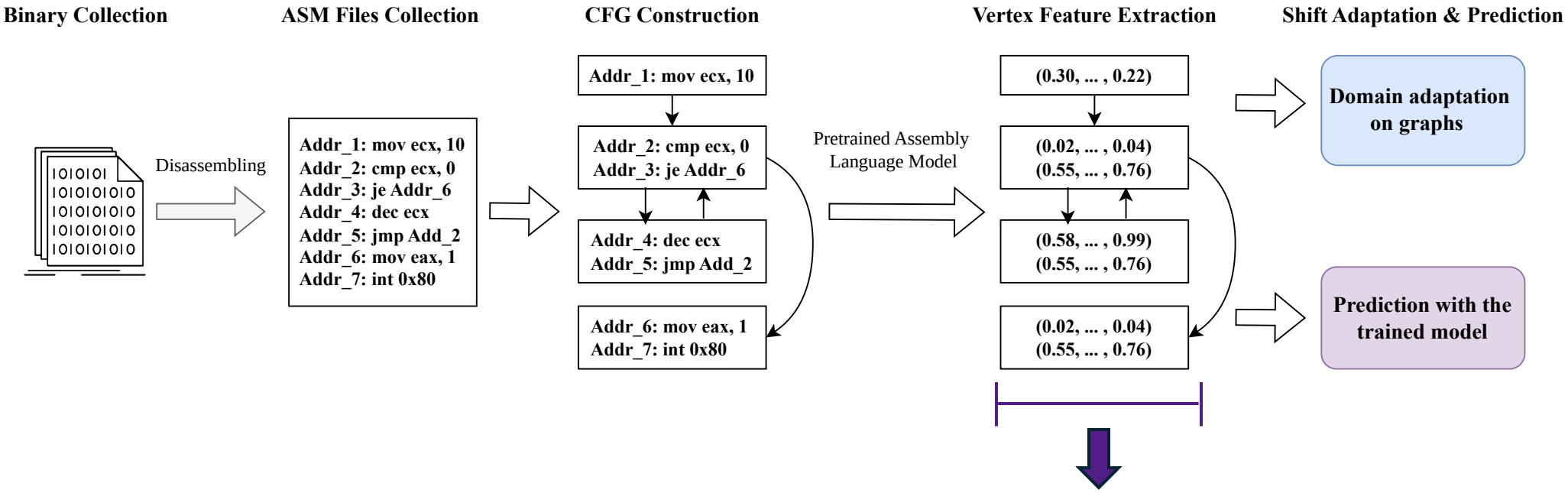
[1] Y. Chen, Z. Ding, and D. Wagner, Continuous learning for android malware detection. USENIX Security 23.

Framework Overview



Disassemble the malware binary to extract the CFG from the assembly code

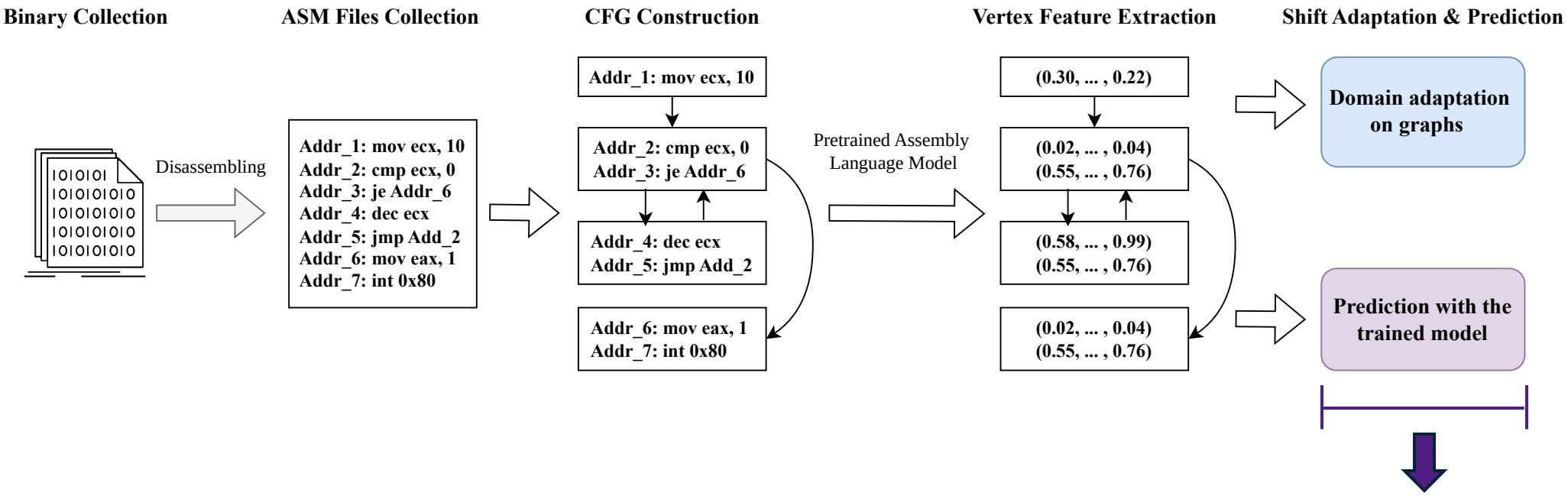
Framework Overview



Convert the raw instructions in each basic block of the CFG into feature vectors
Using a pre-trained assembly language model **PalmTree**[4] to generate instruction embeddings

[4] X. Li, Y. Qu, and H. Yin, Palmtree: Learning an assembly language model for instruction embedding. CCS 2021

Framework Overview



The drift adaptation model directly learns from existing and drifted malware CFGs

Shift Adaptation: Model Design

Goal: Learn an intermediate representation containing information that remains consistent before and after the drift while still being sufficient to make a good classification

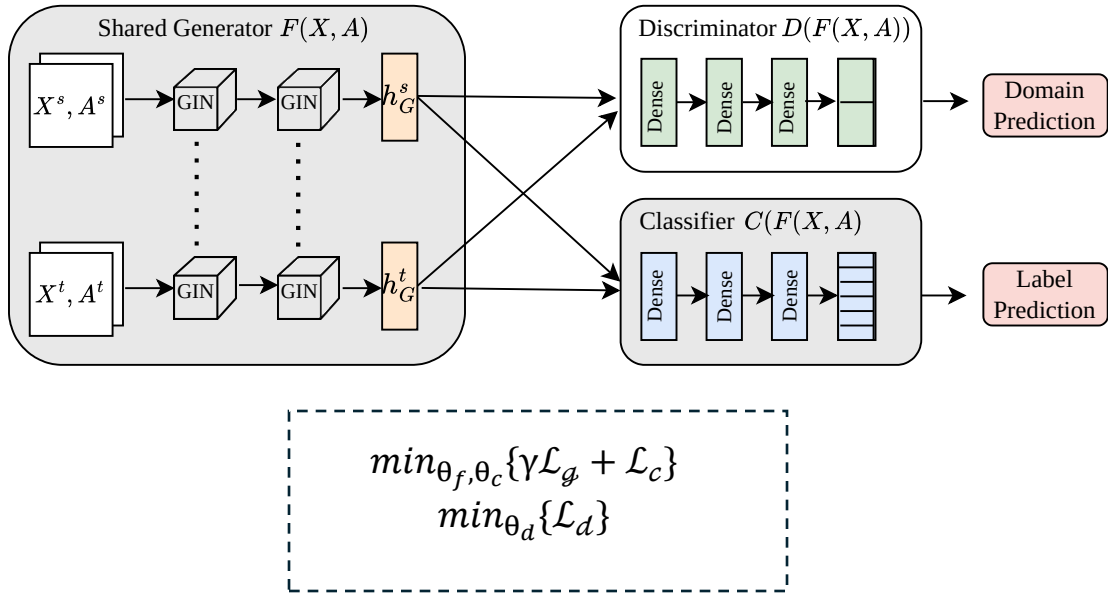
- Propose a graph-based **domain adaptation (DA)** method to address malware drift
- **Label Prediction (LP) task**, based on the Graph Isomorphism Network (GIN)

$$\mathcal{L}_c = -\sum_{i=1}^{N_s} Y_i^s \cdot log\widehat{Y_i^s} - \lambda \sum_{i=1}^{N_t} Y_i^t \cdot log\widehat{Y_i^t}$$

- **Adversarial Training (AT) task**, involving training of two networks through minimax optimization to predict the input domain (pre-drift or post-drift)

$$\mathcal{L}_g = -\sum_{i=1}^{N_s+N_t} [(1-d_i)log\hat{d}_i + d_i log(1-\hat{d}_i)]$$

$$\mathcal{L}_d = -\sum_{i=1}^{N_s+N_t} [d_i log\hat{d}_i + (1-d_i)log(1-\hat{d}_i)]$$



Generating Drifted Malware Clusters

Motivations: Leave-one-out evaluation of malware detectors can overestimate accuracy by not verifying actual concept drift. Success with related families can mask a lack of true drift adaptation

Using Graph Auto-Encoder + Consensus Clustering to generate statistically distinct malware clusters

- Using the GAE to learn graph representations that could reconstruct the input CFG
- Using a weighted consensus clustering approach that leverages multiple clustering algorithms and assigns more weight to predictors that yield better clustering results

Evaluation on Research Dataset

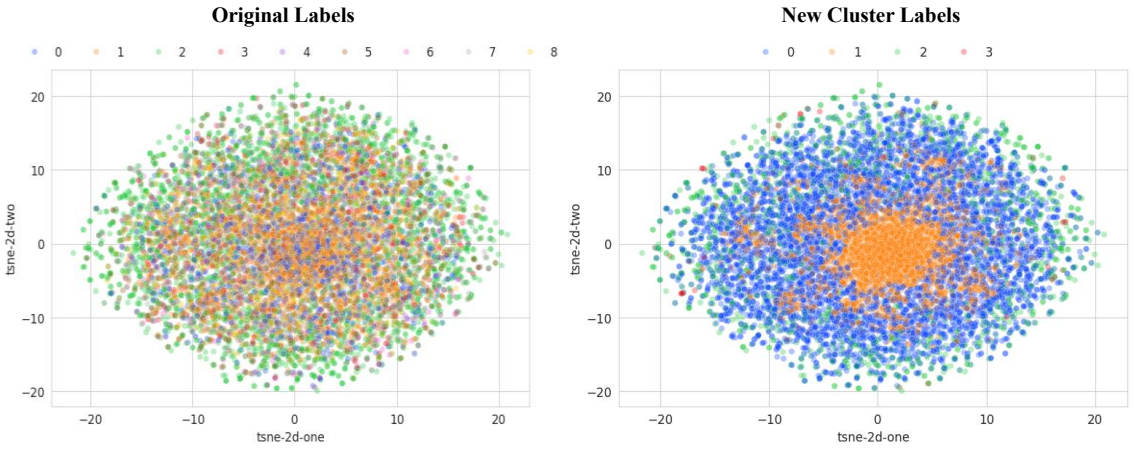


Fig 4. Visualization of the graph feature vector with their labels. The left part of the figure shows the data with the original labels from Big 15[5], and the right one shows the newly learned clusters. The legend represents the mapping between labels and colors.

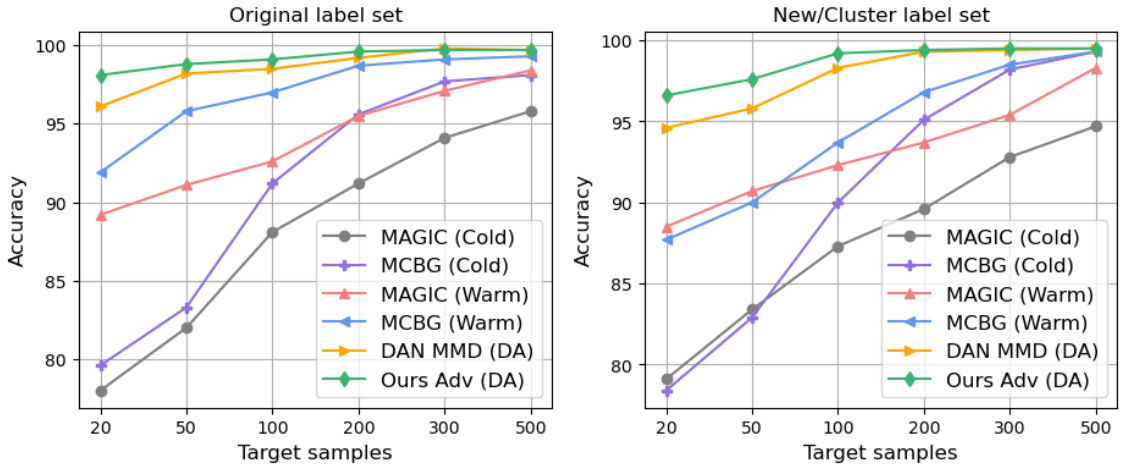


Fig 3. Given a set of fixed target training labels, we compute the accuracy of the target testing data for different baseline techniques and our method. The left diagram reports the averaged accuracy based on the original label set of Big-15, and the right one reports results based on the cluster label assignment.

- DA methods yield the highest performance, with warm-start learning trailing behind the DA methods and cold-start learning producing the lowest outcomes
- Our method experienced the smallest accuracy drop of 0.5% on average

[5] R. Ronen, M. Radu, C. Feuerstein, E. Yom-Tov, and M. Ah- madi, Microsoft malware classification challenge, arXiv preprint.

Impact of Malware Representations

TABLE II. AVERAGED ACCURACY OF BASELINES WITH VARIOUS MALWARE REPRESENTATIONS. FOR THE CONTENT AND IMAGE REPRESENTATION, WE REPORT THE PERCENTAGE CHANGE IN ACCURACY FROM OUR ADVERSARIAL (ADV) DA METHOD TO THE PEAK PERFORMANCE AMONG ALL BASELINES WITHIN THE SAME REPRESENTATION. ULTIMATELY, WE DEMONSTRATE THE IMPROVEMENT OF OUR FULL PIPELINE (GRAPH + ADV DA) COMPARED TO BOTH CONTENT + ADV DA AND IMAGE + ADV DA.

Malware Representation	Strategy	Method	Target samples					
			20	50	100	200	300	500
Content-based (CB)	Cold	SVM	67.2	71.4	75.4	79.3	83	85.6
		MLP	75.9	79.4	85.3	91.6	92.9	94.9
	Warm	SVM	70.2	74	79.1	83.2	86.7	90.5
		MLP	**91**	93.4	95.4	96.2	**97.4**	**97.9**
	DA	DAN (MMD) + MLP	90.8	**93.9**	**95.4**	**96.9**	97.1	97.5
		Ours (Adv) + MLP	**94.1** ↑3.1	**95.2** ↑1.3	**96.2** ↑0.8	**97.1** ↑0.2	**97.7** ↑0.3	**97.8** ↓0.1
Image-based (IB)	Cold	ResNet-50	60.6	60.6	70.6	74.3	77	84.4
	Warm	ResNet-50	**86.2**	**87.7**	89.5	**91.9**	**93.5**	**94.4**
	DA	DAN (MMD) + CNN	85.4	87.5	**89.9**	91.6	93.1	94
		Ours (Adv) + CNN	**88.6** ↑2.4	**90** ↑2.3	**92.2** ↑2.3	**93.1** ↑1.2	**94** ↑0.5	**94.6** ↑0.2
Graph-based (GB)	DA	Ours (Adv) + GIN	**96.6**	**97.6**	**99.2**	**99.4**	**99.5**	**99.5**
Improvement over CB: Ours (Adv)			↑2.5	↑2.4	↑3	↑2.3	↑1.8	↑1.7
Improvement over IB: Ours (Adv)			↑8	↑7.6	↑7	↑6.3	↑5.5	↑4.9

- Our shift adaptation component consistently has the highest accuracy across all feature representations
- Combining the adaptation approach with our graph representations achieve the best result

Real World Malware Dataset

TABLE III. SUMMARY OF THE MB-24 DATASET.

Summary	Mar	April	May	July	Aug	Sep
# Samples	1505	1080	1496	1618	1613	1337
# Malware Families	104	81	99	126	111	92

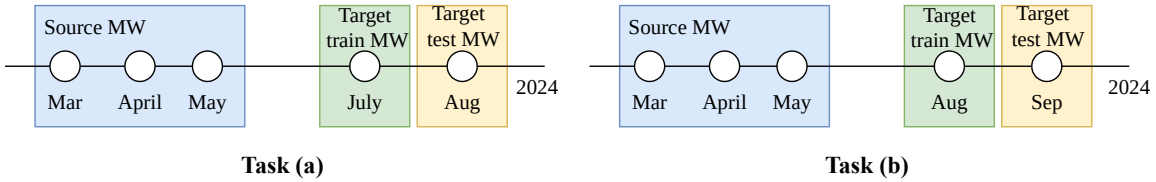


Fig 6. Source and target malware datasets setup for the monthly model update in July 2024 (Task (a)) and August 2024 (Task (b)).

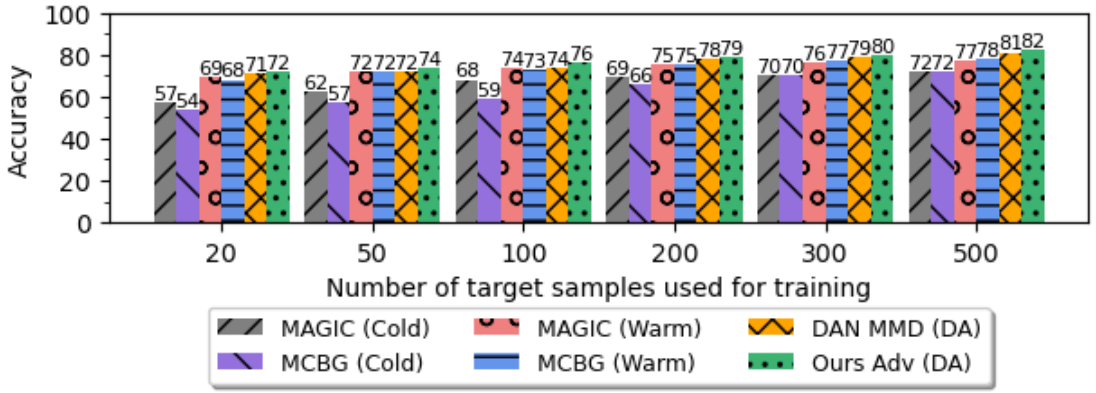


Fig. 7. The averaged accuracy on the target testing data using the experimental setup described in Fig.

Our approach achieves the same prediction accuracy as labeling 75% of the data (e.g., 1209/1613 in August and 1002/1337 in September) with just 200 labeled samples per month. A warm-start strategy achieves the same result with 500 samples

Multi-family Classification

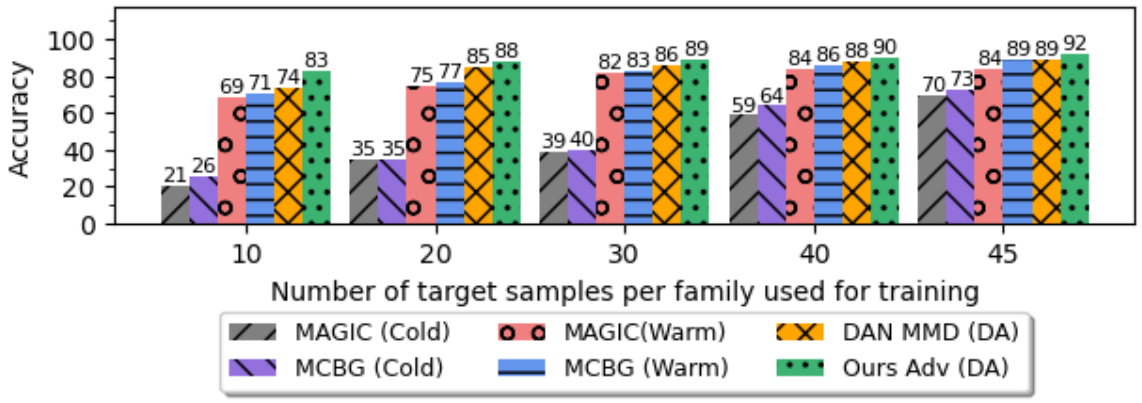


Fig 3. Accuracy on post-drift data using 10-45 labeled data from each post-drift family.

Our method improved family-level classification by 9 − 14% over the baselines with just 10 new samples per family

TABLE IV. EVASION RATE AND DETECTED UNKNOWN SAMPLES RATIO ON THREE TESTING DATASETS WITH 1, 5 AND 9 NEW FAMILIES.

Metrics	1 new family	5 new families	9 new families
Evasion success rate (%)	1.05	4.04	4.46
Detected/Total unknown samples	2396/2476	7662/9122	8921/10711

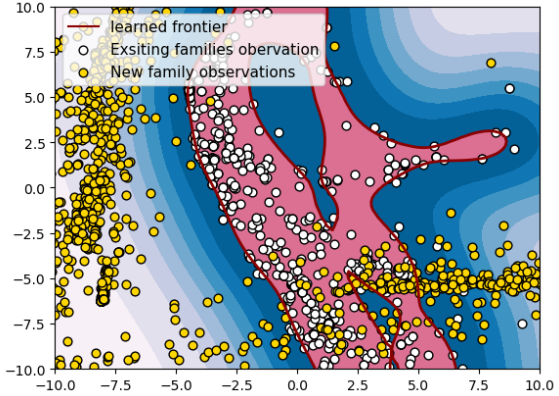


Fig 9. Visualization of the decision function learned by the outlier detection module: the new family observations are outside the learned frontier.

The evasion rate for new malware families remains consistently low (below 5%)

Conclusion

- We address the classification of drifted malware and the challenge of adapting models with limited labels
  - We introduce domain adaptation to train graph-based malware detectors

- Our approach outperforms strong baselines in three distinct adaptation tasks with increasing adaptation complexity:
  - Research datasets
  - Latest real-world malware samples
  - Classification of multiple malware families

- We highlight the importance of validating actual drift occurrences in research datasets
  - We show in the Big-15 dataset malware from different families exhibit highly similar characteristics
  - Evaluation relying on those family labels is likely to overestimate the accuracy of the prediction

Q&A

- Code: https://github.com/gloryer/malware-detection-concept-drift
- Email: li3944@purdue.edu