FS3: Few-Shot and Self-Supervised Framework for Efficient Intrusion Detection in Internet of Things Networks

Presenter: *Ayesha S. Dina*

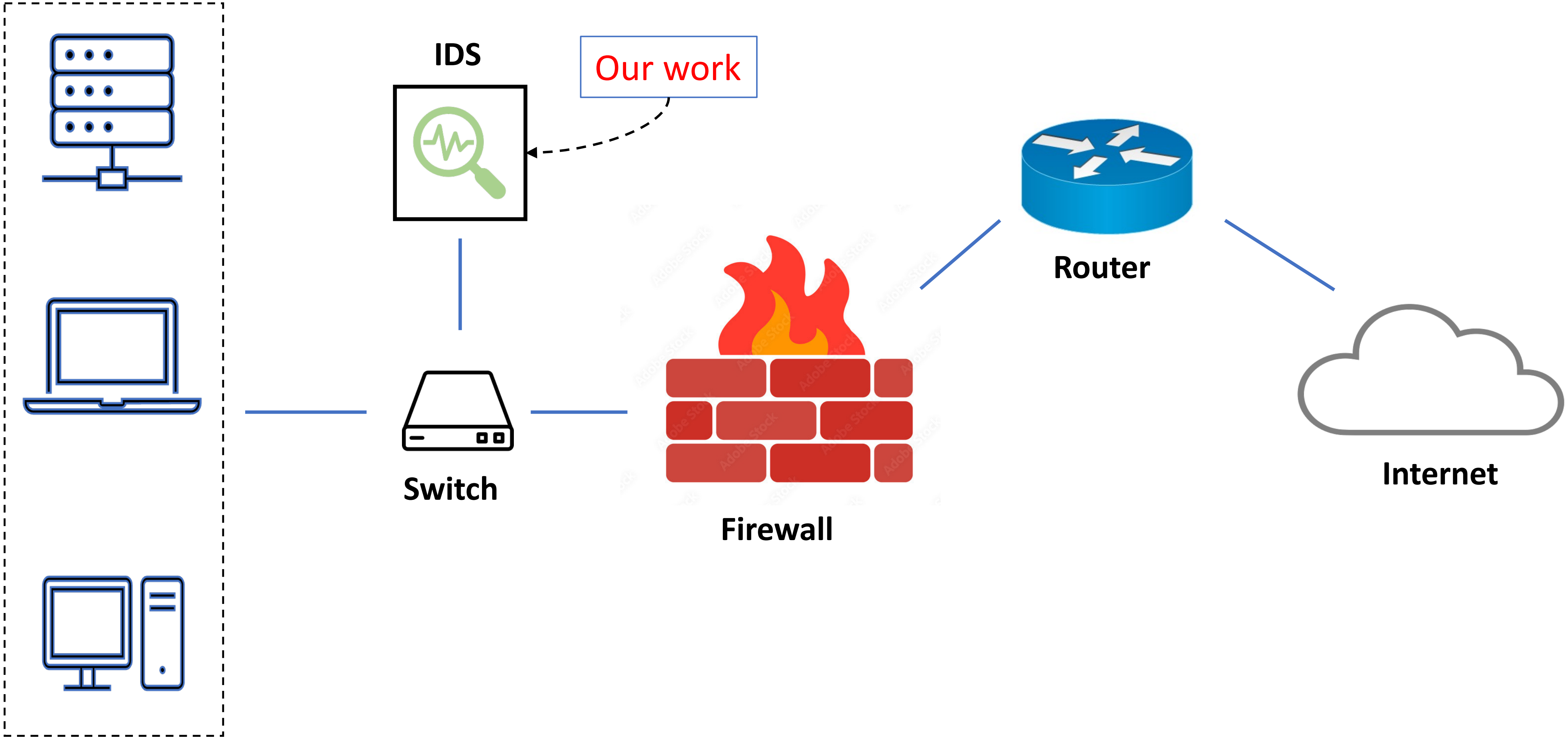
Authors: **Ayesha S. Dina, A. B. Siddique, D. Manivannan**

FS3- source code: https://github.com/ayeshasdina/FS3

Outline

- Intrusion Detection System (IDS)
- Background and related works
- Problem addresses in this work
- Proposed Framework
  - Phase 1: Self Supervised Learning
  - Phase 2: Triplet Loss Function
  - Phase 3: Nearest Neighbor Classification
- Experiment
  - Datasets
  - Evaluation
- Conclusion

Intrusion Detection in Computer Networks



Background and Related works

- Intrusion Detection System (IDS)
  - Signature based IDSes
  - Anomaly based IDSes
- Anomaly based IDSes use one of the following approaches
  - Statistical approach
  - Knowledge based approach
  - Machine learning (ML) approach

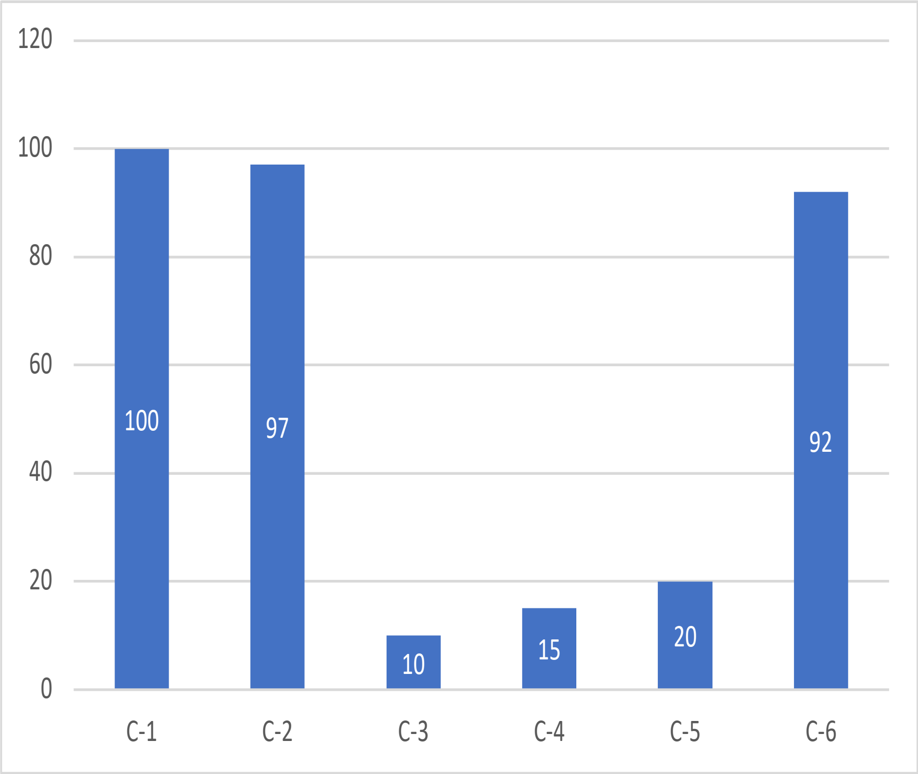
Background and Related works: Literature Review

- IDSes based on ML- Binary classification and multi class classification
- Many ML- classifiers such as KNN, DT, FNN, etc. were used.
- Used different datasets - KDD99, NSL-KDD, UNW-NB15, etc. for evaluation
- In all these datasets, the data is imbalanced. i.e., not all attack classes had equal number of samples
- Imbalanced data could yield poor performance on the ML.
- Traditional approaches,
  - Up sampling and down sampling for balancing data.
  - Each method faces the challenges of overfitting and underfitting, respectively.

Ayesha S. Dina and D. Manivannan, "Intrusion detection based on Machine Learning techniques in computer networks", in Internet of Thing journal, 8 October, 2021 , (**h5-Index: 42**)

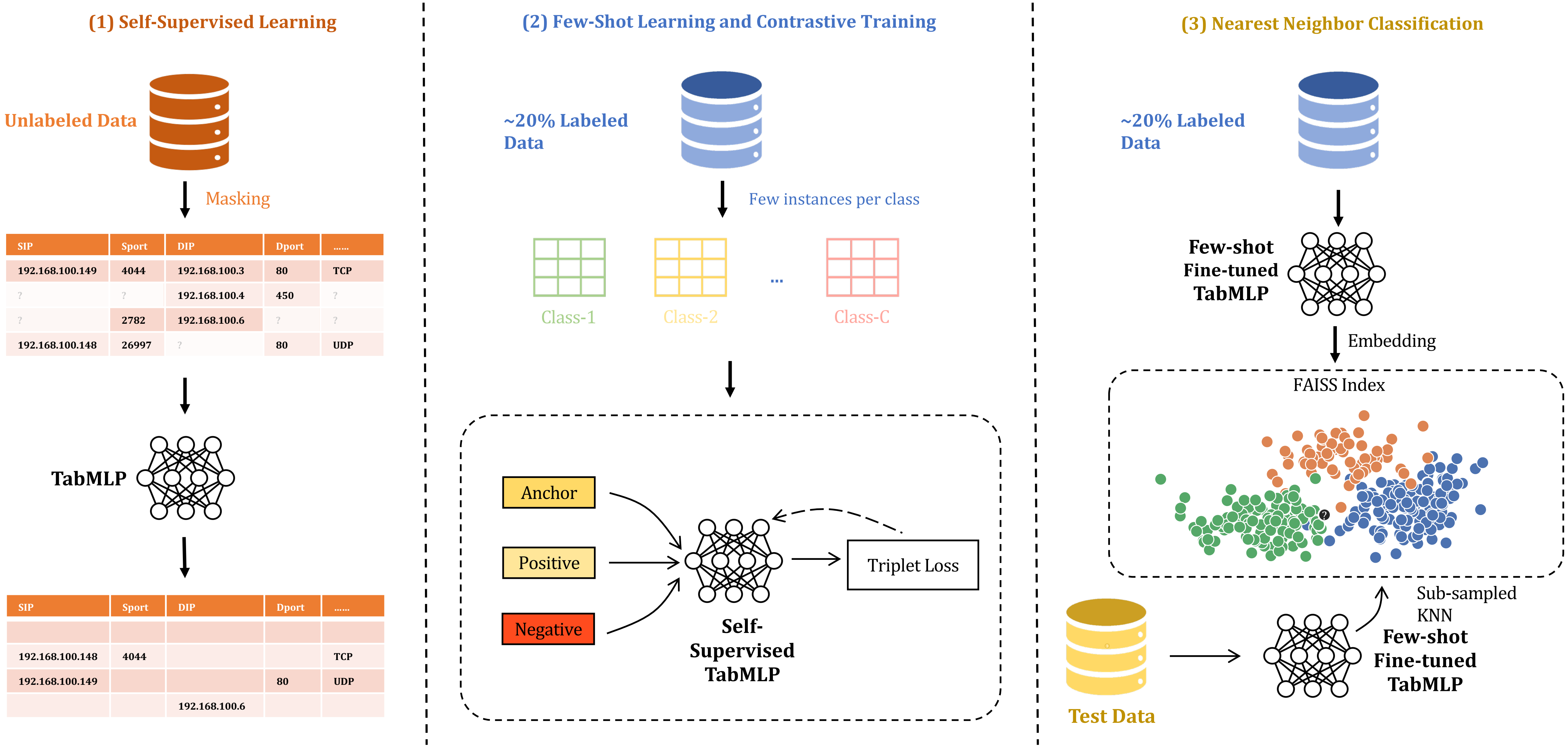
Problems addressed in this work

- Data Imbalance in datasets used for evaluation
- Lack of availability of large labelled datasets



Imbalanced data

FS3: A Framework for Intrusion Detection

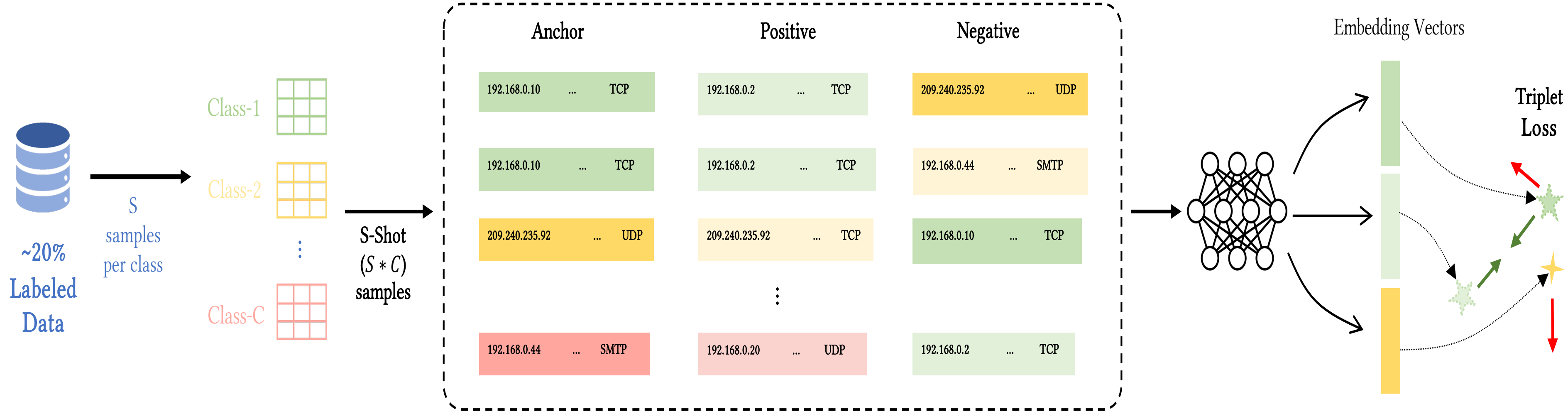


Phase 1: Self-Supervised Learning

- We use TabNet (Attentive Interpretable Tabular Learning) as the backbone model for self-supervised learning.
  - Use encoder-decoder structure to learn important features from the input data and predict the masked or target variable.
- We used the masking objective to mask 20% of the features in the input data.
- Tabular Multilayer Perceptron (TabMLP)
  - Two dense layer

Phase 2: Triplet Loss Function

- Utilize Few-shot learning (FSL) with contrastive training to further train the pre-trained model using a small number of instances.
- Leverage Triplet Loss:
  - Minimize the distance between the anchor and the positive point
  - Maximizing the distance between the anchor and the negative point.
- Shots:
  - 5-Shot and 10-Shot
- Perform contrastive training five times for both 5-Shot and 10-Shot scenarios



Overview of few-shot learning using contrastive training with triplet loss

Phase 3: Nearest Neighbor Classification

➢Used FAISS library for efficient and scalable search

➢We used a sub-sampled KNN algorithm to further address class imbalance :

$$W_i = Max(1 - \sqrt{\frac{t}{p_i}}, a)$$

Where:

- W_i is the weight assigned to the ith class.
- t is a hyperparameter controlling sub-sampling.
- p_i is the size in the ith class.
- a is a constant defining the minimum weight for each class.

Experiment: Datasets

Table 1: Statistics of WUSTL-EHMS dataset.

Class	Train	(%)	Test	(%)
Normal	10275	87.47	2855	87.44
Attack	1472	12.53	410	12.56

Table 2: Statistics of WUSTL-IIoT dataset.

Class	Train	(%)	Test	(%)
Normal	797261	92.71	221462	92.71
DoS	56379	6.56	15661	6.56
Reconnaissance	5932	0.69	1648	0.69
Command Injection	185	0.02	52	0.02
Backdoor	152	0.02	43	0.02

Table 3: Statistics of Bot-IoT dataset.

Class	Train	(%)	Test	(%)
DDoS	1233052	52.52	385309	52.51
DoS	1056118	44.98	330112	44.99
Reconnaissance	58335	2.48	18163	2.48
Normal	296	0.01	107	0.015
Theft	52	0.002	14	0.002

Experiment: Evaluation

- Quantitative Evaluation
  - Precision
  - Recall
  - F1_Score
- Qualitative Evaluation
  - Draw some samples from the dataset and perform t-SNE projection to evaluate the performance of various methods on this subset.
- Ablation Study
  - Gaining a more profound comprehension of how each component impacts the model's efficacy facilitates the evaluation and enhancement of our approach.

Experiment: Competing Methods

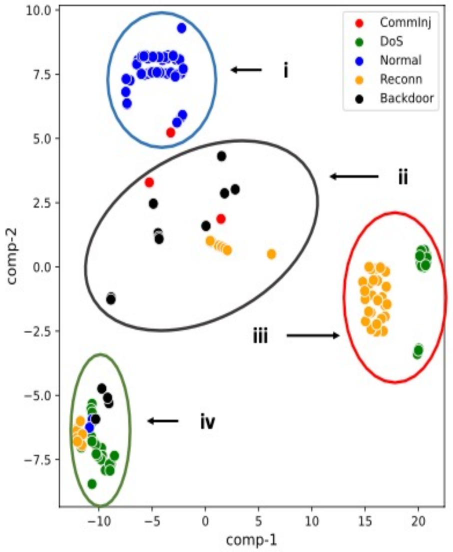
- State-of-the-art Models
  - CNN-BiLSTM
  - PB-DID
  - DBN-IDS
  - CTGANSamp: *Models were trained on the training datasets balanced using synthetic samples.*
  - Focal: *Models were trained using focal loss function*
- Baseline Models
  - ORG: *Models were trained using original datasets (i.e., without balancing the dataset)*
  - RND: *Models were trained on the datasets, balanced using random oversampling*
  - Dice: *Models were trained using dice loss function*

Evaluation: Quantitative Analysis

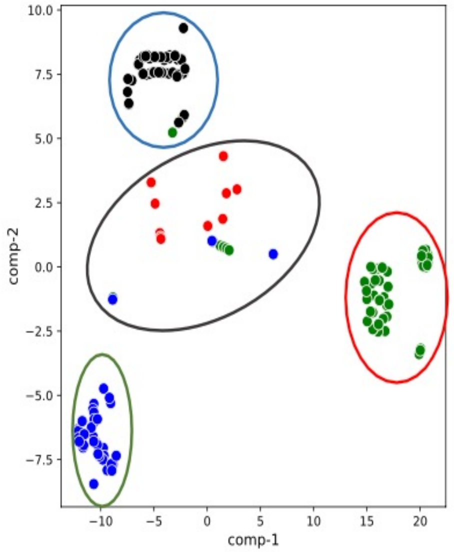
Table 4: Performance comparison of all methods on WUSTL-EHMS, WUSTL-IIoT, and BoT-IoT datasets.

DL Models	Classifier's Name	WuSTl-EHMS			WuStl-IIoT			BoT-IoT		
		Pre	Rec	F_1	Pre	Rec	F_1	Pre	Rec	F_1
State-of-the-art Models	CNN-BiLSTM [45]	0.9010	0.7305	0.7851	0.7222	0.4349	0.5086	0.2477	0.2563	0.0778
	PB-DID [59]	0.4372	0.4998	0.4664	0.2105	0.1110	0.0214	0.1717	0.2037	0.1448
	DBN-IDS [5]	0.7362	0.7105	0.7222	0.4631	0.6339	0.3851	0.1185	0.5582	0.1652
	FNN-CTGANSamp [12]	0.9294	0.7364	0.7962	0.6628	0.3437	0.4050	0.4991	**0.8652**	0.5540
	CNN-CTGANSamp [12]	0.9107	0.7360	0.7921	0.7025	0.5122	0.5533	0.4298	0.7988	0.4536
	FNN-Focal [13]	0.9524	0.7369	0.8011	0.3854	0.293	0.3194	0.5559	0.6380	0.5784
	CNN-Focal [13]	0.9423	0.7338	0.7963	0.8198	0.6617	0.6974	0.6165	0.6325	0.5853
Baseline Models	FNN-ORG	0.9382	0.7359	0.7975	0.5254	0.4151	0.4578	0.5073	0.6345	0.5436
	FNN-RND	0.9339	0.7367	0.7974	0.5834	0.7630	0.5850	0.4990	0.4990	0.5275
	FNN-Dice	0.9336	0.5000	0.4665	0.1854	0.2000	0.1924	0.0900	0.2000	0.1241
	CNN-ORG	0.9284	0.7327	0.7927	0.7486	0.6142	0.6558	0.4434	0.5347	0.4211
	CNN-RND	0.9272	0.7362	0.7956	0.5894	**0.7720**	0.5942	0.5349	0.7843	0.5680
	CNN-Dice	0.0628	0.5000	0.1116	0.1854	0.2000	0.1924	0.0900	0.2000	0.1241
FS3 **(This work)**	AVG 5-Shot	**0.9794**	0.9812	0.9801	**0.8897**	0.6804	0.7017	0.6198	0.6030	0.5960
	AVG 10-Shot	0.9698	**0.9941**	**0.9809**	0.7847	0.7050	**0.7144**	**0.6314**	0.6297	**0.6046**

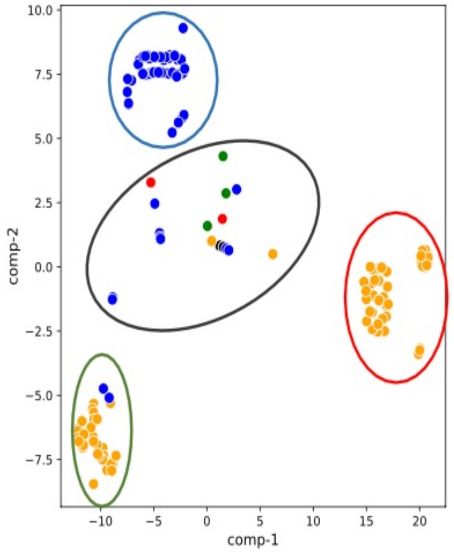
Evaluation: Qualitative Analysis



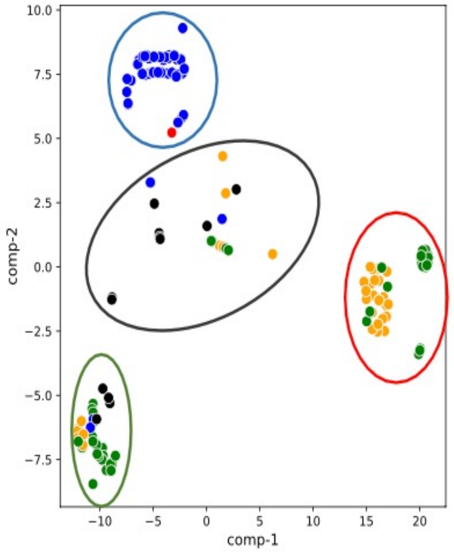
(a) Ground Truth



(b) CNN-RND



(c) CNN-Focal



(d) FS3

Evaluation: Ablation Study

Components Name	Phase of FS3	Different Strategy of KNN	WUSTL-EHMS			WUSTL-IIoT			BoT-IoT		
			Pre	Rec	F_1	Pre	Rec	F_1	Pre	Rec	F_1
Self-Supervised Encoder	Phase 1	Classical	0.8434	0.7703	0.8007	0.7249	0.6488	0.6769	0.5248	0.5388	0.5043
		Inverse of Class Size	0.8567	0.8179	0.8357	0.6460	0.6829	0.6569	0.4790	0.6381	0.4891
Fine-tuned Encoder	Phase 2	Classical	0.9294	**0.9708**	0.9488	0.7095	0.7023	0.6925	0.5581	0.7952	0.5645
		Inverse of class Size	0.7965	0.9477	0.8458	0.6276	**0.7457**	0.6713	0.5518	**0.7995**	0.5531
	Phase 3	Sub-Sampled KNN	**0.9571**	0.9533	**0.9552**	**0.9774**	0.6734	**0.7154**	**0.5904**	0.7176	**0.5871**

Conclusion

- FS3
  - Self-supervised learning, which utilizes SSL to extract latent patterns and robust representations from unlabeled data
  - Few-shot learning (FSL) and contrastive training, which enables the model to learn from a small number of labeled examples
  - Sub-sampled KNN- based classification

- FS3 leverages only 20% of the labeled training samples for making predictions, reducing the reliance on a large amount of labeled data as well as minimizing the startling effect of extreme class imbalance

- Source code: *https://github.com/ayeshasdina/FS3*

Thank you!!!

Website: *https://ayeshasdina.github.io/*