8th International Symposium on Cyber Security Cryptography and Machine Learning (CSCML 2024)

TECHNICAL REPORT

Editors

Shlomi Dolev, Oded Margalit and

Yonah Alexandre Bronstein

Technical Report

December 29, 2024

The BGU-Negev Hi-Tech Faculty Startup Accelerator,

Department of Computer Science,

Ben-Gurion University, Beer Sheva, Israel

Table of Contents

Ph.D./ Master's Student Research Track
Chair: Oded Margalit

Ph.D./ Masters Student Research Track chaired by Prof. Oded Margalit

In the PhD track this year, we've heard from all the letters of the acronym CSCML:

- **C**yber-**S**ecurity: for example, a paper from Ariel University in Israel about detecting security patches in git repositories;
- **C**ryptology: for example, a work from Amrita University in India about cryptanalysis of SLIM cipher;
- **M**achine **L**earning: for example, research from Ben Gurion University in Israel about scoring the quality of physiotherapy treatment from video input.

Like in previous years, the audience of this session enjoys a "tasting menu" of the state-of-the-art in CSCML research topics, while the speakers get experts' feedback on their work.

I enjoy chairing these sessions and invite all relevant researchers from all over the world to actively attend CSCML-2024, either as an attendee, or, preferably, as a speaker.

See you at CSCML 2025

Regards,

Prof. Oded Margalit,
CSCML 2024 Ph.D./Masters Track Chair
Computer Science Department, BGU
and Advanced Research Center, Trellix

Ad Hoc Blockchain for Private Contracts*

(Preliminary Version)

Hillel Avni[1], Shir Buchner[1], Shlomi Dolev[1], and Moti Yung[2]

[1] Department of Computer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel
[2] Google, USA

Abstract. Off-chain blockchain contracts are useful for parties that do not want to reveal records of the contract activities. Ad hoc installation of blockchain instances over mutually agreed-upon servers forms a reliable infrastructure for such contracts. We detail the core post-quantum algorithmic ingredient for structuring ad hoc blockchain infrastructure and contracts.

- Initialization, including the agreement on servers and blockchain software, e.g., Ethereum version. Approved choices by bidding digital signatures.
- SMPC-based zero-knowledge proof of Lamport signature. Efficient implementation extending the ZKBoo (SMPC in the head) to the case of verified signatures and a guarantee for signature release when two participants out of five may decide not to cooperate in the SMPC or reveal the signature according to the contract command.

Keywords

Blockchain, Post Quantum, Ad Hoc

1 Introduction

The rapid proliferation of blockchain technology has ushered in a new era of decentralized finance (DeFi), where individuals can engage in financial transactions without the need for traditional intermediaries. Private contracts over an ad-hoc instance of a mature blockchain software yields the benefit of open source reliability while keeping total privacy on the buisniss logic and parties identity.

The contracts are general in nature allowing the state of the art contracts to run on the state of the art blockchain software. For example, one of the newly developed key features of the blockchain landscape is the ability to swap various tokens across different blockchain networks. However, as these swaps occur on bridges between networks, there is not yet a solution to ensure the completeness of trades between untrusted parties, involving different tokens.

Ingredient of the solution. Secure multi-party computation in the head is used for post-quantum zero-knowledge proofs. There is no need to consider adversarial participants in the multi-party computation in the scope of such zero-knowledge in the head proofs. The design of such secure multi-party computation can, therefore, become efficient.

We demonstrate the usefulness of such a design in the scope of distributed zero-knowledge proof of a signature beyond the centralized SMPC in the head. Using SHA256 as a building block of post-quantum Lamport signature, implemented by [8], we can verify the computation of all participants, as the output should be identical to the signed message digest.

To avoid fake output, we need to ensure that the output of the SMPC is revealed atomically by all the participants. To do so, each participant commits to its output share (say by computing and outputting the SHA256 result over its share) before the first share is revealed.

If three participants execute the SMPC, they can reveal the signature prematurely and nullify the contract conditions (e.g., skipping the approval of goods reception) for releasing the signature. In the

* Partially supported by the Rita Altura Trust Chair in Computer Science and the Israeli Science Foundation (Grant No. 465/22). Contact author: Shir Buchner.

case of three participants, two or fewer participants cannot reveal their signatures based on their own decisions.

Another aspect we must consider is revealing the signature according to the contract instructions. In [5] all three participants should reveal their secret shares, and if even one does not obey the signature cannot be used. Thus, we first suggest employing four participants instead of three and running in parallel four (randomization) independent SMPC instances, each omitting a distinct participant of the four. Thus, if only one of the participants refuses to reveal their signature share, an SMPC instance will exist that excludes this participant.

To make the scheme robust against a collusion of two participants, we may use five participants and have each combination of five choose three (namely, twenty) running an instance. Alternatively, we may use nine servers, where three distinct groups of three servers run an SMPC version on (randomization) independent shares, thus running only three (rather than twenty) instances of the original SMPC.

Related work. Arbitrators for enforcing private ad hoc infrastructure for enforcing agreed-upon contracts were introduced in [3] and [7]. The contract was designed based on the identity revealing of anonymous dishonest participants. Later [1], general contracts were introduced by the Ethereum blockchain global infrastructure. Here we employ the stable and worldwide useful Ethereum blockchain and contract infrastructure for the cases of ad-hoc private contracts.

2 Framework and Installation

The use of cryptocurrency (and limited-time contracts) over other chains is an example of a "check" with an unrevealed signature.

The root of trust for our scheme can be contracts on global (rather than ad hoc) blockchain. For example, contracts in the global Ethereum and Solona. Continuing the example, the contract of *Alice* is visible to *Bob* on the Ethereum blockchain. The contract specifies that the Ethereum cryptocurrency, *ETH*, will be transferred to *Bob* when a signature of *Alice* on the transfer is released, the contracts reserve the amount of Ether to be paid for long enough period that suffices to complete the deal. On the other side, *Bob* exposes a contract over Solona to *Alice*, with a reservation of the cryptocurrency *SOL* to be transferred to *Alice* when the signature of *Bob* for the transfer is released.

The contracts over the global Ethereum and Solona composed by *Alice* and *Bob*, respectively, declare additional conditions for transferring the cryptocurrencies. The contract of *Alice* (*Bob*, respectively) requires that the initialization process of the ad hoc blockchain instance is approved by a signature of *Alice* (*Bob*, respectively). Thus, both parties agree on the set of servers that participate in the ad hoc blockchain, the parties keep the signatures, to be presented to the global blockchain contracts. Similarly, both parties sign the software version of the ad hoc blockchain, to be checked by the global contracts before transferring the cryptocurrency.

3 Implementation

Our scheme is an extension of the ZKBoo [5] in-the-head, and aims to reduce prover's computational burden, and proof size. These parameters grow as we replace the ZKBoo (n, n), with $(3, n)$ sharing, allowing $n - 3$ adversarial parties. Unlike ZKBoo, the decentralized approach does not rely on honest execution, which can be difficult to ensure in some practical implementations. In the following, Section 3.1 presents XOR-based secret sharing, Section 3.2 discusses the flow of the contract, and Section 3.3 explains how we support bizantine parties.

3.1 XOR-based secret sharing (XSS)

The polynom-based Shamir's Secret Sharing (SSS) is homomorphic for addition and multiplication but bitwise operations such as XOR or AND, which are common in SHA256, will require a separated field-size share for each bit [4,2]. XSS supports the reconstruction of XOR operations among shares with minimal overhead, and the overhead it has on ADD is minimal, which is why it is used here and in [5] to calculate SHA256 in secret sharing. XSS splits the input X into n secret shares X_1, $X_2 ... X_n$, such that:

- $X_1 ... X_{n-1}$ are perfectly random
- $X_n = X_1 \oplus X_2 ... \oplus X$
- $X = X_1 \oplus X_2 ... \oplus X_n$ (reconstruction)

XSS has several advantages over SSS. XOR operations are computationally light, making them highly efficient and performant. XOR-based sharing has minimal storage overhead because shares are the same size as the original secret, while SSS often increases share size due to field expansion.

Some operations, such as XOR, RIGHTROTATE and SHIFT are naturally reconstructed. They work on the local secret share and do not require communication. Other functions, such as AND or ADD, require communications, and each party requires input from one other party.

Algorithm 1 AND function for XSS

```
1: function MPC-AND(x, y, i)
2:     r ← R[i]
3:     write(x, y, r)
4:     x', y', r' ← read()                    ▷ The read acts also as a barrier
5:     result ← (x ∧ y') ⊕ (x' ∧ y) ⊕ (x ∧ y) ⊕ r ⊕ r'
6:     i ← i + 1
7:     return result, i
8: end function
```

The AND operation is demonstrated in Algorithm 1. If we send the result of one cycle to the next party, to execute the next cycle, it will combine this with its local result and be able to extract all the parts and find the secret and calculate the result. In line 5, the MPC-AND is XORing the result with $r \oplus r'$ to avoid this volunerability. As all parties multiply by this pattern, each random number is XORed twice and the reconstructed result is correct.

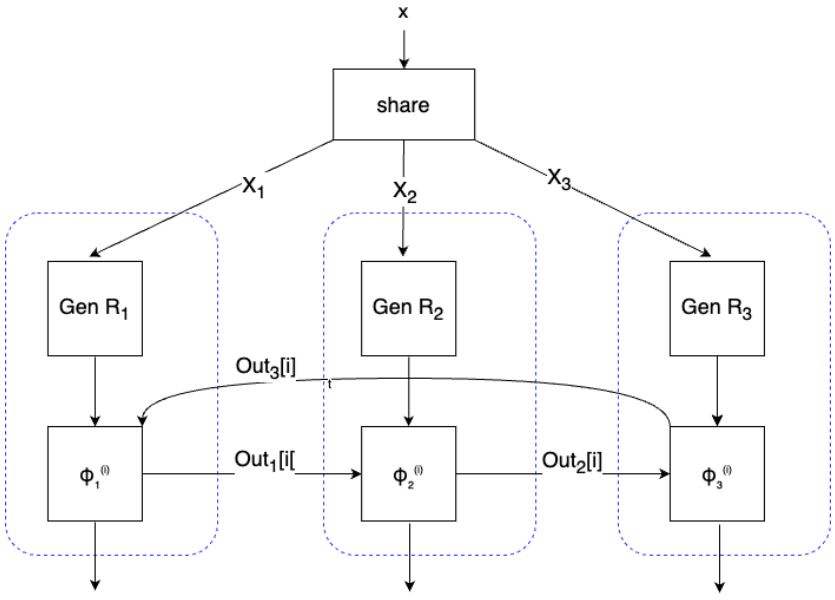


Fig. 1. Contract is signed and peocessed by the trusted parties

3.2 Contract Flow

The trusted parties receive XOR-based secret shares of the original private key. The parties collaborate in (CMP) computation of SHA256 of the private key, and then exposes the SHA256 of its secret share

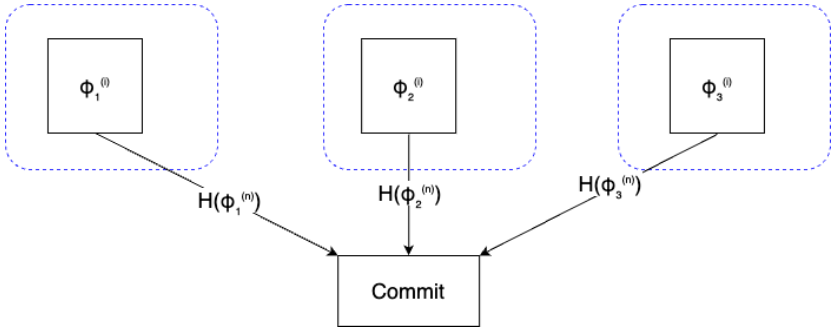


Fig. 2. The parties commit to the share they will publish

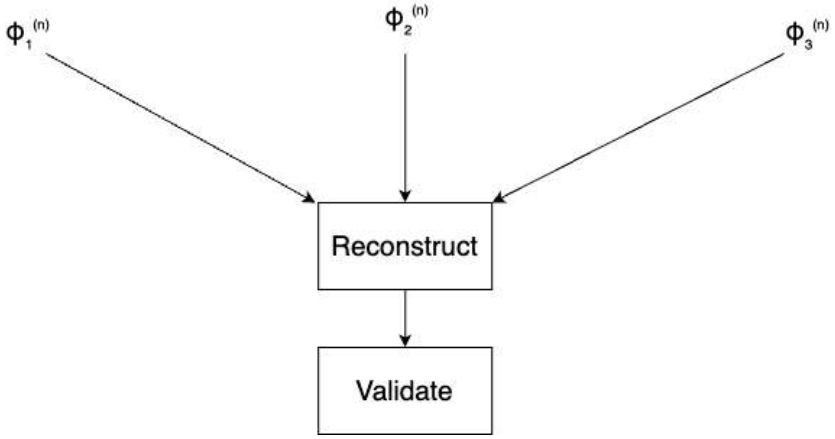


Fig. 3. Validate the SHA256 of tyhe preimage is correct

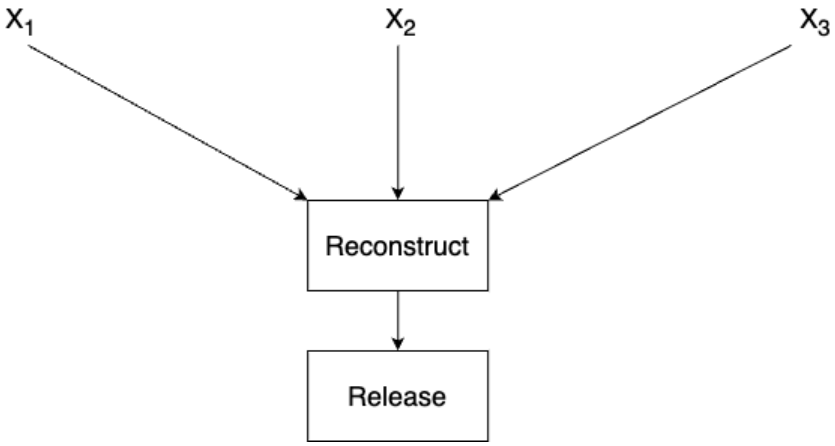


Fig. 4. Get the contract preimage and release the payment

of the output. The calculation takes n communication cycles, and each party generates a set of n random numbers for obfuscation. Fig. 1 shows that after each party gets the share of the private key, it processes it in cycles, and after each cycle, each party sends the result to the next. The pseudocode of a cycle that performs an AND operation is presented in Algorithm 1. Each cycle starts by sending its data (line 3), and continues reading (line 4). The read operations serialize the cycles, so each cycle gets the result of the previous one. After the calculation is completed, the distributed, quantum-safe smart contract is executing the following steps:

1. **Commit:** Output the hash of the secret share (Fig. 2) to prevent an adversarial party from altering it according to the other parties and get a false positive.

2. **Validate:** Output $\phi_k^{(n)}$ (Fig. 3), the SHA256 of the private key, to ensure that the seller can verify the validity of the contract.
3. **Release:** Output the share of the contract (Fig. 4) so that the money is released and the contract is completed.

3.3 Allowing Bizantine Parties

In the release phase, an advererial may stop and prevent the release of the payment. Preventing this scenario requires an (n, k)-threashold, so $(n\text{-}k)$ adversaries can be tolerated. XOR-based schemes are particularly advantageous for (n, n) where all n shares are needed to reconstruct the secret, offering simplicity and speed without compromising security. However, their straightforward nature makes them less flexible than SSS for general (k, n)-threshold schemes. In [6] they implement (k, n) for XOR by distributing $\binom{n}{k}$ sets of k shares among the n participants, and there exists a set of k shares for each group of size k. We adapt their method. The algorithm can tolerate $(n-3)$ slow responders, but this does not mean any three parties can verify, and the number of participants that are required to verify can be decided by the dealer, e.g. it can demand two groups with at least four participants to emit a positive result.

References

1. Vitalik Buterin. Ethereum: A next-generation smart contract and decentralized application platform. https://ethereum.org/en/whitepaper/, 2013. Accessed: 2024-11-18.
2. Ivan Damgård, Matthias Fitzi, Eike Kiltz, Jesper Buus Nielsen, and Tomas Toft. Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*, pages 285–304. Springer, 2006.
3. Shlomi Dolev, Niv Gilboa, and Ofer Hermoni. Poster: arbitrators in the security infrastructure, supporting positive anonymity. In Yan Chen, George Danezis, and Vitaly Shmatikov, editors, *Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS 2011, Chicago, Illinois, USA, October 17-21, 2011*, pages 753–756. ACM, 2011.
4. Shlomi Dolev and Ziyu Wang. Sodsbc/sodsbc++ & sodsmpc: Post-quantum asynchronous blockchain suite for consensus and smart contracts. In Colette Johnen, Elad Michael Schiller, and Stefan Schmid, editors, *Stabilization, Safety, and Security of Distributed Systems - 23rd International Symposium, SSS 2021, Virtual Event, November 17-20, 2021, Proceedings*, volume 13046 of *Lecture Notes in Computer Science*, pages 510–515. Springer, 2021.
5. Irene Giacomelli, Jesper Madsen, and Claudio Orlandi. Zkboo: Faster zero-knowledge for boolean circuits. In Thorsten Holz and Stefan Savage, editors, *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016*, pages 1069–1083. USENIX Association, 2016.
6. Jun Kurihara, Shinsaku Kiyomoto, Kazuhide Fukushima, and Toshiaki Tanaka. On a fast (k, n)-threshold secret sharing scheme. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 91-A(9):2365–2378, 2008.
7. Shlomi Dolev Ofer Hermoni, Niv Gilboa. Digital arbitration, USA patent number US20140101440A1, 2012.
8. Sobuno/ZKBoo. MPC_SHA256.

Deep Learning-based Network Intrusion Detection System using Image Data

Leonard L. Mutembei[1,2], Makhamisa C. Senekane[2,3][0000−0002−0122−3076], and Terence Van Zyl[1,2][0000−0003−4281−630X]

[1] Academy of Computer Science and Software Engineering, University of Johannesburg, South Africa lmutembei@uj.ac.za
[2] Institute for Intelligent Systems, University of Johannesburg, South Africa {smakhamisa,tvanzyl}@uj.ac.za
[3] National Institute for Theoretical and Computational Sciences, South Africa

Abstract. Deep Learning algorithms have been advancing with the technology. Convolutional Neural Network (CNN) architecture is among the prominent deep learning algorithms for processing data in image, video, and text formats. Network security experts have been using deep learning techniques to implement Network Intrusion Detection Systems (NIDS) to fight against cyber attacks. This study used CNN to implement the NIDS model in multi-class classification with the NSL-KDD dataset. The evaluation metrics which were used in this study are accuracy, precision, recall, F1-score and weighted average. The results showed how the implemented model has detected minority classes in an imbalanced dataset. Furthermore, the study showed the connection between batch size and validation split size with the performance of the minority classes.

Keywords: Convolutional Neural Network · Deep Learning · Network Intrusion Detection System.

1 Introduction

Cyberspace provides opportunities for good and bad people to use. The advancement of technology has also led to complex cyber attacks. [2] shows how cyber attacks like Distributed Denial of Service (DDoS) have been in use by different groups. The advancement of deep learning helps implement network security tools like Network Intrusion Detection System (NIDS) to protect cyberspace [6]. Network traffic in a real environment is imbalanced; the normal traffic data has many data points, compared to malicious traffic data. [5] showed how the imbalanced data affect the performance of the classifiers, which may mislead the network security experts. Furthermore, [4] has suggested ways to handle imbalanced data. Thus, the deep learning-based NIDS model favours the majority traffic during detection and fails to detect the minority traffic.

Researchers have been working in deep learning-based NIDS to improve the detection rate and reduce the false alarm rate (FAR). [11] has worked using CNN and pre-processed the dataset using a discretization algorithm and transformed data into image format. The study used an imbalanced dataset and managed to identify

minority classes. Many researchers try to balance the datasets, while normal traffic is always more than malicious traffic in a real network environment. Deep learning models need reproducibility properties for other researchers to validate results. Information about datasets, methodologies, and hyper-parameters are needed to help the models be replicated [3, 7].

In this study, we demonstrate the deep learning-based NIDS algorithm using CNN. The remainder of this paper is structured as follows. The next section discusses the methodology used in the study. This is followed by the section which provides and discusses the results obtained. Finally, the last section concludes this paper.

2 Methodology

The study used CNN architecture with convolutional layers, batch normalization and maximum pooling. Also, we used the dropout layer and Adam as an optimizer.

2.1 Dataset

The NSL-KDD dataset was used to evaluate the model. It was developed in 2009 by solving the limitations of a KDDcup99 dataset [10]. The NSL-KDD dataset has grouped the data into a training dataset of 125,973 samples and a test dataset of 22,544 samples, as shown in Table 1. The network attack categories are normal, Denial of Service (DoS), Remote to Local (R2L), User to Root (U2R) and Probe.

Table 1: NSL-KDD Dataset Distribution

Type	**Distribution**	**Percentage (%)**
Train data	Normal:67343	53.46
	DoS:45927	36.46
	R2L:995	0.79
	Probe:11656	9.25
	U2R:52	0.04
Test data	Normal:9711	43.08
	DoS:7458	33.08
	R2L:2754	12.22
	Probe:2421	10.74
	U2R:200	0.89

The CNN algorithm performs better in image data. Different ways are used to transform tabular data to images (Tab2Img). [8] implemented an algorithm which inspired the development of the Tab2Img Python tool. The use of transfer learning with distillation has shown promising results in transforming tabular data into image data [1]. Furthermore, [9] proposed a method to transform non-image data into images from 1-dimensional vectors to 2-dimensional images. In this study, we have preprocessed

the dataset and used a tool to transform tabular data into image format. Tab2Img was used to convert network traffic from tabular format to images automatically. The transformed data was inserted into the CNN classifier for prediction. The implemented model is based only on one architecture from the previous study [11].

2.2 Tuning Hyper-parameters

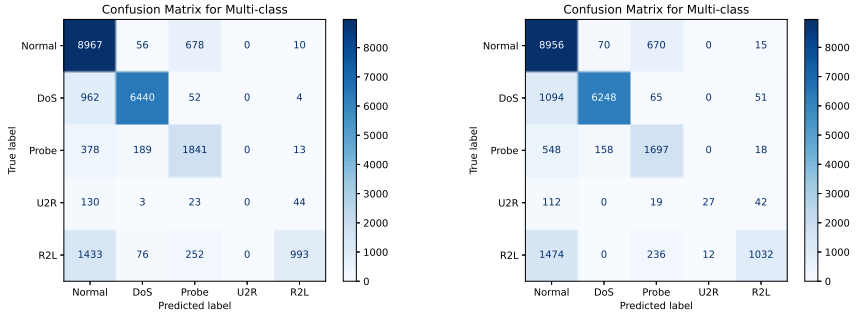
The study used Google Colab on a free account but failed due to the limited time allocated. We then used a local machine with Ubuntu 22.04.2 LTS, Intel Core i7, 2.80GHz, and 16GB of RAM. A random search was used to find the optimal parameters. The search used 12 hours to fine tune hyper-parameters, which are batch size of 32, decay of $5.061193489414997e-07$, dropout of 0.5 and learning rate of $6.448231721948543e-05$. After using the optimal parameters on the test dataset, the model could not identify the minority class, U2R. Thus, we changed the batch sizes from 32 to 64, 128 and 256.

3 Results and Discussion

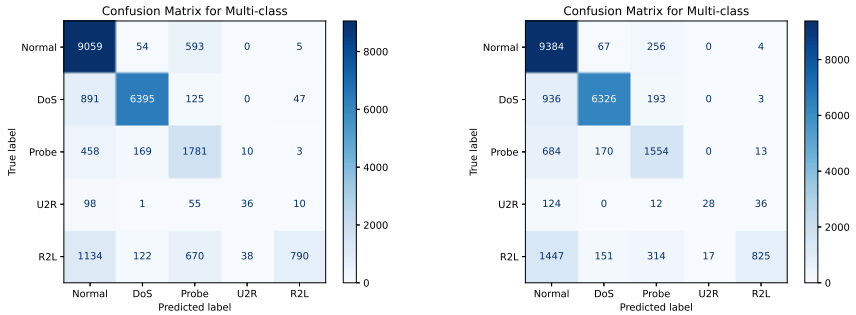
In this study, multi-class classification was done, and evaluation metrics used are accuracy, precision (PR), recall (RC), F1-score (F1) and Weighted Average (WAVG) of PR, RC and F1 scores. Figure 1 depicts the results from our implementation compared with [11] on the discretization of Continuous and Discrete Features (CD-1: grayscale and CD-2: colour image data). We used only grayscale image data and in some metrics, have outperformed even the coloured image data used by [11]. Increasing the batch size from 32 to 128 and 256 improved the performance of the minority samples like U2R, as depicted in Figure 2. Furthermore, the increase of the validation split from 20% to 35% and 40% also improved the performance. There is a possibility of increasing the model's accuracy by working on other hyper-parameters like the number of neurons and the number of hidden layers.

Study		Normal (%)	DoS (%)	Probe (%)	U2R (%)	R2L (%)	WAVG (%)	Patience: 9
Ours	PR	77	95	65	**0**	93	83	Batch size: 128
	RC	92	86	76	**0**	36	81	Validation split: 30%
	F1	84	91	70	**0**	52	80	Accuracy: 81.25%
		Normal (%)	DoS (%)	Probe (%)	U2R (%)	R2L (%)	WAVG (%)	Patience: 9
Ours	PR	77	96	63	**69**	89	83	Batch size: 128
	RC	92	84	70	**14**	37	80	Validation split: 35%
	F1	84	90	66	**23**	53	80	Accuracy: 80.09%
		Normal (%)	DoS (%)	Probe (%)	U2R (%)	R2L (%)	WAVG (%)	Patience: 9
Ours	PR	75	94	67	**62**	94	83	Batch size: 256
	RC	97	85	64	**14**	30	80	Validation split: 40%
	F1	84	89	65	**23**	45	79	Accuracy: 80.36%
		Normal (%)	DoS (%)	Probe (%)	U2R (%)	R2L (%)	WAVG (%)	Patience: 9
Ours	PR	78	95	55	**43**	92	83	Batch size: 256
	RC	93	86	74	**18**	29	80	Validation split: 35%
	F1	85	90	63	**25**	44	79	Accuracy: 80.16%
		Normal (%)	DoS (%)	Probe (%)	U2R (%)	R2L (%)	WAVG (%)	
[11] CD-1	PR	86	95	71	**25**	74	70	
	RC	94	86	75	**10**	69	67	Batch size: 256
	F1	90	90	73	**15**	71	68	Accuracy: **85%**
		Normal (%)	DoS (%)	Probe (%)	U2R (%)	R2L (%)	WAVG (%)	
[11] CD-2	PR	85	88	69	**78**	55	75	
	RC	91	88	68	**16**	63	65	Batch size: 256
	F1	88	88	68	**25**	70	68	Accuracy: **83%**

Fig. 1: Results Analysis.



(a) 128 batch size with 30% validation data (b) 128 batch size with 35% validation data



(c) 256 batch size with 35% validation data (d) 256 batch size with 40% validation data

Fig. 2: Confusion Matrices

4 Conclusion

In this study, we used Tab2Img to convert the tabular dataset into an image dataset. We implemented a deep learning model by using CNN architecture. The model favours majority samples while minority samples struggle to be identified. The study uses the NSL-KDD imbalanced dataset. Future work is to test the implemented model with other public network datasets and test the computational complexity of the model.

References

1. Bragilovski, M., Kapri, Z., Rokach, L., Levy-Tzedek, S.: Tltd: Transfer learning for tabular data. Applied Soft Computing **147**, 110748 (2023)
2. Brooks, R.R., Yu, L., Ozcelik, I., Oakley, J., Tusing, N.: Distributed denial of service (ddos): a history. IEEE Annals of the History of Computing **44**(2), 44–54 (2021)
3. Gundersen, O.E., Kjensmo, S.: State of the art: Reproducibility in artificial intelligence. In: Proceedings of the AAAI conference on artificial intelligence. vol. 32 (2018)
4. Johnson, J.M., Khoshgoftaar, T.M.: Survey on deep learning with class imbalance. Journal of big data **6**(1), 1–54 (2019)
5. Kumar, P., Bhatnagar, R., Gaur, K., Bhatnagar, A.: Classification of imbalanced data: review of methods and applications. In: IOP conference series: materials science and engineering. vol. 1099, p. 012077. IOP Publishing (2021)
6. Mutembei, L.L., Senekane, M.C., van Zyl, T.: Deep learning-based network intrusion detection systems: A systematic literature review. In: Southern African Conference for Artificial Intelligence Research. pp. 207–234. Springer (2024)
7. Semmelrock, H., Kopeinik, S., Theiler, D., Ross-Hellauer, T., Kowald, D.: Reproducibility in machine learning-driven research. arXiv preprint arXiv:2307.10320 (2023)
8. Sharma, A., Vans, E., Shigemizu, D., Boroevich, K.A., Tsunoda, T.: Deepinsight: A methodology to transform a non-image data to an image for convolution neural network architecture. Scientific reports **9**(1), 11399 (2019)
9. Sharma, A., Kumar, D.: Classification with 2-d convolutional neural networks for breast cancer diagnosis. Scientific Reports **12**(1), 21857 (2022). https://doi.org/10.1038/s41598-022-26378-6
10. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the kdd cup 99 data set. In: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications. pp. 1–6 (2009). https://doi.org/10.1109/CISDA.2009.5356528
11. Yoo, J., Min, B., Kim, S., Shin, D., Shin, D.: Study on network intrusion detection method using discrete pre-processing method and convolution neural network. IEEE Access **9**, 142348–142361 (2021)

Emergency Vehicle Accommodation in Dynamic Traffic Scheduling*

(Preliminary Version)

Shlomi Dolev, Ehud Gudes, Amit Hendin, and Hanna Yair

Department of Computer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel

Abstract. The passage of vehicles through a road network is an increasing challenge due to population density and urban centers around the world. Frequent road crossings and heavy traffic loads often lead to congestion and lengthy travel times. The impact of this problem is highlighted when emergency response vehicles (such as ambulances, firemen, police, etc.) pass through the road network, since they require the shortest possible travel times to perform. A traffic scheduling algorithm based on a cooperative vehicle that allows traffic to pass junctions and improve overall travel time through the network is devised in [1]. Building on the design in [1], we develop a modification to the algorithm that can accommodate the schedule of emergency vehicles passing through by allowing them to bypass non-emergency vehicles while maintaining a reasonable schedule on a first-come first-served basis.

Keywords: Traffic Management, Smart City, Scheduling.

1 Introduction

First of all we would like to express our gratitude to our partners in the Technion for their support in this research.

The ability to efficiently travel through a road network is one of vital importance to a functioning economy. In todays interconnected world where goods manufactured in once place and consumed in another, where people go to work far from where they live, transportation becomes vital to productivity. To enable an economy to grow, the time it takes to travel between two places must kept to a minimum, this is especially true with regards to first responders and emergency vehicles. Giving emergency vehicles the right of way is already common practice across the world, even letting them pass through red lights, but this practice can be very inefficient for the passage of traffic at large and at times even dangerous, incurring the risk of an accident when an emergency vehicle assumes right of way over an inattentive driver, especially so with todays many smartphone distractions. Efficiently managing traffic through intersections in the road network can be key to minmizing travel times and accommodating the passage of emergency vehicles through the network.

Previous work. An algorithm for scheduling vehicles at the various junctions they will enter during their journey is designed in [1], the algorithm optimised their path based on available space and time of arrival. The algorithm works by building a path for each vehicle that enters the road network, while building the path, the algorithm determines which junctions are fastest to pass through both in terms of distance and traffic levels. When choosing to direct a vehicle through a junction, the algorithm will then reserve a spot for the vehicle for the time it is expected to arrive at the junction. This then, creates a situation where each junction has a schedule of vehicles which will pass through it which is dependant on the schedule of neighboring junctions and the vechicles adherence to those schedules. Therefore, any deviation a vehicle makes from the schedule of it's next junction, will potentially cause deviation from the schedule of the following junction as so on..., potentially requiring an entirely new scheduling for all junctions ahead in it is path.

* Contact author: Amit Hendin, `hendina@post.bgu.ac.il`. The research is partially supported by the Rita Altura trust chair in computer science, the Israeli Smart Transportation Research Center (ISTRC), and the Israeli Science Foundation (Grant No. 465/22).

2 Prioritized Scheduling

In this paper we expand on the work on [1] by introducing a modification to their algorithm which allows emergency vehicles to have priority over other vehicles in the network in the scheduling of each junction they pass through, thereby integrating their right of way into the algorithm.

3 General dynamic traffic scheduling

To begin, we start with a simple procedure to schedule a single vehicle. This procedure is taken directly from [1] and uses the existing structures defined in the paper.
The procedure invokes the *dwDijkstra* algorithm in order to find a valid and optimal scheduling in the provided conflict graph for the input vehicle.
The variable P contains the path through the road network that was found, the variable T contains the times at which the vehicle must arrive at each intersection along the path, and the variable I determines the index of the slot scheduled of the vehicle at each intersection along the path.
Once these variables are obtained, the procedure updates the conflict graph and the vehicle of the scheduling in order to reserve the slots and notify the vehicle of which velocity to maintain in order to conform to the schedule. This step is dependant on the cooperation of the vehicle.

Algorithm 1 schedule a single vehicle

Require: $vehicle, CG$ and vehicle and the conflict graph
$P, T, I = dwDijkstra(vehicle.source, vehicle.destination, CG)$
$CG = update(vehicle, P, T, I)$
$vehicle.setVelocity(P, T, I)$

4 Accommodating emergency vehicles

In order to dynamically schedule vehicles as they leave and enter the road network, we maintain a queue of vehicles to schedule VQ and a queue of vehicles that have already been scheduled SQ. This allows us to reschedule vehicles as needed.
The idea behind the modification is as follows, when an emergency vehicle approaches a junction, it must be able to pass through as quickly as possible meaning without stopping. Practically in terms of the scheduling algorithm, it should be able to claim any slot in the junction it is heading for provided that slot is not currently occupied by another emergency vehicle of that same or higher priority. This means that if a vehicle of lower priority where to already be occupying that slot, that vehicle would need to be assigned a different slot, and since it's arrival time is assumed to already by the quickest, that different slot must be at a future time meaning the vehicle must be in a subsequent slot behind the emergency vehicle currently being scheduled. We can therefore obtain a optimal scheduling with the emergency vehicle passing at the time optimal for it by simply rescheduling all vehicles that where scheduled prior to it.

This approach may not be optimal in terms of runtime, since we may reschedule vehicles that where unaffected by the emergency vehicle thereby rescheduling them right back to where their original schedule. However, it does gaurantee an optimal schedule which respects first come first serve under the given prioritization constraint.

4.1 Prioritization Order

In order to further optimize the flow of traffic, we consider passage of vehicle platoons through the network. A platoon of vehicles is defined as a group of cars moving in the same direction as a selected car, called, the representative vehicle, and at a distance of each other which is less than a given threshold. If we consider giving vehicles in a platoon a higher priority, we can both encourage their

formation and, further optimize our schedule. By clearing the slots for the platoon vehicles, we can make sure those slots which may have been sparsely populated to now be densly populated. For this reason, we assign regular vehicles a priority level of 0, and platoon vehicles a priority level of 1 and emergency vehicles an even priority level $2k$ for $k > 0$ and emergency vehicle platoons a priority level of $2k + 1$ for the varing levels of urgency k.

4.2 The Algorithm

Definition 1. *For any vehicle v, let $v.priority$ be the priority level of v represented by an integer.*

Definition 2. *Let VQ be a priority queue of vehicles ordered by time of entry into the road network and prioritized by $v.priority, \forall v \in V$.*

Outline of the algorithm. The algorithm works by removing the reserved slots of all vehicles of lower priority before scheduling a vehicle, then inserting them to the front of the queue by order of time they entered the road network. This will ensure that they will be rescheduled before we schedule the next vehicle and in their original order. We do this by breaking up SQ into 2 temporary queues TSQ and TVQ, where TSQ will contain the vehicles or equal or greater priority, and TVQ will contain the vehicles of lower priority. We then apply the rescheduling operations to the vehicles in TVQ by the order they appear in that queue and take TSQ as a new SQ.

The algorithm starts by initializing an empty queue SQ, which will hold the vehicles that are scheduled in CG in line 1. At the begining of each iteration of the outer loop in lines 3-5, we dequeue the next vehicle to be scheduled $vehicle \in VQ$ and initialize 2 new temporary queues called TSQ, TVQ. Then, in the the inner loop at lines 6-18, We repeatedly dequeue vehicles $prev \in SQ$ until SQ is empty and for each vehicle we check if $prev.priority < vehicle.priority$. If true, we erase u from it's scheduled slots in CG in line 10, and enqueue it in TVQ in line 9. Otherwise, in lines 11-16, we check if $prev$ is still on the road network, and add it to TSQ in line 13 if it is, if it's not we simply erase it from it's scheduled slots in line 15. Once SQ is empty we reach lines 18-22 where we obtain a state where all vehicles with priority lower than $v.priority$ are in TVQ at the order they where scheduled, and all others are in TSQ. Next we simply set $SQ \leftarrow TSQ$ to obtain the updated list of scheduled vehicles in SQ, we add all vehicles in TVG to the front of VQ in the same order, then schedule $vehicle$ in CG and add it to SQ.

Algorithm 2 schedule all vehicles entering the road network

Require: VQ, CG the queue of vehicles to schedule and the conflict graph

```
 1: SQ ← empty queue
 2: while VQ is not empty do
 3:     vehicle ← VQ.dequeue()
 4:     TSQ ←a new queue
 5:     TVQ ←a new queue
 6:     while SQ is not empty do
 7:         prev ← SQ.dequeue()
 8:         if prev.priority < vehicle.priority then
 9:             TVQ.enqueue(prev)
10:             CG.eraseScheduling(prev)
11:         else
12:             if prev is still on the road network then
13:                 TSQ.enqueue(prev)
14:             else
15:                 CG.eraseScheduling(prev)
16:             end if
17:         end if
18:     end while
19:     SQ ← TSQ
20:     concatinate TVQ to the front of VQ
21:     scheduleVehicle(vehicle, CG)
22:     SQ.enqueue(vehicle)
23: end while
```

Lemma 1. *For any vehicle v, $v \in SQ$ if and only if v is scheduled in CG.*

Proof. We begin by proving $v \in SQ \implies v$ is scheduled in CG.
Let $v \in SQ$ be some vehicle from SQ. v was either enqueued in line 22 or 13.
If v was enqueued in SQ at line 22, then v as scheduled in line 21 therefore v is scheduled in CG.
If v was enqueued in line 13, then v was dequeued from SQ in line 6, which means that v was inserted into SQ in a previous iteration. Since SQ was initialized as an empty queue, any item in SQ was enqueued in line 22 at some iteration, therefore v was scheduled in line 21. Next we prove v is scheduled in $CG \implies v \in SQ$.
Let v be some vehicle scheduled in CG.
The only way v was scheduled was by reaching line 21, it's trivial to see that any vehicle which reaches line 21 also reaches line 22 since there is no branching logic preventing it, therefore every vehicle scheduled in CG is also present in SQ.■

Lemma 2. *Any pair of vehicles $v, u \in SQ$ at indexes i, j respectively, which satisfy $v.priority = u.priority$. If v got dequeued from VQ before u then $i < j$.*

Proof. As previously stated, any vehicle $v \in SQ$ got enqueued in SQ at line 22 at the same iteration it got dequeued from VQ. Let $v \in SQ$ be a vehicles at index i, dequeued at iteration t and let $u \in SQ$ be a vehicle at index j dequeued a iteration $t < r$ which satisfies $v.priority = u.priority$. Suppose that $j < i$, this could only be possible in one of the two case:
1. u reached line 22 before v and therefore got enqueued in SQ before v.
2. v got dequeued from SQ at iteration r of the outer loop, and did not enqueue in TSQ at line 13.
Case 1 is impossible since we know that v got dequeued from VQ at an iteration $t < r$, and it's apparent that any vehicle that gets dequeued from VQ, get enqueued in SQ in the same iteration of the outer loop. Therefore, since $t < r$, v reached line 22 before u.
Regrading case 2, v may not be enqueued in TSQ if v either exited the road network, in which case it is irrelevant to the algorithm altogether, or if $v.priority < u.priority$ at iteration r of the outer loop. Since $v.priority = u.priority$ this is also false therefore case 2 is also impossible, which means i must be less than j. ■

Claim. Algorithm 2 produces an optimal schedule in GC for any given tuple of vehicles with respect to vehicle and start time priority.

Proof. Let v be a vehicle dequeued from VQ.
In lines 5 to 17, we remove all vehicles from SQ. Each vehicle removed from SQ gets inserted into VQ if it's priority is less than $v.priority$, otherwise it gets inserted into TSQ. By lemma 2, after line 18, SQ contains all vehicles that where dequeued from VQ before v and that have equal or higher priority to v.
By lemma 1, any vehicle not in SQ have it's slot reservations removed from CG.
Scheduling v on CG using $dwDijsktra$ algorithm gaurantees us an optimal schedule for v, as proved in [1] on first-come first-serve pricipal, meaning vehicles that where scheduled first will get their slots reserved before vehicles that came after them.
Therefore, scheduling v on this state of CG gaurantees an optimal schedule for v with respect to start time and priority.
Now, all vehicles that came before v and had lower priority, sit at the front of the queue, in the order they where dequeued from VQ, which means in the next iteration of the outer loop these vehicles will get dequeued before other vehicles that where already in VQ and came after them. Again, by optimality of $dwDijkstra$, we get an optimal scheduling for those vehicles as well, which means that algorithm 2 produces an optimal schedule with respect to start time and priority for all vehicles. ■

Definition 3. *Let n_n be the number of nodes in CG and let n_v be the number of vehicles scheduled in CG.*

Claim. Algorithm 2 schedules a vehicle in $O(n_n^2 \cdot n_v \cdot log(n_v))$ time in the worst case.

Proof. In the best case scenario, a vehicle v is scheduled once at line 21, and then exits the network before any vehicle of higher priority enters it, which means it won't be transfered back to VQ for scheduling.

In the worst case scenario, every time v gets scheduled, an iteration later a vehicle of higher priority gets scheduled which causes it to go back to VQ and get scheduled again. As this theoretically can last for ever in the case a continuous intermittent stream of emergency vehicles enter the road network, we will focus on a single iteration of the outer loop in which v gets scheduled once.
To schedule v once, the algorithm iterates over all currently scheduled vehicles and then calls $dwDijkstra$ to schedule v. Each enqueue and dequeue operation takes $O(log(n_v))$ time when implementing a priority queue using max-heap, therefore the iteration over all vehicles takes $O(n_v \cdot log(n_v))$ time.
Scheduling a single vehicle using $dwDijsktra$ takes $O(n_n^2 \cdot n_v \cdot log(n_v))$ time, as shown in [1] which means algorithm 2 schedules a vehicle in $O(n_n^2 \cdot n_v \cdot log(n_v) + n_v \cdot log(n_v))$ time.
This then gives us an overall runtime of $O(n_n^2 \cdot n_v \cdot log(n_v))$ ■

References

1. Hannah Yair, Shlomi Dolev, and Ehud Gudes. Amaze: Non-stop source to destination optimal vehicle scheduling. In *International Conference on Intelligent Traffic and Transportation*, 2024.

Fast Real-World Classification of ECH-enabled Applications

Yizhak Kahana[1,2,*], Yaakov (J) Stein[1], Gabriel Zigelboim[1], and Oren Glickman[2]
[1]*Allot*
[2]*Bar Ilan University*
[*]*correspondence should be addressed to ikahana@allot.com*

November 29, 2024

Abstract

Encrypted Client Hello is plugging information leakage in encrypted packets, in order to thwart Deep Packet Inspection. Various machine learning mechanisms have been proposed to classify ECH network connections, but previous research relied on unlimited classification time and mostly dealt with classifying a small number of classifications in highly constrained settings. We demonstrate highly accurate classification in no more than one second of a large number of applications for a real-network dataset, and propose a novel hierarchical mechanism that leverages protocol-specific knowledge.

1 Introduction

For various reasons, Service Providers need to know characteristics of the network traffic their networks are forwarding. Perhaps the most obvious reason is security, but visibility, QoS monitoring and assurance (including congestion mitigation), policy enforcement, differentiated charging, ensuring QoE levels, fault root cause analysis, user analysis for *upsell* opportunities, meeting regulatory and compliance requirements, and many other use cases are also important. And DPI is conventionally the heart of all systems that need to know characteristics of Internet traffic.

In the Internet protocol (IP) suite, packets are crafted to be self-describing, comprising a sequence of packet headers before the user content (e.g., the encoded audio). This structure facilitates standards-based recursive parsing by the intended recipient, but has the unintended consequence of revealing information about the packet to potentially malicious parties observing the packet on its way from source to destination. This situation has been rectified by adding various cryptographic mechanisms, and it is now often stated that the great majority of Internet traffic is encrypted.

However, this statement regarding the ubiquity of encryption actually only signifies that user content is encrypted, while Internet packet headers still include unencrypted metadata. This metadata can be exploited by Deep Packet Inspection (DPI) to identify the application being used.

DPI systems comprise carefully crafted rule sets that map logical combinations of metadata to applications. During operation, DPI associates packets to connections (sequences of packets between the same endpoints and sharing identifiers), and parses each individual packet exploiting the self-describing nature of IP packets to extract per-packet metadata. The DPI system then compares metadata collected from packets belonging to a connection to application signatures stored in an application signature rule-set. In this way, state-of-the-art DPI systems are able to recognize thousands of different applications from dozens of application categories, and accurately classify connections after observing a very small number of packets.

A notable (but far from the only) example of unencrypted fields are those in the TLS handshake. The TLS handshake is used by two parties wishing to securely communicate, before they have established their secure channel, and consists of a Client Hello (CH) message followed by a Server Hello (SH) message. The CH specifies the cyber suites the client supports, various crypto-related parameters, and optionally the name of the server (SNI) the client wants to reach (since there may be many virtual servers hosted behind a single IP address!). Since the cyber suite has not yet been agreed upon, the CH message is unencrypted, passing SNI and other information in the clear.

Over the past few years Internet traffic has become more fully encrypted, with sophisticated methods of encrypting overhead fields, including encrypted Client Hello (ECH) [1]. ECH, as its name implies, encrypts the TLS CH message using a public key previously obtained.

Encrypting metadata is an effective weapon in the battle against pervasive monitoring and revealing private information to malicious parties, but as collateral damage, it renders the job of DPI systems more complicated or even impossible. This directly impacts the aforementioned crucial functions for which DPI is employed, including security, visibility, QoS monitoring and assurance, etc.

However, even encrypting all packet header fields does not truly remove all distinctive metadata [2]. At least three per-packet characteristics remain. First, the sequence of directions of packets comprising the connection (e.g., whether from client to server or from server to client) may be indicative of the service type. For example, video streaming is characterized by a small number of request (GET) packets from client to server followed by a large number of response packets from server back to the client. On the other hand, voice or video calls are on the whole symmetric with approximately equal numbers of packets in each direction. Second, the size of the packets is generally preserved by metadata concealment techniques, or at most modified by the addition of additional header(s). Thus, even a fully encrypted connection can be characterized by a tell-tale sequence of packet sizes. For example, video streaming will consist of a large number of very large response packets from server to client, interspersed with small 'ACK' packets from client to server acknowledging their receipt. Thirdly, packet timings are generally not significantly perturbed by metadata obscuration, although they are influenced by the underlying latencies in the network. From here on we shall use the term PPI (Per Packet Information) to mean the time series of time, size, and direction of the initial packets in a connection.

There is a rich literature on how to use packet size and timing statistics to classify tens of different applications from a small number of application categories, and accurately classify connections after observing a relatively large number of packets.

However, these machine learning methods suffer from at least three drawbacks. First, they necessitate greater computational resources and consume more energy than DPI methods. Second, in contrast to DPI which classifies after observing a few packets, obtaining reliable statistics requires observing large numbers of packets, making these methods unsuitable for many scenarios (e.g., blocking). Third, they are limited to a small number of applications, as compared to the thousands of applications recognized by DPI.

Here we describe an alternative machine learning paradigm that avoids these drawbacks. Instead of statistics we directly use the PPI time series, without needing access to per-packet header field values. Classifying using only the PPI time series is known as *behavioral* classification. Only a short PPI time series, corresponding to a small number of packets, need be used. Furthermore, instead of building a single large classifier, we build a number of small classifiers, each model classifying a subset of applications. Since ECH leaves the IP layer visible, only applications that share the same server address are potentially confused and need to be distinguished using machine learning. These small models require short time series and have extremely high classification accuracy and recall.

2 Related work

In recent years, researchers have proposed using Machine Learning and Neural Network Learning models to deal with encrypted traffic classification. However, many of the conventional methods are based on the byte content of packets of the connection [3, 4, 5, 6]. These models learn patterns in unencrypted metadata, such as SNI, which will not be relevant once ECH is widely adopted. Another drawback in current studies is that the classification task was over a very few number of applications (12-19) [3, 4, 5, 6]. Many articles train and test their models on small limited data sets, old data sets such as the ISCX VPN/non-VPN dataset [7], or network simulations [8].

The main drawback of current methods of behavioral classification is lengthy classification times. FlowPic [9], for example, requires 30-second images and degrades with significantly shorter times. Studies that do attempt to minimize time to classification, tend to classify only a limited number of apps. For example, FlowFormers [10], which use attention-based Transformers to achieve high accuracy classification within 10 seconds, have only been demonstrated on 5-class problems. A recent paper from Huawei France [11] uses the PPI of the first 100 packets for TCP connections and 10 packets for UDP ones (unless the entire connection has fewer packets). Although this paper achieves approximately 90% accuracy for a large

number (the top 200) of applications, the analysis of 100 packets can be time-consuming, and the paper does not impose any time limit.

Despite the limitations mentioned above, new avenues for research are emerging with the availability of more relevant and extensive datasets. For example, the CESNET dataset [12, 13] provides a rich source of encrypted traffic data that has not been sufficiently embraced by other studies.

To the best of our knowledge, no comprehensive large-scale study has been carried out for a large number of applications with strongly limited classification time. As compared to the literature our approach innovates in three areas:

Operational Network Data Training and test data are collected from a large operational mobile network and labeled using a state-of-the-art commercial DPI system.

Real-Time Applicability Our approach supports immediate actions required in operational networks, by limiting PPI duration to 1 second.

Domain Knowledge Rather than blindly throwing a large machine learning model at the connection classification problem, we build a hierarchical structure of smaller classifiers based on clustering of server IP addresses.

3 The dataset format

Our aim is to identify the application supported by a network connection from among a large number of distinct applications and to accomplish this identification within stringent constraints — specifically ensuring that the total time elapsed from the reception of the first packet to the last does not exceed one second. Hence our data consists of network connections, each comprising connection identifiers (the so-called 5-tuple of 2 IP addresses, the layer-4 protocol number, and 2 TCP or UDP port numbers), up to 50 PPIs (but imposing the one-second constraint), and an application identifier (label). In order to preserve user privacy the IP addresses were anonymized using prefix-preserving CryptoPAN [14].

More precisely, for each connection we have the per-connection fields:

- time of first packet (not used for classification)
- obfuscated source IP address (not used for classification)
- obfuscated destination IP address
- layer 4 protocol number (UDP, TCP, other)
- source port number (not used for classification)
- destination port number (443/8443, 80/8080, other)
- AppID, a numeric value (label) identifying the application
- length of the PPI array (maximum of 50 packets)

and for each packet:

- packet arrival time relative to the initial packet in the connection (in milliseconds)
- packet direction (client-to-server or server-to-client)
- packet payload size (bytes in TCP/UDP payload)
- flag byte (see below).

The flag byte merits special attention. For TCP it contains the eight TCP flags, from which we use the SYN flag only for classification. For UDP packets it contains the first byte of the UDP payload, which for QUIC packets contains the 4 unencrypted bits (header format, fixed bit, long packet type field, etc.), and we utilize only the header format bit.

The AppID label is supplied by a state-of-the-art commercial DPI system, as will be discussed in the next subsection.

3.1 Data collection

Our dataset was constructed from live traffic observed in a large mobile network starting in September 2024. We divided the dataset into several subsets, each containing approximately four days of network traffic including both weekdays and weekends in order to ensure variability in network conditions. The first subset was designated for model training, while the rest were reserved for model testing, at several later dates.

Collection was performed by enhancing a commercial real-time Deep Packet Inspection (DPI) system operating in tap (non-inline) mode on an N*100Gbps bidirectional line. This DPI system is relatively fine-grained, distinguishing between sub-applications from the same provider. As an example, Facebook browsing, Facebook video, and Facebook video upload are considered three distinct applications.

The DPI system provides the AppID label based on a rule set which is updated biweekly. This DPI system has approximately 95% coverage (i.e., it does not recognize about 5% of the connections) and an extremely low misclassification rate, far surpassing simplistic labeling methods used in much of the literature, such as relying solely on Server Name Indication (SNI) or server IP addresses.

3.2 The raw data

To enable the reader to better envision our data we will describe the file selected for training. The connections sampled are selected by hashing the client IP address, thus enabling varying the sampling rate. The training data file encompassed 195 hours of traffic and contained 313 million raw connections, implying an average rate of 445 connections per second.

Of these connections

- 146 million (47%) were unencrypted DNS sessions,
- 31 million (10%) were not completely identified by the DPI system.

The fact that half of the connections were DNS is to be expected. Of the insufficiently specified connections over half were truly unknown applications (to be expected since the DPI coverage is on the order of 95%) and the rest were only partially identified. In any case such connections are uninteresting for training purposes and were removed, leaving 136 million nontrivial identified connections (194 per sec).

From previous research (based on querying multiple public DNS resolvers) we know that about 80% of popular server IP addresses correspond to essentially a single domain name. Here we observe a long tail of many less popular IP addresses that have a low probability of being hosted alone on an IPv4 address!

This cleaned data set was found to obey the following statistics:

- About 69% of the server IP addresses represent a single application;
  - 18% were observed to host two distinct applications;
  - 6% were observed to host three distinct applications.
- About 52% of the connections map to a server IP with *essentially* a single application (where *essentially* means that 95% of the connections were identified as belonging to this application);
  - 23% map to a server IP address with essentially two distinct applications;
  - 15% map to a server IP address with essentially three distinct applications.

So, about half of the sampled connections collected can be mapped to a unique application purely by the server IP address! Such connections are thus trivially classified and do not require further machine learning mechanisms.

In addition, while we observe more than 750 distinct applications in the training set, many of these are too rare to merit separate classification. We thus lumped together all applications that appeared in less than 0.05% of the connections into a single aggregate label *Other*.

The test sets were collected using the same setup and underwent identical clean-up, but were amassed one week, one month, and two months later.

A slightly restricted version of this dataset can be made available for academic research purposes. Interested parties should contact the corresponding author.

3.3 Applications

Of the 763 applications reported by the DPI in the training set, 115 passed the popularity criterion, namely were shared by at least 0.05% of the connections. Applications that attracted at least 1 % of connections are presented in Table 1.

GoogleServices	14.56 %	WhatsApp Transfer	2.08 %
TikTok	10.55 %	Google Search	2.01 %
Facebook	6.98 %	Advertisements	1.98 %
AppleServices	6.66 %	Computing sites	1.76 %
WhatsApp	4.37 %	Telegram	1.61 %
Instagram	3.79 %	YouTube	1.43 %
Analytics	3.33 %	iTunes	1.36 %
Google Play	3.33 %	Banners	1.29 %
Other	3.03 %	Snapchat	1.27 %
iCloud	2.90 %	Generic CDN	1.19 %
YouTube Browsing	2.22 %	Facebook Chat	1.14 %

Table 1: Most popular applications with their appearance percentages.

The reader will note that applications are identified by the DPI system in an *actioanable* manner. For instance, the most popular application, entitled GoogleServices, is actually an aggregate of multiple Google-provided services that are mostly transparent to the user, including push notifications, time services, website optimizations, translation, etc. AppleServices includes similar miscellaneous facilities related to Apple. Analytics and advertisements are what their name implies from many different servers.

Similarly, over 3% of the connections are labeled *Other*, which is not a true application but rather artificial aggregation of all the insufficiently popular ones.

Note that these application popularities are highly dependent on the network being a mobile network, and fixed networks would have different statistics. For example, Netflix accrues only about 0.1 % in this cellular network, while it is usually one of the more popular applications in fixed networks. Furthermore, application popularity is highly dependent on locality.

4 Model Training

We trained two types of classifier (a Random Forest and a Long Short-Term Memory (LSTM) neural network) and employed two stratagems (hierarchical and flat). Comparing the resulting four architectures enables assessing the effectiveness of conventional machine learning approaches versus deep learning methods, and the value of exploiting domain-specific knowledge.

4.1 Random Forest Classifier

For the Random Forest classifier, it was necessary to transform the variable-length PPI array into fixed-length feature vectors. We did this by zero-padding to the length of the longest sequence in the dataset, denoted as $L_{\max}$. The PPI components (time, direction, size, and flags) were then separated into individual feature vectors of length $L_{\max}$. The padded feature vectors are then concatenated with per-connection features (we used IP protocol number, destination port, and PPI length).

The Random Forest classifier was trained using the following 15 trees with a maximum depth of 35 and a random state of 42 (for reproducibility). The classifier was trained on the entire dataset without a separate validation split.

4.2 LSTM Neural Network

Since the Random Forest classifier does not fully capture temporal dependencies, we trained an LSTM neural network capable of processing variable-length sequences and learning temporal patterns inherent in the packet-level data.

The LSTM-based architecture leverages both per-connection and per-packet information:

1. **per connection Information** protocol number, destination port, PPI length
2. **per packet information** packet time, direction, size, and flags.

The *PPI* is first passed through a *Pack Padded Sequence* operation, to prepare it for processing by the LSTMs layers that handle sequences of varying lengths. The packed sequence is then fed into four stacked **LSTM** layers which capture temporal dependencies within the packet sequences. After processing by the LSTMs, the output is *unpacked*, and the final hidden state corresponding to the *last time step* of each sequence is extracted. This hidden state encapsulates the sequential information from the packet data.

The extracted features from the LSTM are then *concatenated* with the connection information, effectively combining sequential and non-sequential features. This combined feature vector is passed through two **fully connected layers**.

The model outputs class scores corresponding to different application IDs.

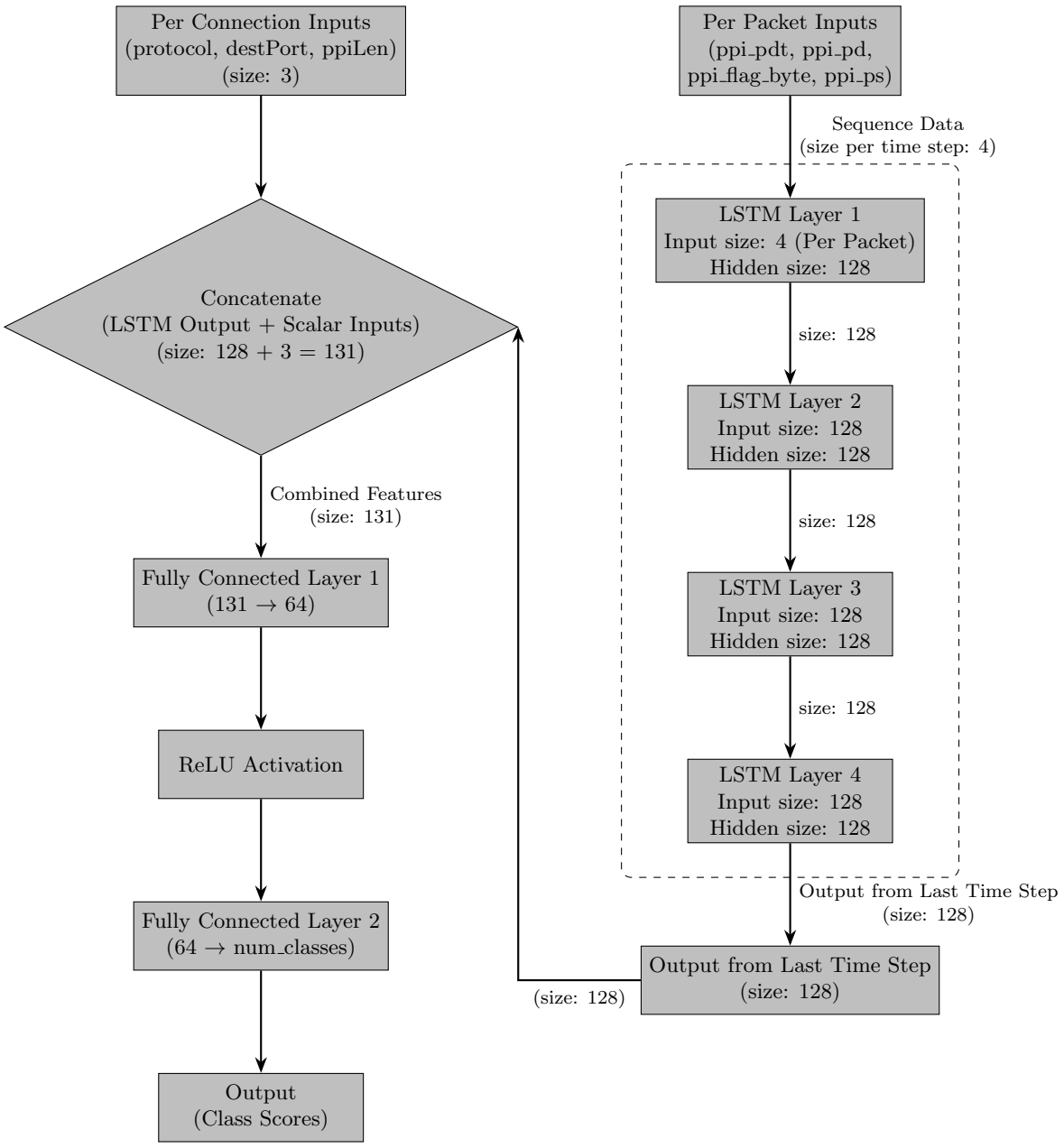


Figure 1: Architecture of the LSTM

The complete LSTM model (see Figure 1) has the following architecture:

- **LSTM Input Layer**: variable length PPI.
- **LSTM network**:
  - Input size: 4 (number of sequence features) times PPI length
  - Number of LSTM layers: 4
  - Number of LSTM units per layer: 128 units
  - Output size: 128
- **Fully Connected Layers**:
  - **Input Layer** per connection features (protocol number, destination port, PPI length).
  - **concatenation layer**
  - fully connected feedforward layer
    - Input size: 131 (128 from LSTM output + 3 scalar features)
    - Output size: 64
  - ReLU activation layer (size: 64)
  - fully connected feedforward layer
    - Input size: 64
    - Output size: C (the number of application classes)

The model was trained using the following settings:

- **Loss Function**: Cross-Entropy Loss, suitable for multi-class classification.
- **Optimizer**: Adam optimizer with an initial learning rate of 1×10^{-3}.
- **Learning Rate Scheduler**: StepLR scheduler reduces the learning rate by a factor of 0.5 every 5 epochs to facilitate convergence.
- **Epochs**: 20 epochs, determined empirically to balance training time and performance.
- **Batch Size**: 128 samples per batch.
- **Regularization**: Gradient clipping with a maximum norm of 5 to prevent exploding gradients.

Model performance was monitored using weighted precision, recall, and F1 metrics on a small validation set.

The model was trained using GPU acceleration.

5 Hierarchical Classification

In the current literature, the predominant method for classifying encrypted network traffic involves constructing a single, monolithic model that attempts to identify all applications. We'll call this the *flat classification* method. For example, in the previously cited Huawei study [11], a single large model is developed to categorize a multitude of applications by analyzing encrypted packet characteristics.

In contrast to the flat method, we introduce a hierarchical approach that incorporates knowledge of the server IP address, which remains unencrypted under ECH. We group network traffic based on server IP address and build unique models that distinguish between the relatively small number of applications specific to each group. This allows integrating domain-specific knowledge directly into the classification process. Instead of relying on a single monolithic model to classify all applications, we construct many smaller specialized models, each focusing on traffic associated with specific server IP addresses. Not only is it easier and faster to train these smaller models, but classification errors are less severe as misclassification is only possible between related applications.

The first step in our hierarchy singles out server IP addresses that were identified in the training set as hosting essentially a single application. A server is deemed to host *essentially* a single application if the

proportion of training connection associated with its dominant application exceeds 95%. For such server IP addresses, classification is trivial.

The next level of our hierarchy locates sets of popular server IP addresses that host multiple applications. We may partition such server IP addresses into distinct groups that share the same applications, using one of three methods:

1. ASN-based grouping
2. application-based grouping
3. application distribution clustering

Connections with server IP addresses not in a defined group are classified using a flat model.

ASN-based grouping maps server IP addresses to their Autonomous System Numbers (ASNs) to leverage the fact that servers within the same Autonomous System are likely to host applications related to the entity owning that AS. This is especially the case where the AS belongs to a *gorilla*, that is, one of the dominant technology giants such as Google, Meta, Amazon, Apple, or Microsoft. Thus, a server IP address belonging to Google's ASN 15169, might host GoogleServices, Google Search, Gmail, Google Maps, Youtube, etc., but would not support Facebook, Instagram, or WhatsApp, which are hosted in Meta's ASN 32934. The disadvantage of this grouping is that a gorilla may subdivide its IP address resources, not hosting all services on all of its addresses.

Application-based grouping partitions servers based on a particular application constituting a significant portion of their activity. All server IP addresses with the same dominant applications are grouped together. Application distribution clustering compares the probability distributions and performs clustering based on the distribution similarity. These grouping methods will be more fully explained in a future publication. The overall performance did not significantly differ depending on the grouping method.

Here we will report on ASN-based grouping, wherein we map popular server IP addresses to their ASN using publicly available information. In our training set we found 11 major ASN groups (each encompassing over 1% of the connections) with between 10 and 30 applications and between 300 and 4000 server IP addresses per group. As expected, ASN groups included Google, Apple, Amazon, Microsoft, Akamai, Cloudflare, and several CDNs. For each ASN group we train a separate classification model that classifies only applications found in that group. Connections belonging to server IP addresses that are not in a major ASN group are classified using a flat model trained on the complete dataset comprising all applications.

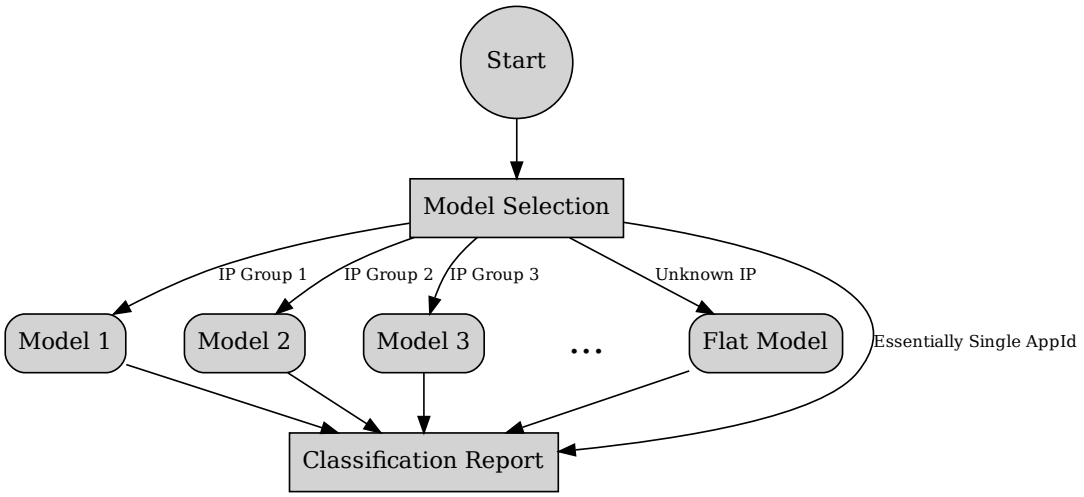


Figure 2: Hierarchical Decision Flow

6 Results

Here compare our hierarchical classification scheme to a flat classifier that does not exploit server IP address at all. We have additionally explored the flat model with server IP address as input; although this improved the model's accuracy, its overall performance was still significantly inferior to that of our proposed hierarchical method.

First, two classifiers, a Random Forest classifier and a deep LSTM, were trained with up to 50 packets per connection, using the flat model and our hierarchical model (with ASN-based grouping). The performance on a test set collected a week after the training are presented in Table 2.

In order to reduce the time to classification, we explored the same methods but with a stricter limit of no more than 10 packets per connection. The performance metrics suffered, but not very significantly, as shown in Table 3.

In a rapidly evolving network landscape, models trained on data from a specific time frame may experience performance degradation when applied to data collected at a later date. Factors such as updates to existing services and changes in user behavior can alter the statistical properties of network traffic, leading to distributional drift.

To assess the time stability of model we conducted a drift evaluation using two additional datasets collected a month later and two months later. Results are presented in Tables 4, 5, 6 and 7. We observe that our hierarchical model maintained close to 90% accuracy, recall, and F1-score, even after two months. Assessment of geographical stability will be reported elsewhere.

7 Discussion

The introduction of ECH strongly impacts DPI systems, and requires the introduction of more sophisticated machine learning techniques. Previous research has mostly ignored the real-world issues of time-to-classification (allowing 30 seconds or a minute before classifying) and the large number of potential applications (typically assuming tens of applications suffice).

We have presented what we believe to be the first research into replacing DPI in real-world settings to counter the influence of ECH. We find that, based solely on packet size, direction and time of arrival, we can classify over 100 applications in less than 1 second, with accuracy of approximately 90%. Indeed, even a mere ten packets suffice for high-accuracy classification. We find little advantage to using deep LSTM classifiers as compared to legacy Random Forest ones.

We furthermore proposed a hierarchical model that integrates domain-specific knowledge through the utilization of server IP addresses, which are not hidden in the ECH scenario. This hierarchical approach not only reduces computational complexity of training and inference, but noticeably improves performance as compared to the flat model. Furthermore, any residual errors are limited to within server IP groups, which are less impactful in practice.

Our methods demonstrate robust generalization when tested on a real-world mobile network, and do not degrade significantly after a month or even two months have gone by.

Table 2: Performance Comparison with up to 50 PPIs

Classification Method	Model	Precision	Recall	F1 Score
Flat Model	Random Forest	88.42	88.26	88.07
	LSTM	90.06	89.93	89.84
ASN-Based Grouping	Random Forest	93.58	93.54	93.42
	LSTM	94.68	94.64	94.57

Table 3: Performance Comparison with up to 10 PPIs

Classification Method	Model	Precision	Recall	F1 Score
Flat Model	Random Forest	88.15	88.11	87.92
	LSTM	86.34	86.35	86.05
ASN-Based Grouping	Random Forest	92.71	92.71	92.58
	LSTM	92.15	92.12	91.97

Table 4: Performance Comparison with up to 50 PPIs one month after training

Classification Method	Model	Precision	Recall	F1 Score
Flat Model	Random Forest	86.62	86.21	85.95
	LSTM	88.16	87.81	87.64
ASN-Based Grouping	Random Forest	92.16	92.01	91.85
	LSTM	93.13	93.05	92.93

Table 5: Performance Comparison with up to 10 PPIs one month after training

Classification Method	Model	Precision	Recall	F1 Score
Flat Model	Random Forest	86.21	86.00	85.70
	LSTM	84.19	84.11	83.65
ASN-Based Grouping	Random Forest	91.10	91.05	90.85
	LSTM	90.25	90.23	89.98

Table 6: Performance Comparison with up to 50 PPIs two months after model training

Classification Method	Model	Precision	Recall	F1 Score
Flat Model	Random Forest	82.03	80.79	80.38
	LSTM	83.19	82.22	81.90
ASN-Based Grouping	Random Forest	90.48	90.31	90.06
	LSTM	91.27	91.17	90.98

Table 7: Performance Comparison with up to 10 PPIs two months after training

Classification Method	Model	Precision	Recall	F1 Score
Flat Model	Random Forest	81.43	80.34	79.95
	LSTM	79.25	78.15	77.59
ASN-Based Grouping	Random Forest	89.63	89.58	89.35
	LSTM	88.75	88.69	88.43

References

[1] Eric Rescorla, Kazuho Oku, Nick Sullivan, and Christopher A. Wood. TLS Encrypted Client Hello. Internet-Draft draft-ietf-tls-esni-22, Internet Engineering Task Force, September 2024. Work in Progress.

[2] Martino Trevisan, Francesca Soro, Marco Mellia, Idilio Drago, and Ricardo Morla. Attacking doh and ech: Does server name encryption protect users' privacy? *ACM Transactions on Internet Technology*, 23(1):1–22, 2023.

[3] Weiping Zheng, Jianhao Zhong, Qizhi Zhang, and Gansen Zhao. Mtt: an efficient model for encrypted network traffic classification using multi-task transformer. *Applied Intelligence*, 52(9):10741–10756, 2022.

[4] Hong Ye He, Zhi Guo Yang, and Xiang Ning Chen. Pert: Payload encoding representation from transformer for encrypted traffic classification. In *2020 ITU Kaleidoscope: Industry-Driven Digital Transformation (ITU K)*, pages 1–8. IEEE, 2020.

[5] Peng Lin, Yishen Hu, Yanying Lin, Kejiang Ye, and Cheng-Zhong Xu. Pean: A packet-level end-to-end attentive network for encrypted traffic identification. In *2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*, pages 267–274. IEEE, 2021.

[6] Jee-Tae Park, Chang-Yui Shin, Ui-Jun Baek, and Myung-Sup Kim. Fast and accurate multi-task learning for encrypted network traffic classification. *Applied Sciences*, 14(7):3073, 2024.

[7] Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A Ghorbani. Characterization of encrypted and vpn traffic using time-related. In *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*, pages 407–414, 2016.

[8] Siddhant Ray. Advancing packet-level traffic predictions with transformers. Master's thesis, ETH Zurich, Networked Systems, 2022.

[9] Tal Shapira and Yuval Shavitt. Flowpic: Encrypted internet traffic classification is as easy as image recognition. In *IEEE INFOCOM 2019-IEEE conference on computer communications workshops (INFOCOM WKSHPS)*, pages 680–687. IEEE, 2019.

[10] Rushi Babaria, Sharat Chandra Madanapalli, Himal Kumar, and Vijay Sivaraman. Flowformers: Transformer-based models for real-time network flow classification. In *2021 17th International Conference on Mobility, Sensing and Networking (MSN)*, pages 231–238. IEEE, 2021.

[11] Lixuan Yang, Alessandro Finamore, Feng Jun, and Dario Rossi. Deep learning and zero-day traffic classification: Lessons learned from a commercial-grade dataset. *IEEE Transactions on Network and Service Management*, 18(4):4103–4118, 2021.

[12] Jan Luxemburk, Karel Hynek, Tomáš Čejka, Andrej Lukačovič, and Pavel Šiška. Cesnet-quic22: A large one-month quic network traffic dataset from backbone lines. *Data in Brief*, 46:108888, 2023.

[13] Jan Luxemburk and Tomáš Čejka. Fine-grained tls services classification with reject option. *Computer Networks*, 220:109467, 2023.

[14] Jun Xu, Jinliang Fan, Mostafa H. Ammar, and Sue B. Moon. On the design and performance of prefix-preserving ip traffic trace anonymization. In *Proceedings of the First ACM SIGCOMM Workshop on Internet Measurement*. ACM, 2001.

Bloom Filter Look-Up Tables for Private and Secure Distributed Databases in Web3

(Preliminary Version)

Shlomi Dolev[1], Ehud Gudes[1], and Daniel Shlomo[1]

Department of Computer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel
dolev@cs.bgu.ac.i, gudes@bgu.ac.il, danshl@post.bgu.ac.il

Abstract. The rapid evolution of decentralized systems in the Web3 ecosystem introduces critical challenges, particularly in ensuring data security, privacy, and scalability. Traditional systems rely on centralized architectures, which are vulnerable to breaches and single points of failure. Decentralized databases, while promising enhanced privacy and autonomy, face unique issues such as key management, concurrent updates, and resilience against node failures.
This research proposes a novel decentralized database framework utilizing a Bloom Filter-based Look-Up Table (BFLUT). The system addresses the secure management and retrieval of cryptographic keys without explicit storage, even in the face of compromised nodes. By leveraging IPFS, OrbitDB, and the InterPlanetary Name System (IPNS), this framework ensures efficient, consistent, and secure data distribution tailored to Web3 applications.

1 Introduction: System Purpose and General Structure

The proposed system aims to create a decentralized and secure database environment for Web3, enabling efficient, scalable, and privacy-preserving data management. The primary objective is to organize the database into key-value pairs, where each value represents a file stored on a unique node in the decentralized network. By using these distributed files, the system securely stores cryptographic keys by activating specific bits, ensuring privacy and resilience against node failures.

1.1 Overview of BFLUT

At the core of the proposed system lies the **Bloom Filter Look-Up Table (BFLUT)** mechanism. BFLUT is a probabilistic data structure designed to encode and retrieve data efficiently without explicitly storing sensitive information. It combines the properties of Bloom Filters, which allow compact and fast lookups, with a structured approach for securely storing keys. identical figure can be found in [ref] the figure appears here for the sake of readability.

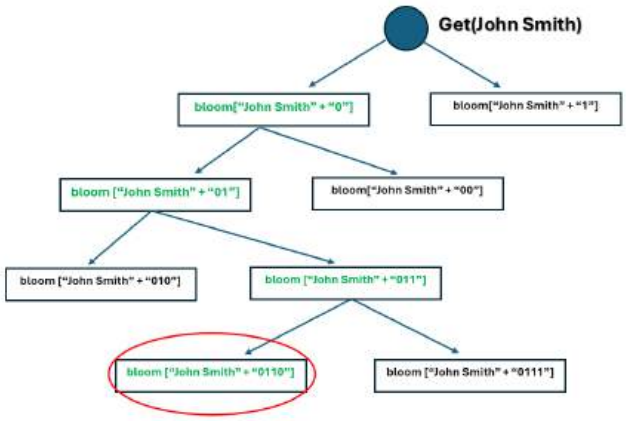


Fig. 1. Example of BFLUT operation: Key lookup for "John Smith"

The diagram in Figure illustrates how BFLUT utilizes hierarchical hashing to securely encode and retrieve keys.

Identical figure can be found in [3] the figure appears here for the sake of readability.

2 Data Insertion Process

When inserting a new key, the system performs the following steps:

1. **Generate a Unique Key:** A new key is created based on the user's input (e.g., username and password). These inputs are combined with a prefix of the key and hashed to ensure secure encoding.
2. **Divide Key into Prefixes:** The key is divided into sequential prefixes. For example, the key "1011" is broken into "1", "10", "101", and "1011".
3. **Calculate Hash for Prefixes:** Each prefix, combined with the username and password, is hashed using a secure hashing function to determine the locations of the bits to activate in the Bloom Filter.
4. **Locate and Activate Bits:** For each hashed prefix, the system uses OrbitDB's decentralized lookup mechanism to identify the nearest file in the network. The file is then retrieved from its node, and the relevant bits corresponding to the prefix's hash are activated, securely encoding the key's presence without directly storing it.
5. **Update File on the Network:** After updating the bits, the file is saved back to IPFS, and its IPNS reference is updated to point to the new version.

3 Search Process

When searching for a key, the system performs the following steps:

1. **Generate Hashed Prefixes:** The user inputs their key, which the system breaks into prefixes. For each prefix, a hash value is calculated using a secure hashing function.
2. **Match Prefixes with Bloom Filter:** The hash values of the prefixes are checked against the bits activated in the Bloom Filter. If the corresponding bits for a prefix are active, the system proceeds to the next prefix; otherwise, the key is considered invalid.
3. **Locate File with Key's Data:** For prefixes that match, the system uses OrbitDB to locate the nearest file on the network based on hash proximity. This file holds the information related to that key's segment.
4. **Reconstruct the Key's Data:** By retrieving all the matching file segments from the distributed nodes, the system reconstructs the complete data for the requested key. This ensures secure and accurate retrieval without exposing sensitive information.

4 Conclusion

The proposed framework effectively addresses critical challenges in decentralized systems, providing a secure, private, and scalable solution for Web3 applications. By integrating advanced technologies such as OrbitDB, IPFS, and BFLUT, the system enhances key management, data consistency, and overall resilience. By distributing files across multiple nodes and ensuring secure key retrieval without explicit storage, the approach establishes a robust and practical foundation for managing data in decentralized environments.

References

1. Alice Brown and Bob Green. Secure key management in distributed systems: Challenges and solutions. *IEEE Transactions on Cloud Computing*, pages 1–12, 2023.
2. John Doe and Jane Smith. Secure and effective key management using secret sharing schemes in cloud computing. *International Journal of Cloud Applications and Computing (IJCAC)*, 10(3):45–63, 2020.
3. Shlomi Dolev, Ehud Gudes, Erez Segev, Jeffrey Ullman, and Grisha Weintraub. Bflut bloom filter for private look-up tables. In *Proceedings of the CSCML 2022*, volume LNCS 13301, pages 499–505. Springer Nature Switzerland AG, 2022.

LLM-Based De-Anonymization (Prevention)*

(Preliminary Version)

Shlomi Dolev and Rie Ruash

Department of Computer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel

Abstract. The widespread sharing of publicly available datasets has amplified privacy challenges, particularly when anonymized data remains vulnerable to re-identification through sophisticated de-anonymization techniques. These methods exploit dataset characteristics, posing significant risks despite advancements in anonymization practices. This research leverages Large Language Models (LLMs) to address these challenges by fine-tuning them on de-anonymization methods. It proposes a framework where finetuned LLMs analyze the characteristics of new datasets, identify similarities with previously studied datasets, and recommend effective de-anonymization techniques from existing research. By integrating LLM capabilities with a similarity-based approach, this work provides a scalable and intelligent solution for enhancing privacy risk assessments and understanding data vulnerabilities. In particular, a data set can be automatically verified to be resistant to known (and inspired by known) re-identification techniques.

Keywords: Data Privacy, De-anonymization, Machine Learning , Large Language Models(LLMs), Retrieval-Augmented Generation (RAG).

1 Introduction

The rapid digitization of data and the proliferation of available public datasets across different domains have created new opportunities for research and innovation. However, this surge in data sharing comes with significant privacy challenges, particularly when datasets include sensitive information such as personal identifiers, healthcare records, or proprietary details. To address these concerns, different anonymization techniques have been developed to remove or obfuscate identifiable information, making it possible to publish and share data more safely in compliance with regulations such as the General Data Protection Regulation (GDPR) and the Health Insurance Portability and Accountability Act (HIPAA). Despite these efforts, achieving an optimal balance between privacy preservation and data utility remains an ongoing challenge.

While advances have been made in anonymization techniques, their limitations become evident when faced with sophisticated de-anonymization methods. De-anonymization, or the process of re-identifying individuals within anonymized datasets, exploits auxiliary data sources, statistical correlations, and computational models to infer sensitive information. Quasi-identifiers—seemingly harmless dataset attributes — can often be cross-referenced to infer identities, exposing anonymized data to privacy breaches. Advancements in computational power and algorithmic sophistication have further intensified these risks, undermining the effectiveness of traditional anonymization strategies.

However, The success of de-anonymization techniques is not universal, as it highly depends upon the specific characteristics of the datasets to which they are applied. Factors such as data sparsity, dimensionality, and the presence of quasi-identifiers influence the success of re-identification efforts. These challenges highlight the need for context-aware approaches that consider dataset-specific attributes, in order to evaluate and mitigate privacy risks effectively.

Large Language Models (LLMs), such as GPT-4, BERT, and LLaMA, have emerged as powerful tools with impressive capabilities in natural language understanding, generation, and pattern identification. LLMs have proved exceptional useful in domains ranging from medical research to legal analysis, making them invaluable for investigating multi-faceted problems where contextual understanding is needed. Their ability to process and analyze vast amounts of unstructured data enables

* Contact author: Rie Ruash, rie@post.bgu.ac.il. The research is partially supported by the Rita Altura trust chair in computer science, the Israeli Smart Transportation Research Center (ISTRC), and the Israeli Science Foundation (Grant No. 465/22).

them to uncover intricate patterns, synthesize complex concepts, and offer insights that were previously unattainable. This potential makes them an ideal tool for bridging the gap between theoretical insights into de-anonymization and practical solutions for privacy preservation in data sharing.

2 Research Target

The research goal is to propose a novel framework that leverages fine-tuned LLMs to evaluate the applicability of de-anonymization techniques to new datasets. By analyzing the new dataset's characteristics within a similarity-based framework, the LLM can assess which previously studied datasets share resembling characteristics. This allows for the identification of de-anonymization methods that might be effective, offering a data-driven, context-aware approach to privacy risk assessment.

This methodology not only underscores the adaptability and versatility of LLMs but also provides a systematic means to bridge theoretical advancements and practical application in privacy research. By leveraging the capabilities of LLMs, this research aims to create a scalable and intelligent framework that enhances our ability to address privacy challenges in the era of big data.

3 Related Work

De-anonymization, or re-identification, remains a critical challenge in data privacy, with research exposing the vulnerabilities of anonymized datasets. Prior studies have explored de-anonymization techniques, emphasizing how dataset characteristics influence their effectiveness, while large language models (LLMs) have introduced transformative approaches to both anonymization and de-anonymization. This section examines these methods, focusing on how dataset characteristics impact de-anonymization success and the role of LLMs in addressing privacy challenges.

One prominent example of de-anonymization demonstrated the weaknesses of traditional anonymization by re-identifying users in the Netflix Prize dataset via linkage with their movie rating patterns [14]. Datta et al.[4] provided formal proofs explaining the effectiveness of such attacks on high-dimensional microdata. A decade later, a retrospective analysis emphasized how advancements in data collection and computational techniques have increased the complexity and effectiveness of de-anonymization, particularly in high-dimensional datasets [15].

Gambs et al.[7] introduced a de-anonymization method using the Mobility Markov Chain (MMC) model for geolocated data , while Ji et al.[9], reviewed techniques for graph-based datasets. Other studies have explored different de-anonymization techniques for specific types of datasets, including mobile sensor data [10] and online health posts [8], showing how de-anonymization methods can be tailored to the structural characteristics of various datasets. Torres and Olivares [22] extended these findings by applying probabilistic record linkage techniques to high-dimensional datasets like movies and books, illustrating the variation of de-anonymization efficiency across diverse dataset structures.

Al-Azizy et al.[2] classified de-anonymization techniques based on auxiliary information and dataset structure. Their work covers various approaches, with a primary focus on traditional methods, emphasizing that feature-matching techniques are more effective for datasets with similar features, like different geolocated databases, while graph-based techniques excel in social network datasets. This classification underscores the critical role of dataset characteristics in determining the success of de-anonymization attacks.

The advent of large language models (LLMs), such as GPT-4 and LLaMA, has introduced new avenues in both the fields of anonymization and de-anonymization. In the context of anonymization, Pissarra et al.[19] demonstrated how LLMs can effectively anonymize clinical text, while maintaining contextual coherence and protecting sensitive patient information. Staab et al.[21] proposed an LLM-based adversarial anonymization framework that leverages the strong inferential capabilities of LLMs to guide the anonymization process, and Dou et al.[5] explored LLMs' role in reducing privacy risks in online self-disclosures. Building on these efforts, Yang et al.[23] introduced a framework which focuses on balancing privacy protection with data utility for downstream tasks while Frikha et al.[6] developed IncogniText, an LLM-based text anonymization method using private attribute randomization to mislead adversaries.

In the realm of de-anonymization, LLMs have demonstrated significant potential to exploit vulnerabilities in anonymized data. Nyffenegger et al.[16] explored this risk by evaluating the re-identification

capabilities of LLMs on anonymized court rulings and a masked Wikipedia dataset. Their findings highlight how factors like model size, input length, and instruction tuning influence the ability of LLMs to perform re-identification, demonstrating the critical interplay between dataset characteristics and the effectiveness of anonymization. Another work introduced a framework, named DIRI, which uses LLM to re-identify the patient who corresponds to a redacted clinical note [13]. Similarly, Patsakis and Lykousas [18] investigated how LLMs can deanonymize texts anonymized by the Textwash tool, revealing that LLMs outperform humans in identifying anonymized individuals. Staab et al.[20], who also contributed to anonymization strategies [21], showcased the ability of well-trained LLMs to deduce personal attributes from text, such as location and income.

This study builds on advancements in de-anonymization techniques by leveraging fine-tuned large language models (LLMs) to tackle modern privacy challenges. It introduces a novel framework where LLM, trained on prior de-anonymization studies, analyzes dataset characteristics to recommend effective de-anonymization technique for a new dataset. By bridging dataset-specific vulnerabilities with targeted strategies, this approach advances de-anonymization research, and offers new insights into addressing privacy risks in the era of powerful AI technologies. In particular, the unsuccessful application of our de-anonymization technique serves as a verification for a given dataset.

4 Methodology

To achieve the objective of this research, the proposed framework methodology consists of the following steps:

- **Collection of a comprehensive dataset of research papers:** The research papers focus on de-anonymization techniques and utilize publicly available or well-known datasets. This step involves systematic keyword-based searches on platforms such as ScienceDirect, Google Scholar, and other relevant databases. Keywords may include “de-anonymization“, “re-identification“ and “anonymity breach“.
- **Fine-tuning a Large Language Model (LLM) using a Retrieval-Augmented Generation(RAG) framework:** The fine-tuned model will analyze the collected papers, focusing on:
  - Extracting the methodological steps of the de-anonymization techniques described in the papers.
  - Identifying the details of the dataset(s) used in the paper, such as the data modality (e.g., tabular, textual, or image) and the reported evaluation metrics of the de-anonymization.
  - Associating each methodology with its corresponding evaluation dataset(s) used in the paper. For datasets corresponding to multiple papers, the framework will prioritize the most effective de-anonymization technique reported it.
- **Similarity based analysis for a given new dataset:** Evaluating the analysis by comparing it with previously studied datasets, and identifying the most relevant dataset based on this similarity.
- **Recommending the most suitable de-anonymization methodology:** Recommendation based on the identified similarities. The framework will retrieve and provide the methodological steps of the de-anonymization attack associated with the most similar dataset, ensuring a practical, data-driven, context-aware recommendation.
- **Validate the framework:** Validation by testing its recommendations on new datasets with known characteristics.

5 Experiments

As part of this ongoing research, we conducted two initial experiments to establish a foundation for the proposed framework. These experiments were designed to explore the feasibility of the approach and provide preliminary insights. While they represent an important starting point, further experiments are required to refine the methodology. This section provides a brief overview of the design, implementation, and early findings of these experiments.

5.1 Extracting Methodology from a De-Anonymization Paper

In this experiment, we evaluated the ability of a fine-tuned LLM to extract the methodological steps of a de-anonymization technique from a PDF version of the research paper by Lee et al.[11]. To recover the methodological details of the attack, we used Meta's LLaMA 3.1 [1] integrated with a Retrieval-Augmented Generation (RAG) framework and fine-tuned the model on the paper's content. We posed targeted questions to the methodology and the model successfully extracted from the paper key steps of the de-anonymization process, such as data collection and feature extraction, as well as the machine learning optimization using a PRF-SVM model.

The figure below (Fig. 1) presents the interaction, showing the queries we used and the model's detailed responses. The queries and the responses are also documented in **Appendix A**.

Fig. 1. Experiment 1- extracting methodology steps described in a de-anonymization research paper, using fine-tuned LLaMA

The results demonstrate the framework's ability to accurately analyze and extract the methodological details of a de-anonymization method from a provided research paper.

5.2 Datasets Similarities

In this preliminary experiment, we aim to demonstrate the feasibility of comparing two datasets to assess whether the same de-anonymization attack methodology would be effective on both. The experiment investigates structural and statistical similarities between the datasets and serves as an important step in our framework development.

For this purpose, we used a synthesized dataset, referred to hereafter as the "Synthesized Adults Dataset", similar to the Adult Database from the UCI Machine Learning Repository [3]. The Adult database was also used in one of the experiments in the paper by Machanavajjhala et al.[12], where they found vulnerabilities in the k - anonymized version of it, using a proposed Homogenity Attack. The synthesized dataset we used in this experiment is with no missing values and was created using OpenAI's ChatGPT-4o [17], with $k = 5$ anonymity applied to anonymize the data. The synthesized dataset contains 101 records and 5 attributes: `age`, `marital-status`, `occupation`, `race` and `salaryclass` as sensitive attribute. We successfully evaluated the Homogenity Attack, following the principles outlined in the paper by Machanavajjhala et al.[12], in order to verify the vulnerability of this dataset.

We created a comparable dataset, referred to hereafter as the "Health-Sports Dataset", by modifying the columns names to health and sports related terms and replacing each unique value in the original synthesized dataset with another unique value, ensuring that the mapping was consistent. This process preserved the structure and the statistical distribution of the data, which allowed us to test whether the de-anonymization attacks could generalize across datasets with similar characteristics. Subsequently, we applied $k = 5$ anonymity to the dataset.

In order to design the analysis, we consulted ChatGPT-4o (version from December 12, 2024), [17], which suggested a systematic approach for comparing the two datasets. The interaction is detailed in **Appendix B** and outlines the rationale and methodology for assessing the datasets' similarities. Specifically, ChatGPT-4o [17] proposed evaluating four key metrics: unique combinations ratio, average feature correlation (using Cramér's V correlation, since there are several categorical features), feature-level entropy, and cluster density via KMeans clustering. Following the proposed analysis, we implemented the code and verified the results. The results are presented in Table 1.

Metric	**Anonymized Dataset Version**	**Value**
Unique Combinations Ratio	anonymized Synthesized Adults	0.31
	anonymized Health-Sports	0.31
Average Feature Correlation	anonymized Synthesized Adults	0.41
	anonymized Health-Sports	0.41
Feature-Level Entropy	anonymized Synthesized Adults	age: 0.42, marital-status: 1.09, occupation: 1.55, race: 1.38, salaryclass: 0.69
	anonymized Health-Sports	age_group: 0.42, activity: 1.09, favorite-sport: 1.55, smoking-status: 1.38, health: 0.69
Cluster Density	anonymized Synthesized Adults	[0.48, 0.3, 0.23]
	anonymized Health-Sports	[0.41, 0.33, 0.27]

Table 1. Experiment 2- comparison of the metrics

As detailed in the table, the unique combinations ratios (≈ 0.31 %) was identical for both datasets, which indicates similar levels of redundancy. The average feature correlation was ≈ 0.41 in both datasets, reflecting comparable interdependencies between features. Feature-level entropy values were also consistent. For example, the feature `age` in the anonymized Synthesized Adults Dataset and the feature `age_group` in the anonymized Health-Sports Dataset both exhibited an entropy of ≈ 0.42 . Finally, the cluster density distributions (while showing minor variations) remained broadly aligned, with anonymized Synthesized Adults Dataset having densities of [≈ 0.48 , ≈ 0.30 ,≈ 0.23] and the anonymized Health-Sports Dataset having densities of [≈ 0.41 , ≈ 0.33 , ≈ 0.27].

These findings strongly indicate that these two datasets are structurally and statistically similar, suggesting that de-anonymization methodology effective on one dataset is likely to be effective on the other dataset as well. This experiment highlights the utility of leveraging large language models (LLMs) for generating systematic analyses and validates the proposed methodology through application and results verification.

References

1. Meta AI. Llama 3.1: Large language model meta ai. https://www.llama.com/, 2024. Accessed August 12, 2024.
2. Dalal Al-Azizy, David Millard, Iraklis Symeonidis, Kieron O'Hara, and Nigel Shadbolt. A literature survey and classifications on data deanonymisation. In *Risks and Security of Internet and Systems: 10th International Conference, CRiSIS 2015, Mytilene, Lesbos Island, Greece, July 20-22, 2015, Revised Selected Papers 10*, pages 36–51. Springer, 2016.
3. Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.
4. Anupam Datta, Divya Sharma, and Arunesh Sinha. Provable de-anonymization of large datasets with sparse dimensions. In *Principles of Security and Trust: First International Conference, POST 2012, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2012, Tallinn, Estonia, March 24-April 1, 2012, Proceedings 1*, pages 229–248. Springer, 2012.
5. Yao Dou, Isadora Krsek, Tarek Naous, Anubha Kabra, Sauvik Das, Alan Ritter, and Wei Xu. Reducing privacy risks in online self-disclosures with language models. *arXiv preprint arXiv:2311.09538*, 2023.

6. Ahmed Frikha, Nassim Walha, Krishna Kanth Nakka, Ricardo Mendes, Xue Jiang, and Xuebing Zhou. Incognitext: Privacy-enhancing conditional text anonymization via llm-based private attribute randomization. *arXiv preprint arXiv:2407.02956*, 2024.
7. Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. De-anonymization attack on geolocated data. *Journal of Computer and System Sciences*, 80(8):1597–1614, 2014.
8. Shouling Ji, Qinchen Gu, Haiqin Weng, Qianjun Liu, Pan Zhou, Jing Chen, Zhao Li, Raheem Beyah, and Ting Wang. De-health: all your online health information are belong to us. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 1609–1620. IEEE, 2020.
9. Shouling Ji, Prateek Mittal, and Raheem Beyah. Graph data anonymization, de-anonymization attacks, and de-anonymizability quantification: A survey. *IEEE Communications Surveys & Tutorials*, 19(2):1305–1326, 2016.
10. Nicholas D Lane, Junyuan Xie, Thomas Moscibroda, and Feng Zhao. On the feasibility of user de-anonymization from shared mobile sensor data. In *Proceedings of the Third International Workshop on Sensing Applications on Mobile Phones*, pages 1–5, 2012.
11. Wei-Han Lee, Changchang Liu, Shouling Ji, Prateek Mittal, and Ruby B Lee. Blind de-anonymization attacks using social networks. In *Proceedings of the 2017 on Workshop on Privacy in the Electronic Society*, pages 1–4, 2017.
12. Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. *L*-diversity: Privacy beyond *k*-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1):3, 2007.
13. John X Morris, Thomas R Campion, Sri Laasya Nutheti, Yifan Peng, Akhil Raj, Ramin Zabih, and Curtis L Cole. Diri: Adversarial patient reidentification with large language models for evaluating clinical text anonymization. *arXiv preprint arXiv:2410.17035*, 2024.
14. Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125. IEEE, 2008.
15. Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets: a decade later. *May*, 21:2019, 2019.
16. Alex Nyffenegger, Matthias Stürmer, and Joel Niklaus. Anonymity at risk? assessing re-identification capabilities of large language models. *arXiv preprint arXiv:2308.11103*, 2023.
17. OpenAI. Chatgpt-4o by openai (december 12, 2024 version). https://chat.openai.com, 2024. Accessed December 16, 2024.
18. Constantinos Patsakis and Nikolaos Lykousas. Man vs the machine in the struggle for effective text anonymisation in the age of large language models. *Scientific Reports*, 13(1):16026, 2023.
19. David Pissarra, Isabel Curioso, João Alveira, Duarte Pereira, Bruno Ribeiro, Tomás Souper, Vasco Gomes, André V Carreiro, and Vitor Rolla. Unlocking the potential of large language models for clinical text anonymization: A comparative study. *arXiv preprint arXiv:2406.00062*, 2024.
20. Robin Staab, Mark Vero, Mislav Balunović, and Martin Vechev. Beyond memorization: Violating privacy via inference with large language models. *arXiv preprint arXiv:2310.07298*, 2023.
21. Robin Staab, Mark Vero, Mislav Balunović, and Martin Vechev. Large language models are advanced anonymizers. *arXiv preprint arXiv:2402.13846*, 2024.
22. Nicolás Torres and Patricio Olivares. De-anonymizing users across rating datasets via record linkage and quasi-identifier attacks. *Data*, 9(6):75, 2024.
23. Tianyu Yang, Xiaodan Zhu, and Iryna Gurevych. Robust utility-preserving text anonymization based on large language models. *arXiv preprint arXiv:2407.11770*, 2024.

Appendix A: Fine-Tuned LLaMA on Given Paper

Note: The interaction with the fine-tuned LLaMA is included also in **Fig. 1**. Below are the 2 queries we asked the fine-tuned model and their corresponding responses:

Query 1

What is the entire flow of the de anonymization technique in the paper? Include all the steps in the paper.

Answer: The de-anonymization technique in the paper consists of the following steps:

1. nK Structure: Explore multi-hop neighbor information by introducing nK-series structural features for each node. This includes:
   -nK0: The degree of the node (number of neighbors).
   - nK1: The degree histogram of its neighbors.
   - nK2: The degree histogram of its 2-hop neighbors.
   The structural features are represented as a vector ($v(a) = [nK0(a), nK1(a), nK2(a)]\hat{T}$).
2. Diversity Score Calculation: For each node (a), calculate a diversity score (DS(a)) to measure the richness of its structural characteristics. This score is defined using a normalized structural feature vector.
3. Popularity Score Calculation: In each iteration, compute a popularity score (PS(a)) for each node based on its relationships with previously de-anonymized nodes. This score is derived using Jaccard similarity.
4. Structure Score Calculation: Compute a structure score (SS(a)) for each node, combining the diversity score and the popularity score, weighted by a predefined parameter (c).

These steps are part of a blind de-anonymization attack mechanism, which allows for the recovery of identities from anonymized data without prior knowledge of any seed information.

Query 2

What is the entire flow of the de anonymization technique in the paper? Include the use of the ML model specified in the paper.

Answer: The de-anonymization technique in the paper "Blind De-anonymization Attacks using Social Networks" consists of the following steps:

1. Data collection: Gather anonymized graph data and auxiliary graph data from social networks.
2. Feature extraction: Extract features from both graphs using multi-hop neighborhood information, enhancing the structural information used for de-anonymization.
3. Machine learning optimization: Implement a pseudo relevance feedback support vector machine (PRF-SVM) model to optimize the matching process between the anonymized graph and the auxiliary graph. This model uses an iterative approach to refine the matching process based on the relevance feedback from the initial matches.
4. De-anonymization: Perform de-anonymization by matching nodes between the anonymized graph and the auxiliary graph using the optimized PRF-SVM model.
5. Evaluation: Assess the performance of the de-anonymization technique by comparing it with existing techniques, demonstrating its effectiveness and robustness against data perturbations.

The key contributions of this technique include seed-free de-anonymization, multi-hop neighborhood information utilization, machine learning optimization, and robustness to data perturbations. These improvements enable the proposed method to outperform state-of-the-art de-anonymization techniques by up to 10 times.

Appendix B: ChatGPT Interactions

Note: The interaction with ChatGPT-4o is included to ensure transparency and document the rationale behind the analysis methodology. The datasets described in **subsection 5.2** were provided as csv files to the chat. Dataset 1 corresponds to the anonymized Synthesized Adults Dataset and Dataset 2 corresponds to the anonymized Health-Sports Dataset. Below are six queries of the interaction and their corresponding answers, quoting the original conversation:

Query 1

Are these 2 datasets similar in the manner that if i'll apply the same de-anonymization process or methodology of the attack that works well with one of the datasets, it will be effective on the other too? (you can try semantic similarity or something else). If you have catagorical features and want to calculate correlations consider using Cramér's-V correlation and other methods.

Answer: The two datasets contain categorical features:

- **Dataset 1:** Includes columns such as `age`, `marital-status`, `occupation`, `race`, and `salaryclass`.
- **Dataset 2:** Includes columns such as `age_group`, `activity`, `favorite-sport`, `smoking-status`, and `health`.

To determine whether similar de-anonymization methods or attack methodologies would be effective for both datasets, we need to assess the semantic and structural similarities. Here's the plan:

1. **Encoding Categorical Data:** Encode the categorical features to numerical values using techniques like Label Encoding for correlation calculations.
2. **Correlation Analysis:** Use Cramér's V correlation for categorical features to measure the strength of association between variables within each dataset and across datasets.
3. **Semantic Similarity:** Calculate feature name similarity and check for overlapping semantics.
4. **Final Assessment:** Provide a similarity score or conclusion.

I will start by encoding and calculating Cramér's V correlation for these datasets. The Cramér's V correlation matrices for both datasets have been calculated and displayed for analysis. These matrices measure the strength of association between the categorical variables within each dataset. Here's how you can proceed:

- Compare the correlation patterns between the datasets to check for structural similarities.
- For further comparison, we can calculate the semantic similarity between the feature names and analyze how well the features align in terms of meaning.

Would you like to proceed with semantic feature alignment or another specific analysis?

Query 2

Please analyze however you want - will these 2 datasets be affected the same level by the methodology of the same deanonymization attack?

Answer: To analyze whether these datasets will be similarly affected by the same de-anonymization attack methodology, I will consider multiple factors:

1. **Structural Similarity**:
   - Analyze correlations between features within each dataset. If they exhibit similar patterns of association, this suggests structural similarity, potentially affecting de-anonymization methods similarly.
   - Compare the distributions and diversity of values within each dataset. High similarity in value distributions could lead to similar vulnerabilities.
2. **Semantic Similarity**:
   - Examine the alignment between feature names and their encoded meanings.

   - Higher semantic alignment between features (e.g., "age" in one dataset and "age_group" in another) could mean that an attack methodology effective on one dataset would likely work on the other.
3. **Dependency on Features**: Assess if features are interconnected in a way that contributes to identifying individuals uniquely. For example, in one dataset, "marital-status" might heavily correlate with "salaryclass," and in another, "smoking-status" might correlate with "health." Differences here could affect how attacks exploit relationships.
4. **Sensitivity to De-Anonymization Techniques**: Test whether the datasets reveal patterns that can be exploited. For example, if unique combinations of features appear consistently in both datasets, they might both be vulnerable to linkage attacks.

I will start by analyzing the distribution of unique value combinations and feature dependencies for both datasets.

Analysis Results

1. **Unique Combinations Ratio**: Both datasets have the same ratio of unique combinations of rows (30.69%). This indicates a similar level of redundancy or uniqueness in the records, suggesting comparable vulnerability to de-anonymization techniques based on unique attribute combinations.
2. **Average Feature Correlation**: The average feature correlation is identical for both datasets (0.407). This implies a similar level of interdependence among features within each dataset, meaning a similar structure of associations that could be exploited by de-anonymization attacks.
3. **Feature Value Distributions**: The value distributions for individual features show a high level of similarity:
   - For example, in **Dataset 1**, 85% of individuals fall into the `40-79` age range, and in **Dataset 2**, 85% fall into the `50+` age range.
   - Similar patterns exist for other features like "marital-status" vs. "activity" and "occupation" vs. "favorite-sport."

Implications for De-Anonymization:

1. **Linkage Attacks**: If a de-anonymization methodology leverages patterns in feature correlations or unique combinations, both datasets are likely to be equally susceptible due to their identical unique combination ratios and similar correlation levels.
2. **Semantic Mapping**: Features like `age` and `age_group` or `marital-status` and `activity` are semantically aligned, suggesting that techniques effective in exploiting one dataset can likely adapt to the other with minimal adjustment.
3. **Generalization of Attacks**: The similarity in structure and value distributions means that an attack optimized for one dataset is very likely to perform similarly on the other.

Conclusion: The two datasets are structurally and semantically similar to a degree that de-anonymization attacks using the same methodology would likely be equally effective on both. If you'd like, we can test a specific de-anonymization method on these datasets to quantify the impact further. Let me know how you'd like to proceed!

Query 3

Test de-anonymization attacks on datasets.

Answer: De-anonymization attacks often rely on techniques like:

1. **Linkage Attacks:** Matching records from anonymized data to an auxiliary dataset based on overlapping features.
2. **Reconstruction Attacks:** Using combinations of features to identify individuals uniquely.
3. **Probability-Based Attacks:** Leveraging correlations or conditional probabilities between features to infer sensitive data.

To simulate and evaluate the vulnerability of these datasets, I'll:

- Apply a linkage attack by introducing a synthetic auxiliary dataset.
- Test how many records can be accurately matched between the datasets and the auxiliary data.

– Measure the effectiveness by calculating precision, recall, and overall matching success rate.

Let's create and test the linkage attack now.
Linkage Attack Results:

1. **Dataset 1**:
   – **Matches:** 50
   – **Success Rate:** 100% (All records in the auxiliary dataset were correctly matched to the original dataset).
2. **Dataset 2**:
   – **Matches:** 50
   – **Success Rate:** 100% (All records in the auxiliary dataset were correctly matched to the original dataset).

Interpretation: Both datasets are equally vulnerable to a linkage attack, achieving a 100% success rate in matching records between the auxiliary and original datasets. This indicates that the structure and redundancy within the datasets make them fully susceptible to such attacks.
Next Steps:

– If you'd like, we can test other de-anonymization techniques, such as reconstruction or probability-based attacks.
– Alternatively, we can explore strategies to improve anonymization, like introducing noise or generalization. Let me know how you'd like to proceed!

Query 4

What is the auxilarily data you used?

Answer: The auxiliary data was created by randomly sampling 50% of the rows from each of the original datasets. This ensures that the auxiliary dataset represents a subset of the original dataset, mimicking a real-world scenario where an attacker might have partial access to the anonymized data. Here's how it was generated:

1. For **Dataset 1**, 50% of the rows were sampled randomly without replacement.
2. For **Dataset 2**, the same approach was applied.

This subset was then used as the auxiliary dataset to simulate a linkage attack. If you want, I can show the sampled auxiliary data or save it for your reference. Would you like to see or analyze the auxiliary data?

Query 5

Are there more test that can support your decision if the datasets are similar in their effect to the deanonymization methodology?

Answer: Yes, there are additional tests and analyses we can conduct to further evaluate whether the datasets are similarly susceptible to the same de-anonymization methodology:

1. **K-Anonymity**
   – Compute the k-anonymity level for both datasets. A dataset is k-anonymous if each combination of quasi-identifiers appears at least kkk times.
   – If both datasets have similar k-anonymity levels, their susceptibility to de-anonymization via record linkage would be comparable.
2. **L-Diversity**
   – Evaluate the diversity of sensitive attributes for records with the same quasi-identifier group.
   – If both datasets exhibit similar l-diversity levels, they would offer comparable resistance to attacks targeting sensitive attributes.
3. **Quasi-Identifier Overlap**
   – Identify quasi-identifiers in both datasets and measure the extent of overlap.
   – If the quasi-identifiers align well between the datasets, the same attack methodologies are more likely to be effective.

Query 6

What if I don't know the anonymity technique ?

Answer: If the anonymity technique used in the datasets is unknown, the analysis must focus on intrinsic properties of the datasets rather than their adherence to specific anonymization principles. Here's how we can extend the analysis:

1. **Uniqueness of Records**
   - **Test**: Determine the percentage of unique records in each dataset. This measures the likelihood that an individual record can be uniquely identified.
   - **Implementation**: Check the ratio of unique rows in the dataset to total rows.
2. **Feature-Level Entropy**
   - **Test**: Calculate the entropy of each feature in both datasets. High entropy indicates greater variability, which may affect re-identification probabilities.
   - **Implementation**: Use Shannon entropy to assess feature diversity.
3. **Intersection with Auxiliary Data**
   Test: Simulate an auxiliary dataset by generating a subset of data and measure how many records match between the original and auxiliary datasets.
4. **Clustering Analysis**
   - **Test**: Use clustering techniques to group records based on features. Assess the density and separability of clusters.
   - **Insight**: Datasets with tightly packed clusters are more vulnerable to group-based attacks.
5. **Exploitability of Combinations**
   - **Test**: Analyze combinations of quasi-identifiers to determine how often they lead to unique identification.
   - **Implementation**: Test various combinations of features to assess their re-identification potential.
6. **Randomized Re-Sampling**
   - **Test**: Apply random sampling and generalization techniques to simulate potential anonymization and evaluate the stability of features.
   - **Insight**: If key features remain stable across randomizations, the dataset may retain vulnerability to de-anonymization.

Would you like me to implement any of these methods in addition to the current analysis?

Non-tandem spoofing robust speaker verification Technical Report

Amro Asali[1][0009−0007−8044−8129], Yehdua Ben-Shimol[1][0000−0002−4905−2085] and Itshak Lapidot[2,3][0000−0003−1066−3441]

[1] Electrical and computer engineering school, Ben Gurion University of the Negev, Beer-Sheva, Israel
[2] Afeka Tel-Aviv Academic College of Engineering, Israel
[3] Avignon University, LIA, France
asaliam@bgu.ac.il
benshimo@bgu.ac.il
itshakl@afeka.ac.il

Abstract. In this report, we propose a spoofing robust automatic speaker verification (SASV) system. Instead of relying on a standalone countermeasure, we adopt an embedding fusion approach. By integrating embeddings from pre-trained countermeasure and speaker verification models, to this end we use the state-of-the-art AASIST and ECAPA-TDNN models, subsequently producing spoofing robust speaker embeddings. The employed back-end classifier is composed of fully connected layers and parametric ReLU activations. Experiments carried out on the ASVspoof 2019 logical access dataset demonstrate the effectiveness of our approach, achieving equal error rates of 2.0% and 5.1% for the development and evaluation sets, respectively.

1 Introduction

Automatic speaker verification (ASV) systems represent a convenient approach to biometric authentication. The objective of these systems is to ascertain whether a given trial speech utterance corresponds to a claimed enrolled speaker. These systems have a variety of applications, and it is important to ensure their reliability in the face of target-impostor trials and to guarantee security against the presence of spoof attacks, namely fabricated or manipulated speech inputs.

Recent studies have demonstrated that ASV systems are undergoing a gradual evolution, acquiring the capacity to reject spoofed inputs in a zero-shot manner. However, the rapid advancements in speech synthesis techniques, such as *text-to-speech* (TTS) or *voice conversion* (VC), highlight the ongoing necessity to further enhance spoofing-robust ASV systems (SASV) [1]. The results demonstrate that current SASV systems markedly outperform conventional ASV systems in terms of the *spoofing robust automatic speaker verification equal error rat*e (SASV-EER), which is estimated using trials encompassing all three classes: target, bona fide non-target, and spoofed non-target. Reliable speaker verification typically comprises two distinct subsystems: *automatic speaker verification*

(ASV) and spoofing *countermeasure* (CM) classifiers. This leads to the question of how the two subsystems should be integrated in order to achieve robust speaker verification. The results presented in [2] indicate that while joint optimization enhances the reliability of ASV at the SASV level, superior performance is achieved by integrating pre-trained, fixed subsystems. The CM and ASV can be combined in either a cascade or parallel fashion, with integration typically occurring at the score or embedding levels, as evidenced in references [3–5].

2 Background

This section presents a short review of the literature relevant to the proposed solution, a brief overview of the ASV and CM systems utilized for embedding extraction.

2.1 Automatic speaker verification system

In this study, we use the ECAPA-TDNN speaker verification system [6]. The system employs 80-dimensional *Mel-frequency cepstral coefficients* (MFCCs) as features and utilizes a modified Res2Net as its backbone processing block, enhanced with dimensional *squeeze-excitation* (SE) blocks, to model global channel interdependencies. Furthermore, the model incorporates attentive statistics pooling, which enables the model to select the frames it deems relevant, and multi-layer feature aggregation, which captures both shallow and complex speaker identity features at the frame level and aggregates them into utterance-level embeddings.

2.2 Countermeasure system

In this study, we adopt the *audio anti-spoofing using integrated spectro-temporal graph attention networks* (AASIST) approach, which is based on the RawNet2 frontend and graph attention [7]. The model employs a Sinc convolution encoder to extract time-frequency representations, which are subsequently passed through a residual network for high-level feature learning. Two graph modules are then used to model the spectral and temporal domains, respectively.

2.3 Related Work

This study examines two baseline approaches, a score-sum fusion (B_1), and a DNN back-end fusion (B_2). Both approaches use pre-trained ECAPA-TDNN and AASIST models. As illustrated in Figure 1, B_1 employs score-level back-end fusion, which combines the outputs of the ASV and CM subsystems through score addition. In contrast, B_2 employs embedding-level fusion, utilizing a DNN comprising three fully connected layers, each followed by leaky ReLU activation function. This network operates on a pair of speaker embeddings, one extracted from the enrollment utterance and the other from the test utterance, along with a CM embedding derived exclusively from the test utterance [8].

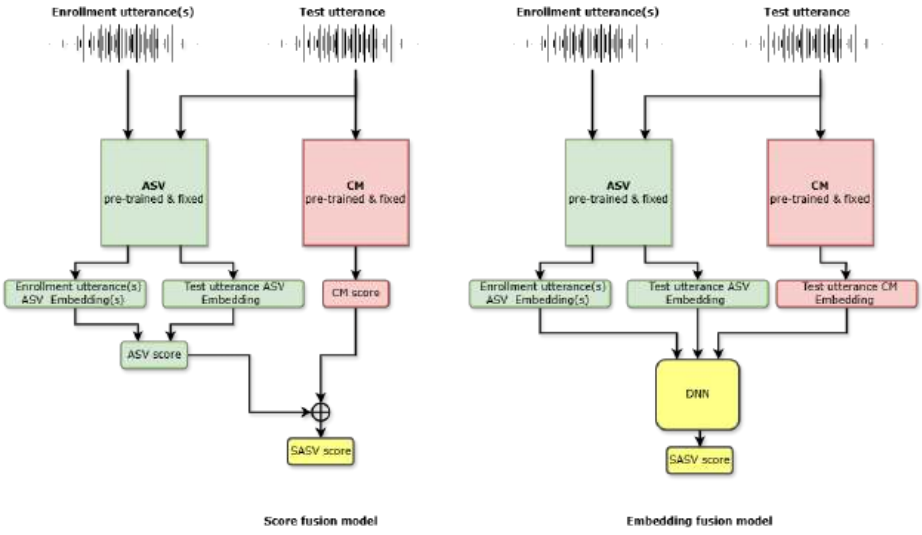


Fig. 1. Two non-tandem solutions. Left panel: a score-sum fusion approach; right panel: a DNN-based, speaker and countermeasure embeddings fusion approach.

3 Proposed architecture

This section presents the proposed SASV system architecture, defines the factorized rectified linear unit (fReLU) activation function used in the design, and provides a brief outline of the system's training details.

3.1 Model

The proposed methodology adopts an embedding fusion-based approach similar to that employed in the B_2 baseline, as depicted in Figure 2. However, we incorporate structured transformation and network-wise, domain adaptation techniques. For each utterance, speaker embeddings are extracted using a pre-trained ECAPA-TDNN model, while countermeasure embeddings are obtained from a pre-trained AASIST model, specifically from the second-to-last layer. Subsequently, the enrollment, trial speaker, and countermeasure embeddings are concatenated to form spoofing-aware speaker embeddings. The fused embeddings are subsequently fed through affine layers and *factorized ReLU* (fReLU) activations, which ultimately produce a logistic score $\hat{y} \in \{0, 1\}$. A score of 1 indicates a target trial, whereas a score of 0 indicates otherwise. The fReLU activation function, applied to the hidden pre-activation $W_i x + b_i$ where W_i and b_i are the learnable weights and biases of the affine layer, is defined as follows [9, 10]:

$$\text{fReLU}_{W_a}(W_i x + b_i) = \max\left(W_a\left(W_i x + b_i\right), 0\right) \tag{1}$$

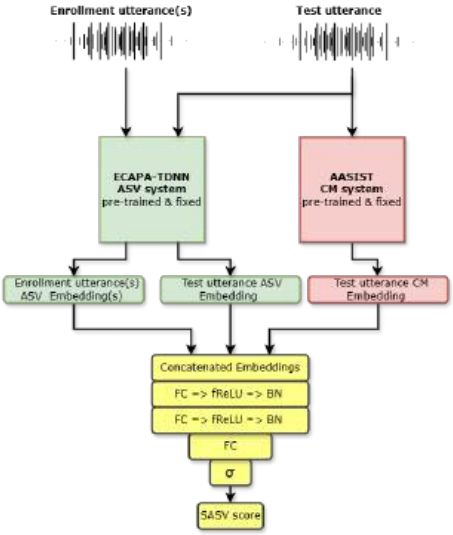


Fig. 2. Proposed back-end classifier.

where W_a is the learnable structural transformation, initialized as the identity matrix.

3.2 System training

Our model was trained for 50 epochs with a learning rate of 10^{-4}. In each epoch, a random 10% subset of the approximately 2M-sample custom dataset, comprising the ASVspoof19 LA dataset's training partition and the VoxCeleb1 E and H partitions is randomly selected [11].

4 Experimental setup

The ASVspoof 2019 Logical Access (LA) dataset is a widely utilized resource for the LA sub-challenge of the ASVspoof 2019 challenge [12]. The dataset comprises genuine speech utterances and those that have been spoofed using a variety of text-to-speech (TTS) and voice conversion (VC) techniques. The dataset is divided into three partitions: training (Train), development (Dev), and evaluation (Eval). The official development and evaluation protocols are included with the dataset. For each test trial, there are multiple corresponding enrollment utterances, which are used to register the target speaker.

5 Results

In order to demonstrate the efficacy of our proposed solution, we compare it with the individual ASV and CM systems utilized for feature extraction, as well as

fusion-based baseline systems. While the ASV and CM subsystems achieve state-of-the-art performance on their respective tasks, as anticipated, they both prove ineffective when applied individually to the spoofing-robust automatic speaker verification task.

Table 1. Comparison of our method with SASV baselines, standalone ASV, and CM systems in terms of SASV-EER on the ASVspoof19 LA dataset evaluation partition.

Systems	**SASV-EER**	
	Dev	Eval
ECAPA-TDNN [6]	17.31	23.84
AASIST [7]	15.86	24.38
Baseline1 [8]	13.06	19.31
Baseline2 [8]	3.10	6.54
Proposed solution	2.01	5.12

6 Conclusions

Given that spoofing scenarios and speaker verification represent distinct domains, it is proposed that, in addition to network-wise adaptation, structural transform adaptation through parameterized ReLU activation can lead to a notable improvement in system performance. In the future, alternative approaches to integrating the CM and ASV systems will be explored, emphasizing embedding-level fusion and other intricate integration methodologies.

Acknowledgments

This work is supported by the Israel Innovation Authority under project numbers 82457 and 82458.

References

1. J. weon Jung, X. Wang, N. Evans, S. Watanabe, H. jin Shim, H. Tak, S. Arora, J. Yamagishi, and J. S. Chung, "To what extent can asv systems naturally defend against spoofing attacks?" in Interspeech 2024, 2024, pp. 3240–3244.
2. W. Ge, H. Tak, M. Todisco, and N. Evans, "On the potential of jointly-optimised solutions to spoofing attack detection and automatic speaker verification," in IberSPEECH 2022, 2022, pp. 51–55.
3. Y. Zhang, G. Zhu, and Z. Duan, "A probabilistic fusion framework for spoofing aware speaker verification," in The Speaker and Language Recognition Workshop (Odyssey 2022), 2022, pp. 77–84.

4. J.-H. Choi, J.-Y. Yang, Y.-R. Jeoung, and J.-H. Chang, "Hyu submission for the sasv challenge 2022: Reforming speaker embeddings with spoofing-aware conditioning," in Interspeech 2022, 2022, pp. 2873–2877.
5. X. Liu, M. Sahidullah, K. A. Lee, and T. Kinnunen, "Generalizing speaker verification for spoof awareness in the embedding space," IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 32, p. 1261–1273, 2024. [Online]. Available: http://dx.doi.org/10.1109/TASLP.2024.3358056
6. B. Desplanques, J. Thienpondt, and K. Demuynck, "Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification," in Interspeech 2020, ser. interspeech-2020. ISCA, Oct. 2020. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2020-2650
7. J.-w. Jung, H.-S. Heo, H. Tak, H.-j. Shim, J. S. Chung, B.-J. Lee, H.-J. Yu, and N. Evans, "Aasist: Audio anti-spoofing using integrated spectro-temporal graph attention networks," in ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2022, pp. 6367–6371.
8. J. weon Jung, H. Tak, H. jin Shim, H.-S. Heo, B.-J. Lee, S.-W. Chung, H.-J. Yu, N. Evans, and T. Kinnunen, "Sasv 2022: The first spoofing-aware speaker verification challenge," 2022. [Online]. Available: https://arxiv.org/abs/2203.14732
9. C. Zhang and P. C. Woodland, "Dnn speaker adaptation using parameterised sigmoid and relu hidden activation functions," in 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2016, pp. 5300–5304.
10. P. Bell, J. Fainberg, O. Klejch, J. Li, S. Renals, and P. Swietojanski, "Adaptation algorithms for neural network-based speech recognition: An overview," IEEE Open Journal of Signal Processing, vol. 2, p. 33–66, 2021. [Online]. Available: http://dx.doi.org/10.1109/OJSP.2020.3045349
11. A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: A large-scale speaker identification dataset," in Interspeech 2017, ser. interspeech-2017. ISCA, Aug. 2017. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2017-950
12. X. Wang, J. Yamagishi, M. Todisco, H. Delgado, A. Nautsch, N. Evans, M. Sahidullah, V. Vestman, T. Kinnunen, K. A. Lee, L. Juvela, P. Alku, Y.-H. Peng, H.-T. Hwang, Y. Tsao, H.-M. Wang, S. L. Maguer, M. Becker, F. Henderson, R. Clark, Y. Zhang, Q. Wang, Y. Jia, K. Onuma, K. Mushika, T. Kaneda, Y. Jiang, L.-J. Liu, Y.-C. Wu, W.-C. Huang, T. Toda, K. Tanaka, H. Kameoka, I. Steiner, D. Matrouf, J.-F. Bonastre, A. Govender, S. Ronanki, J.-X. Zhang, and Z.-H. Ling, "Asvspoof 2019: A large-scale public database of synthesized, converted and replayed speech," 2020. [Online]. Available: https://arxiv.org/abs/1911.01601

Teaching CS101 Using Competitive Programming

anonymous
Mathematics and Computer Science Department
anonymous
anonymous
anonymous

anonymous
Mathematics and Computer Science Department
anonymous
anonymous
anonymous

anonymous
Software Engineering Department
anonymous
anonymous
anonymous

***Abstract*—This article discusses how we integrated methods of competitive programming (CP) into an introductory computer science course at HIT College for first-year students. We provide a detailed review of the CP landscape and previous studies that used CP in computer science education to offer a comprehensive understanding of the field. Additionally, we describe the development and practical application of a tool designed, by one of the authors, to support this approach. Finally, based on our class teaching experience, we demonstrate a significant positive impact of using CP on students' performance compared to those who followed conventional teaching methods.**

***Index Terms*—CS101, teaching CS, CP, Competitive Programming, Hands-on teaching**

I. Introduction

This paper is divided into three major parts. The first section will give a brief overview of CP and some of its uses in education. This will be followed by a short review of the tool we have built to embed CP into CS education. The final chapter will discuss our experience using the system in a live course.

II. Competitive Programming

Before proceeding to the main body of this work, it is essential to review related works and the landscape on which it is based. We review the current status regarding competitive programming: history, approaches and relevant competitions.

There are several aspects of computer science that one can consider competitive: algorithms, hacking and making, entrepreneurship, Cyber competition (CTF), and others. The roots of competitive programming can be traced back to the American Computation Machinery Society's International Collegiate Programming Contest (ACM ICPC), which is widely considered the pioneer of competitive programming.

A. Major competitive programming contests

In this section, we will describe the main competitions that comprise the world of CP today, giving the reader a taste of this vast world. Naturally, it is impossible to include every competition, but we will attempt to highlight the major international ones.

1) ICPC - International Collegiate Programming Contest: The ACM International Collegiate Programming Contest (ICPC [1]) is often referred to as the 'granddaddy' of programming competitions. This nickname is a testament to its long-standing history and the significant role it has played in shaping the competitive programming landscape.

It is an annual team-based programming competition for university students. Where teams of 3 compete to solve complex algorithmic problems within a 5-hour time frame.

The competition utilizes online testing systems with standardized judging. It was founded in 1970 at Texas A&M University and has grown into a global event with regional contests leading to the World Finals. Every year, over 50,000 students from over 3,000 universities across 100+ countries participate in regional contests. Around 130 teams qualify for the World Finals, representing the best of the best.

The ICPC Syllabus focuses on core algorithms, data structures, and problem-solving strategies. Problems range from mathematical modelling to graph theory to dynamic programming and beyond. The difficulty level increases throughout the competition, pushing teams to their limits. It is standard for some problems to remain unsolved during competitions. The ICPC emphasizes logic, creativity, and teamwork to find optimal solutions.

2) IOI - International Olympiad in Informatics: The International Olympiad in Informatics (IOI)[13] is a prestigious annual competitive programming event for high school students worldwide.

The IOI is a globally recognized competitive programming competition for secondary school students (aged 15-19) and is part of the International Science Olympiads. It was founded in 1989 and first held in Pravetz, Bulgaria. It has rotated hosting around the world, fostering cultural exchange and collaboration. Individuals, not teams, compete to solve complex algorithmic problems over two days.

Approximately 90 countries send delegations of up to 4 students and leaders. Each country selects their best young talents through national competitions that should be open to every student in that country. This rigorous selection process ensures that only the most talented and dedicated students represent their countries at the IOI.

The IOI uses an online testing system with standardized judging, similar to the ICPC, but with key differences.:

- Partial scoring is available.
- Competitors' question solution time is not an evaluation factor.
- There is no penalty for wrong answers.

Its syllabus is focused on advanced algorithms, data structures, and innovative problem-solving approaches. Problems encompass diverse areas like graph theory, combinatorics, computational geometry, and other subjects. The difficulty level is high, pushing participants to think creatively and efficiently, although problems tend to be simpler than those in the ICPC. IOI emphasizes logic, creativity, and mental stamina to tackle intricate challenges.

3) IEEExtreme: IEEEXtreme[10], organized by the Institute of Electrical and Electronics Engineers (IEEE), is an annual 24-hour team-based programming competition specifically intended for IEEE student members. It is a global competition in which teams of 3 IEEE Student members compete to solve programming problems within 24 hours, proctored by an IEEE member to ensure fair play and adherence to competition rules. With its unique format and focus, this competition offers a different experience than other CP competitions.

IEEEXtreme was started in 2006 and now boasts over 12,500 participants [1] from more than 2,000 universities across 100+ countries participate each year.

The syllabus is focused on various algorithmic problem-solving techniques, including data structures, algorithms, combinatorics, graph theory, string manipulation and pattern matching.

The difficulty level gradually increases throughout the competition to push teams to their limits. The emphasis is on collaboration, communication, and efficient problem-solving within the team, while the 24-hour time frame dictates efficient time management and sharing.

4) Competitions held by large companies: Large corporations also hold competitions. Even though Google has dropped its competition in the recent cutoffs, this class of competitions is worth mentioning as it boasted an extensive set of problems and greatly impacted CP's progress.

Some competitions are still running, such as the Yandex Cup and Meta's Hacker Cup, but the current trend is that large companies sponsor other contests. For example, Huawei supports ICPC, while Acer supports IOI.

Google Kick Start [8] and Code Jam [7] are online coding challenges Google hosts. They are comprised of a series of online rounds that involve individual participation. The focus of those competitions is algorithms, data structures, problem-solving, and coding skills. It is intended to cater to all skill levels, whereas Kick Start is more focused on encouraging newcomers to participate in competitive programming.

Meta Hacker Cup [15] is a prestigious online competition hosted by Meta (Facebook), featuring challenging algorithmic problems and real-world scenarios. It contains several online rounds culminating in a final round with individual participation. Those rounds are focused on advanced algorithms, data structures, problem-solving under pressure, and software engineering knowledge. Difficulty level increases at each round. It is primarily aimed at experienced programmers and software engineers.

Yandex algorithm [22] is a competition hosted by Yandex. It was held annually from 2011 to 2018 and is now a separate track of the Yandex Cup. It has a few online rounds, culminating in a final round with 20 participants held in person. This competition is focused on every aspect of competitive programming. Although the competition claims it aims for international participation, it focuses more on Russian-speaking countries.

5) Different competitions that are commonly referred to as coding competitions by mistake: A few other types of competitions might look similar to those mentioned above but are pretty different. Moreover, although they might require writing code as part of the competition, they have a very different nature and should not be considered CP competitions.

Artificial Intelligence (AI) Competitions: In "AI" titled competitions, the code is not evaluated versus a known solution but is evaluated vs code from another competitor running on the same platform. These competitions involve algorithms but focus on strategy and exploiting knowledge about the simulation platform. In Israel there are the skillz [16] competition for secondary school students and codeguru extreme [3] which is based on corewars8086 [5], there are many international competitions such as the Russian AI cup [20], and reply challenge [18].

Approximations Competitions: In approximation competitions, competitors are given a problem known to be very hard (this will typically include some variant of well-known NP or NP-complete problems). Competitors are given plenty of time to provide a solution, and those solutions are typically evaluated in a two-step process: first, as being technically correct. Afterwards, they will be scored based on their relative performance compared to other solutions. This class of problems can be demonstrated by the well-known "travelling salesman" problem, where calculating an optimal solution is very hard. However, given a solution, its viability is easy to verify, and scoring based on various "cost" elements is not too hard. Some examples of this include the Google Hash Code [9] (no longer with us), and the SoGC (Symposium on Computational Geometry) challenge geometric approximation competition, for which solutions can lead to research articles such as [19] and [6].

Data Science Competitions: In "Data Science" competitions, competitors are given a large dataset and should devise an algorithm that provides some insights into the data. The public dataset includes this insight, but a private set also exists, and solutions are evaluated on this "hidden" set. The first evaluation is done on a "public" stage, where results are shown for all competitors, before evaluating a "private" stage, which provides final results. Kaggle is the most well-known location for this type of competition. Participation in its competitions has led to many research articles such as [2] and [12]

[1] 4,231 teams appear in IEEEXtreme 17.0 ranking [11]

6) Less similar competitions, but still in the world of CS: A few other competitions are noteworthy; while involving aspects of CP and other elements from previously mentioned types, we describe them below, but they fall outside the scope of our work.

Capture the Flag - CTF: One type is a "Capture the Flag" (CTF) competition, which is a cybersecurity exercise where teams or individuals compete to find and exploit vulnerabilities in simulated systems or programs to "capture flags", which are pieces of hidden data or secret messages. These flags often take the form of text strings or specific data embedded within the system. It may require programming and algorithms to solve some of those challenges, but this is not the core of the competition.

Hackathons: Another type we mention here is hackathons, which are intense, collaborative events where programmers, designers, and other creative minds come together to build software or hardware projects within a limited timeframe, typically 24 to 48 hours. The result is often a prototype of those projects and sometimes just a presentation of a given idea. Hackathons are often more focused on human networking aspects, such as getting to know each other, meeting mentors or using sponsor-based APIs. The actual coding is often minor, and algorithms are even less so.

B. Previous works on CP in CS education

There are many studies and publicatons on CP and how it might be used in education. In this section, we will review a few of those previous publications that are related to our work. As the scope of this article is limited. So we have hand picked two articles with a strong correlation to our experience.

1) Facilitating course assessment with a competitive programming platform: This paper [4] from 2019 Coore and Fokum describe their experience with the introduction of the use of a competitive programming platform as a mechanism for auto-grading assignments for an introductory course on algorithm design and analysis. They used HackerRank to increase the number of assessed programming exercises to an average of one per week from two in the entire semester of previous years. Large enrolment with only a few graduate assistants available meant that prior to the change, only a few assignments were given, and students waited long for feedback. Fresh problems were developed for deployment on the platform, each targeting the specific learning objectives of the week they were given. The assignments were given in a contest format, and students could submit multiple attempts without penalty. The paper observed an increased degree of engagement with the course content. However, a statistical analysis shows that the impact of the intervention on student performance, relative to previous course instances, was mixed. However, there was some visible improvement in final test scores. Furthermore, no control group existed, so they could not reach a clear-cut conclusion.

We note that in comparison to our work, their class size was much smaller, and that they used previous years as reference, while our group was quite larger and had other groups that took the same class on the same year with the same final test as a comparison point.

2) C++ Teaching Reform and Exploration Based on ACM/ICPC and Live Code: This paper [23] from 2022 by Zheng and Sarem discusses a pedagogy for teaching C++ programming based on ICPC problems and "live" code. They believe that live code pedagogy is a functional, easy and fun method that can reduce the difficulty of teaching the C++ language. They claim that live code teaching can give students visual and tactile stimulation, which aligns with the theories of pedagogy and psychology. More than ten years of practical teaching results show that the proposed live code pedagogy and the ACM/ICPC-based reform improve students' programming abilities.

This work is rigorous and quite different from ours, but it shows the advantages of using "live" code, however since they have no control group, they focus on surveys administered to participating students in order to reach some of their conclusions.

III. Building CPE3 Website

It was decided that in order to facilitate CP in a course teaching, and considering the lack of tools needed in Hebrew. A website is the best option to achieve this goal while providing students with tools to learn about algorithms and data structures "hands-on."

The first question is, why not use something ready or alter an existing solution?

The main reason for our decision was the lack of a suitable system to meet our requirements. We needed a system that could seamlessly support RTL (Right to Left) languages like Hebrew. The primary system we considered, CMS [14], was designed for something other than our purposes (Managing contests, not collecting questions) and needed more utilities, leading us to abandon it as an option. Another compelling reason for our decision was the fascinating prospect of creating a system from scratch, documenting the challenges encountered, and enhancing the research work. However - this is not the focus of this article.

A. Designing the system

The system design was based on the first author's experience in building a few relevant systems, such as Mathter [21] for collecting Math problems and Turtleacademy [17] for teaching programming to kids.

The requirements necessary for a basic proof of concept:

- There must be a way to show different problems in the system. The presentation should be attractive enough for users and very readable.
- Every problem should include a description and some sample tests that are shown to the user.
- The system must support seamless changing of languages for users. The minimal set should include Hebrew and English, but extending this list in the future should be possible.

- The system must allow users to register to submit their solutions. There is no such thing as an anonymous solver.
- The system should automatically support the evaluation of user submissions, providing them with information about the results of their efforts as quickly as possible. The evaluation process is quite complex and will be detailed later in this chapter.
- There must be an easy way for teachers/administrators to add new questions to the system.

There are also a few more items that were already implemented more will be added as the site progresses:

- There should be a way to withhold results of testing (the process of pre-testing).
- Users should be able to track their progress on the site.
- Users that have solved a given problem should be able to see other solutions for the same problem.
- It should be possible to ”hack” a solution, which is to provide inputs to the problem that will make a few given solutions fail.
- Integration with third-party Single Sign On (SSO) solution will prove quite helpful with administration.
- Marking problems and assigning them to specific topics.

More features will remain for future work:

- Adding the ability to obfuscate questions by changing a given question ”story” while keeping the algorithmic core and example the same, maybe using a GPT.
- Generation of unique inputs for tasks while keeping the random seed.
- Advanced plagiarism detection at the object level.

B. The resulting site

Figure 1 shows the home page, where people start their journeys; it clearly shows what can be done; the options to login/register are visible at the top, and the option to switch languages. A card-based design leads to the main elements of the site.

We will elaborate more about the challenges in creating this site in a future article.

IV. HIT CS101 - Introduction to Computer Science

The site was used for exercises in an introductory computer science course in HIT. The Holon Institute of Technology (HIT) is a public Israeli institution known for its focus on science, engineering, and technology. HIT offers undergraduate and graduate programs in computer science and other fields.

HIT offers a three-year Bachelor of Science (B.Sc.) program in Computer Science designed to equip students with a well-rounded understanding of the field. The program starts with a few mandatory classes that each student must attend. One of them is ”Introduction to CS, using the C language”, in which the author of this thesis had the chance to use the developed site as a part of the course.

The course conducted at the first semester had 555 students in 7 different classes, guided by six instructors. Another class was dedicated to army personnel and reserves because of the current war; that class had 69 students excluded from the statistics collected here because of the inherent differences. The classes are quite different because some classes target working students (evening classes) and have many participants with prior coding experience.

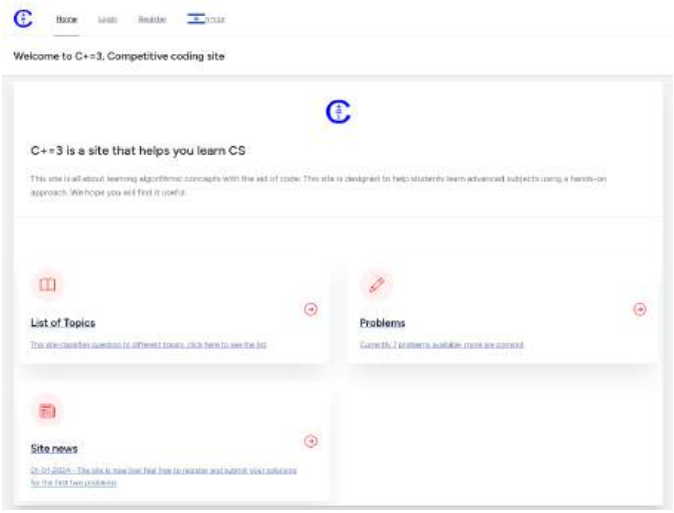


Fig. 1. CPE3 Front page

Unfortunately for us, this was a unique semester in which war was a part of daily life, causing a few changes to the course:

- Start of courses was delayed by two months, and the semester was shortened by four weeks.
- There were no mandatory assignments.
- The class of army personnel students had a particular semester of only nine weeks. They were evaluated using a test that was different from the other groups.

The author’s original plan was to use the site as a tool for the mandatory assignments given during the course, differentiating based on assignment usage and other factors. However, since the system was new and this semester was more complex, it became very hard to convince the course’s staff to provide assignments, as a few objections were raised.

The first suggestion to resolve this scenario was to use a question from the site ”as is” in the test, but this did not go through. Some professors did not want to cooperate with this, and some questions on the site were deemed ”too hard” for first-year students.

However, since the first author was one of the teachers, he was able to suggest using the site for all the students in the course while being more ”pushy” for students in one’s group. Questions were added to the site based on the course’s progress. Students were encouraged to try the problems there (The problems being more challenging than the standard exercise problems, as expected from competitive programming questions). Students were given positive feedback on success and could see other students’ solutions after creating a correct submission.

At the end of the semester, the author promised students a

limited bonus on their final score based on their performance on the site.

The only factor for the grade was a single exam taken at the end of the semester. The results are presented in table I.

TABLE I
STUDENT RESULTS - 1ST EXAM

Class number	Students	Fail %	Average grade	Median grade	STD DEV
1	76	35.5	65.7	71	29.0
2	85	34.1	68.4	75	27.8
3	75	20	74.7	80	27.6
4	88	25	70.8	77.5	27.5
5	53	28.3	70.6	80	28
6	66	16.7	75.5	86	27.4
7	59	33.9	64.3	70	27.8
total	502	27.7	70	77	27.9

Usually, there is a second sitting of the exam for students failing the first test and those who could not attend the test the first time. However, because of the war, students were allowed to take the second test and get the higher results of both tests. The results for this exam may be found in table II.

TABLE II
STUDENT RESULTS - 2ND EXAM

Class number	Students	Fail %	Average grade	Median grade	STD DEV
1	44	52.3	52.5	46.5	27.4
2	44	52.3	56.9	54	27.6
3	36	38.9	57.8	68.5	32.5
4	42	33.3	64.0	70.5	25.7
5	21	61.9	52.2	40	30.1
6	28	39.3	57.9	60	30.8
7	38	47.4	53.4	60	23.7
total	253	45.8	56.6	60	28

A. Analysis of the author's group (#3)

Since participating in the site was not mandatory in any way, this work will have to focus on data that the primary author was able to collect. This data is focused on group #3, the group of students taught by this author.

Since using the site was not directly in the course syllabus, the author has used several methods to get students to participate in the CPE3 site, first by giving the students the site as an additional resource. The author used content from the site within lectures and scheduled the release of new questions based on the materials being taught. Near the end of the class, when the test was approaching, a "secret" bonus was proposed to site participants, using a code they could get from the site after solving several questions.

B. Group statistics

The group had 83 students, of which 79 took the final exam. There were 55 accounts present on the site, of which 54 took the test, 53 had any submissions, and 51 had at least one successful solution. (Given that the question "A+B" is included, this is not much of an achievement, but it shows minimal effort).

The following graphs show the performance of the students in group 3 when taking into account several parameters that might impact overall results.

1) CPE3 Users vs All students: Figure 2 illustrates the difference in average results between the students using the CPE3 site (colored in red) vs All students (CPE3 are in both groups). The score here is before any impact of the bonus promised to site participants.

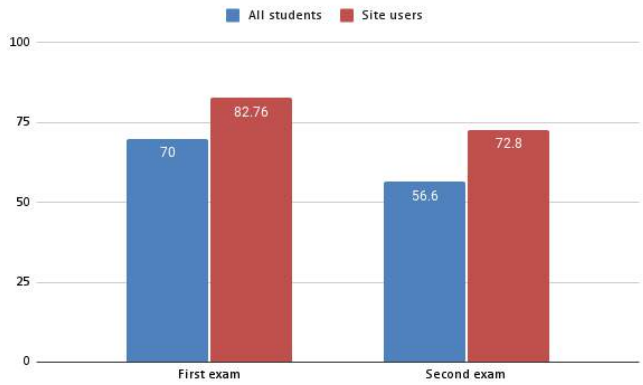


Fig. 2. All students vs Site users in both exams

It is interesting to see that the site users were clearly ahead of the rest, scoring 12.76 points more than the average student on the first exam and 16.2 points more than the average student on the second exam.

2) Inner spilt of users in group #3: Figure 3 shows the inner split between students in group 3 who used the site vs students who never successfully submitted any solution to the site.

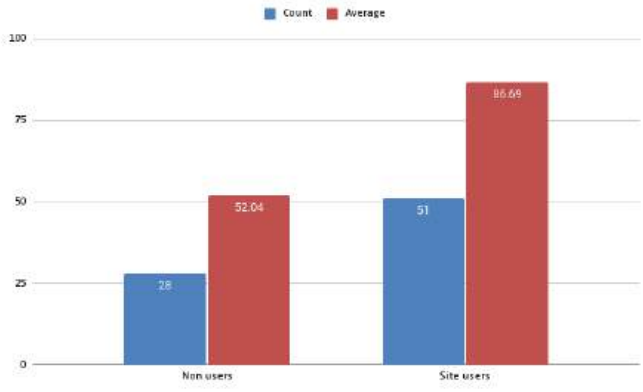


Fig. 3. Users vs Non-users in group 3, count and average

This graph compares the final averages of both groups. We noticed that the non-users, containing 28 students, received a final average score of just 52.04. The students who used the site were able to get a final grade with an average of 86.69,

which is 34.65 points higher. This result limitations are further discussed in the discussion part of our article.

3) The results of putting more effort: Figure 4 shows the correlation between the number of questions successfully solved by users in group 3 and their average score. It is to be noted that every student that solved problems on our site participated in the test.

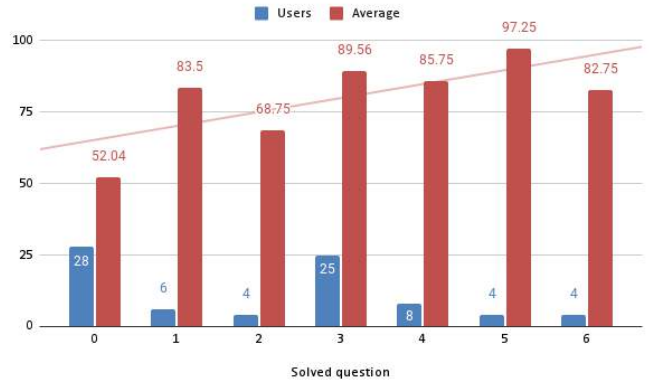


Fig. 4. Users in group 3, by number of successful submissions

The blue bar indicates the number of students in each group, while the red bar indicates the average score for that group.

What is interesting to look at here is that a trend is clearly visible: the more effort students put into the site, the higher they score on the test. Some anomalies are also clearly visible and will be explained. The first anomaly is that there are many users on 3 questions, probably more than there should have been. The reason is that the promised bonus needed a threshold of 3 questions, so many people forced their way there. (The average does not include any bonuses given). The second anomaly is the results for the first and last groups (1 and 6 questions), and it should be noted that the groups are pretty small, so even one student makes a huge difference, which the author assumes is the case in both groups. The first included students who were good enough even without needing to use the site (they might have had previous experience), so after entering the site and submitting a single answer, they decided to stop. The last group included one student who failed to pass the test (twice), which took the group average way down. Without that specific student, the group average for the last group would have been approximately 97 as well).

4) Initial users vs "Bonus" users: As previously mentioned, the author of this thesis had to push his students a little into using the site. This was done by promising students a bonus on their test grade, ranging from 3 to 5 points, given a code they had to provide on their paper that could be achieved on the site after solving a certain number of problems. This naturally pushed some students into "borrowing" submissions from other students, and the author wanted to know what the impact of that was.

Figure 5 divides the students who were considered site users into two groups. This division is done based on the date of their registration to the site, either before or after the bonus was announced.

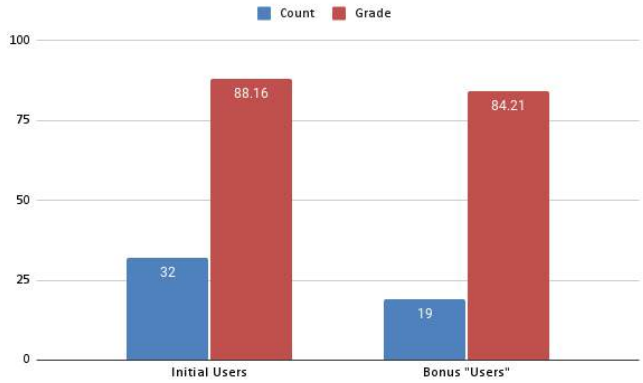


Fig. 5. Initial users vs bonus users

What can be seen here is that there were 19 members of the bonus group, which achieved similar results to the users that initially registered (the difference is under 4 points) and may be considered a marginal error. However, it is still safe to assume that using the site constantly during the class had a better impact than just rushing to submit for a promised bonus.

V. Discussion and Limitations

Drawing definitive conclusions based on the presented results would be presumptuous, and they should be viewed with caution. It's important to keep in mind that this study only involved one group, conducted once in a single course under extreme conditions that significantly impacted the overall environment. There could be other factors contributing to why the students using the site outperformed their peers and other groups. It's possible that this outcome was influenced by "experimental bias," especially since the individuals conducting the test are also the authors of this article.

However, the results are quite striking. The students who utilized the site excelled compared to their counterparts in the same establishment under the same challenging conditions. Additionally, it was evident that the more effort they invested in using the site, the better their results were.

The authors urge further research to validate whether these results are reproducible and applicable across other classes.

References

[1] ACM. International collegiate programming contest. URL https://icpc.global/.

[2] C. S. Bojer and J. P. Meldgaard. Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting*, 37(2):587–603, 2021.

[3] Codeguru. Codeguru xtreme competition. URL https://codeguru.co.il/Xtreme/.

[4] D. Coore and D. Fokum. Facilitating course assessment with a competitive programming platform. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 449–455, 2019.

[5] F. Corno, E. Sanchez, and G. Squillero. On the evolution of corewar warriors. In *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No. 04TH8753)*, volume 1, pages 133–138. IEEE, 2004.

[6] F. Fontan, P. Lafourcade, L. Libralesso, and B. Momège. Local search with weighting schemes for the cg: Shop 2022 competition. In *38th International Symposium on Computational Geometry, SoCG*, 2022.

[7] Google. Google code jam, . URL https://codingcompetitionsonair.withgoogle.com/.

[8] Google. Coding practice with kick start, . URL https://codingcompetitionsonair.withgoogle.com/events/kickstart.

[9] Google. Hash code programming competition, . URL https://en.wikipedia.org/wiki/Hash_Code_(programming_competition).

[10] IEEE. Ieeextreme – 24-programming competition. URL https://ieeextreme.org/.

[11] IEEEXtreme. Ieee 17.0 ranking. URL https://ieeextreme.org/ieeextreme-17-0-ranking/.

[12] V. Iglovikov, S. Mushinskiy, and V. Osin. Satellite imagery feature detection using deep convolutional neural network: A kaggle competition. *arXiv preprint arXiv:1706.06169*, 2017.

[13] IOI. International olympiad in informatics (ioi). URL https://ioinformatics.org/.

[14] S. Maggiolo, G. Mascellani, et al. Introducing cms: a contest management system. *Olympiads in Informatics*, 6:86–99, 2012.

[15] Meta. Meta hacker cup. URL https://www.facebook.com/codingcompetitions/hacker-cup.

[16] I. MOE. Skillz - cyber competition. URL https://pub.skillz-edu.org/portal/.

[17] U. W. Ofer Wald. Turtle academy. URL https://turtleacademy.com.

[18] Reply. Reply code challenge. URL https://challenges.reply.com/.

[19] A. Schidler. Sat-based local search for plane subgraph partitions (cg challenge). In *38th International Symposium on Computational Geometry (SoCG 2022)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2022.

[20] VK. Russian ai cup. URL https://russianaicup.ru/.

[21] O. Wald. Mathter - website for math questions. URL https://mathter.islands.co.il/.

[22] Yandex. Algorithm — yandex cup. URL https://yandex.com/cup/algorithm/.

[23] Y. Zheng and M. Sarem. C++ teaching reform and exploration based on acm/icpc and live code. In *Proceedings of the 2022 5th International Conference on Education Technology Management*, pages 281–286, 2022.

Finding a Vertexwise Minimum Depth Tree Among the Shortest Path Trees*

(Preliminary Version)

Omer Asher[1], Yefim Dinitz[1], Shlomi Dolev[1], and Li-on Raviv[2]

[1] Department of Computer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel
[2] Gilat, Israel

Abstract. This paper introduces a polynomial-time modification to Dijkstra's algorithm aimed at constructing a Minimal Depths Shortest Path Tree (MDSPT) from a source vertex s in a weighted graph. Unlike the standard Dijkstra's algorithm, which prioritizes minimizing path weights without regard for path depths (i.e., the number of edges), this modified approach ensures that among paths of equal weight, the one with the fewest edges is selected. This enhancement is achieved by tracking both path weight and depth during the algorithm's execution. The paper provides formal definitions of key concepts and proves the correctness of the modified algorithm. This work extends the applicability of shortest path algorithms to scenarios where minimizing path depths is also substantial. Although the problem discussed may be consider folklore, here we present a formal treatment and proof for the solution.

1 Introduction

Given a graph with non-negative edge weights and a source vertex s, a Shortest Path Tree (SPT) is found by the classic algorithm Dijkstra [2]. We consider the enhanced problem setting where the path depths in SPT, as the second criterion, should be minimized at every vertex in the graph; we call such an tree Minimal Depths Shortest Path Tree (MDSPT). We suggest a modified version of Dijkstra's algorithm computing MDSPT for an arbitrary given graph, with the same time complexity as algorithm Dijkstra. This algorithm is useful in cases where among paths of equal length, the one with fewer edges is preferred. The standard Dijkstra's algorithm computes the shortest path distances but does not prioritize minimizing the number of edges in the case of ties. In this modified approach, we introduce a depth tracking mechanism to achieve this additional criterion. For a similar approach to prioritized multi-criteria tasks, see [3]. There, the *lexicographic priority* for several criteria is used, meaning: 1) to minimize the first criterion, and 2) while keeping it fixed, to minimize the second criterion, and so on.

Finding a minimal depth tree among all shortest path trees is useful in various domains where both path weight and depth need to be considered simultaneously. This makes the algorithm applicable to problems where traditional shortest path algorithms fall short, such as when there is a need to optimize not only the cost of traversal but also the number of edges traversed.

2 Problem Setting and Notations

Consider a directed graph $G = (V, E)$ with a weight function $w : E \to \mathbb{R}$, where $s \in V$ is the source vertex, and all vertices in V are reachable from s. Note that all results of this paper are correct for the case of undirected graphs, by the known reduction replacing every undirected edge by two anti-parallel edges between the same vertices. For a path P in G, the weight of P, denoted by $w(P)$, is the sum of weights of all edges in P:

$$w(P) = \sum_{e \in P} w(e).$$

* Partially supported by the Rita Altura trust chair in computer science, BGU-NJIT grant, the Israeli Smart Transportation Research Center (ISTRC), and Israeli Science Foundation (Grant No. 465/22).

The depth of a path P in G, denoted by $|P|$, is the number of edges in the path. For any $v \in V$, the shortest path distance is denoted as $\beta(v)$:

$$\beta(v) = \min_{P \in \mathcal{P}(v)} w(P),$$

where $\mathcal{P}(v)$ is the set of all paths from s to v. A Shortest Path Tree (SPT) is a tree in G rooted at s where the path from s to any other vertex v in the tree is the shortest path possible in G.

Definition 1 (Path Depth). *Let $T = (V', E')$ be a tree rooted at s and $v \in V$. The **path depth** of v in T is the number of edges in the path from s to v within T.*

Definition 2 (Minimal Path Depth). *For any $v \in V$, the minimal path depth among all the paths that weight $d(v)$ is denoted as $\gamma_{d(v)}(v)$:*

$$\gamma_{d(v)}(v) = \min_{P \in \mathcal{P}_{d(v)}(v)} |P|,$$

where $\mathcal{P}_{d(v)}(v)$ is the set of all paths from s to v in G that weight $d(v)$.

Definition 3 (Minimal Depths Shortest Path Tree (MDSPT)). *A Minimal Depths Shortest Path Tree (MDSPT) is a Shortest Path Tree T, such that for any vertex $v \in V$, the path depth of v in T is minimal over all possible shortest paths from s to v in G.*

3 Algorithm

Our algorithm (see its pseudo-code in Algorithm 1) uses $(d(v), \ell(v))$ records to maintain the current parameters of the best known path from the source vertex to each other vertex $v \in V$. The $d(v)$ component stores the cumulative weight of the shortest currently known path from s to vertex v, while the $\ell(v)$ component tracks the currently known minimal number of edges (depth) in a path of weight $d(v)$. This dual representation allows resolving ties in path weight by preferring the path with fewer edges. As the algorithm progresses, both components are updated for each vertex. Additionally, the algorithm uses an array `prev[]` to store the predecessor of each vertex on its currently shortest path of the minimal depth from the source s. For a given vertex v, `prev[v]` indicates the vertex immediately preceding v on such a path. This array is crucial for reconstructing the shortest path after the algorithm has completed its execution.

Routine 1 Relax(u,v)

```
1: if d(u) + w(u,v) < d(v) then
2:     d(v) ← d(u) + w(u,v)
3:     ℓ(v) ← ℓ(u) + 1
4:     prev[v] ← u
5: else if d(u) + w(u,v) = d(v) then
6:     if ℓ(u) + 1 < ℓ(v) then
7:         ℓ(v) ← ℓ(u) + 1
8:         prev[v] ← u
9:     end if
10: end if
```

The `Relax(u,v)` method evaluates whether updating the distance to vertex v via edge (u, v) is advantageous. It updates the distance if a path with less weight is found, or if a shallower path with the same weight is discovered.

During execution, the priority queue Q maintains the vertices yet to be processed w.r.t. the lexicographical order of their distance records $(d(v); \ell(v))$. The `ExtractMin(Q)` method selects from Q a vertex with the lexicographically smallest distance. Once all vertices are processed, the queue becomes empty, and the algorithm concludes with the shortest path information stored in the distance records and `prev[]`.

Algorithm 1 MDSPT Finding

Require: Graph $G = (V, E)$ with non-negative edge weights, source vertex s
initialize: distance $d(v) \leftarrow \infty$, $\ell(v) \leftarrow \infty$ and `prev[v]` $\leftarrow$ `null` for all $v \in V$
$d(s) \leftarrow 0$, $\ell(s) \leftarrow 0$
$Q \leftarrow$ priority queue containing all vertices $v \in V$, prioritized by $(d(v), \ell(v))$ lexicographically
$S \leftarrow \emptyset$
while $Q \neq \emptyset$ **do**
    $u \leftarrow$ ExtractMin(Q)
    $S \leftarrow S \cup \{u\}$
    for each neighbor $v \in Q$ of u **do**
        Relax(u, v)
    end for
end while
return `prev[]`

4 Algorithm Correctness

We are given a graph $G = (V, E)$ with non-negative edge weights and a source vertex s. The objective is to formally verify that the algorithm correctly computes $d(v) = \beta(v)$, $\ell(v) = \gamma_{d(v)}(v)$, for all $v \in V$, and that the array `prev[]` defines a Minimal Depths Shortest Path Tree (MDSPT).

Lemma 1. *Once a vertex is added to S, its distance and prev records remain unchanged.*

Proof. The *dist* and `prev[]` records of a vertex v can only change during a relaxation step, where a shorter or shallower path to v is found. However, the algorithm only relaxes edges to vertices that are in the priority queue Q (which contains vertices that have not yet been added to S). Once a vertex u is added to S, it is removed from Q, and no further relaxation involving u occurs. Therefore, distance records of u cannot change after it is added to S.

Lemma 2. *After the i'th step of the algorithm, for any $v \in V$, $\beta(v) \leq d(v)$ and $\gamma_{d(v)}(v) \leq \ell(v)$ is satisfied, and if $d(v) < \infty$, there exists a path from s to v with weight $d(v)$ and depth $\ell(v)$.*

Proof. We prove this by induction on the number of relaxations.

Base case: At the start of the algorithm, we initialize $d(s) = 0$, $\ell(s) = 0$ for the source vertex s, and $d(v) = \infty$, $\ell(v) = \infty$, for all $v \neq s$. Since $\beta(s) = 0, \gamma_0(s) = 0$, it follows that $\beta(s) \leq d(s)$, $\gamma_0(s) \leq \ell(s)$, and for all other vertices v, $\beta(v) \leq \infty = d(v)$, $\gamma_{d(v)}(v) \leq \infty = \ell(v)$.

Inductive Hypothesis: Assume that after the $i-1$'th relaxations, for all vertices $v \in V$, $\beta(v) \leq d(v)$, $\gamma_{d(v)}(v) \leq \ell(v)$, and if $d(v) < \infty$, there exists a path from s to v with weight $d(v)$ and depth $\ell(v)$.

Inductive Step: During the i'th relaxation, the algorithm ensures that $d(v)$ is updated to the minimum of its current value and $d(u) + w(u, v)$. If $d(v)$ changes to $d(u) + w(u, v)$, then $\ell(v)$ is updated to $\ell(u) + 1$. By the inductive hypothesis, $d(u)$ is the weight of a path to u of depth $\ell(u)$. Thus, $\ell(u) + 1$ is the depth of a path to v of the currently optimal weight $d(u) + w(u, v)$.

Otherwise, if $d(v)$ does not change, $\ell(v)$ is updated to $\min(\ell(v), \ell(u)+1)$. If $\ell(v)$ changes to $\ell(u)+1$, then by the inductive hypothesis, the record of $\ell(u)$ is the length of a path to u that weights $d(u)$. Thus, $\ell(u) + 1$ is the length of a path to v through u that weights $d(v) = d(u) + w(u, v)$. If no changes occurred, the lemma holds by the inductive hypothesis. Since each value corresponds to a specific path, and since we defined $\beta(v)$ as the shortest path weight to v, we conclude $\beta(v) \leq d(v)$. Similarly, $\ell(v)$ is the depth of a path that weighs $d(v)$, thus it cannot be smaller than $\gamma_{d(v)}(v)$, which is the minimal depth among all the paths that weights $d(v)$. Therefore, $\gamma_{d(v)}(v) \leq \ell(v)$.

Lemma 3. *Let S_i denote the set of vertices selected after i steps. After the i'th step of the algorithm, for any $v \in S_i$, $d(v) = \beta(v)$, $\ell(v) = \gamma_{\beta(v)}(v)$ are satisfied, and the array `prev[]` restricted to S_i defines an MDSPT for S_i.*

Proof. **Base Case:** Initially, $S_0 = \{s\}$, and the source vertex s has $d(s) = 0$, $\ell(s) = 0$. This case trivially holds, as the shortest path from s to itself is of distance 0 and depth 0, and the array `prev[]` restricted to $\{s\}$ defines a tree that includes only s, which is an MDSPT.

Inductive Hypothesis: Assume that after the $i-1$'th iteration, for all vertices $v \in S_{i-1}$, $d(v) = \beta(v)$, $\ell(v) = \gamma_{\beta(v)}(v)$ and the array `prev[]` restricted to S_{i-1} defines an MDSPT.

Inductive Step: Let v_i denote the vertex added to S by the algorithm during the i-th iteration. From Lemma 1, we know that the distance records of all vertices in S_i, except for v_i, remain unchanged. So, our objective is to establish that $d(v_i) = \beta(v_i)$, $\ell(v_i) = \gamma_{d(v_i)}(v_i)$, and that the `prev[]` array, when restricted to S_i, defines an MDSPT. The distinction between the tree defined by `prev[]` for S_{i-1} and S_i lies in the inclusion of the vertex v_i and its associated edge $(\text{prev}[v_i], v_i)$. Thus, we must also verify that the weight of the path from s to v_i in T_i equals $\beta(v_i)$, and its length is $\gamma_{\beta(v_i)}(v_i)$.

Lemma 2 states that after the ith iteration:

$$\beta(v_i) \leq d(v_i), \tag{1}$$

$$\gamma_{d(v_i)}(v_i) \leq \ell(v_i). \tag{2}$$

Let P be a shortest path from s to v_i that has minimal depth, that is:

$$w(P) = \beta(v_i), \tag{3}$$

$$|P| = \gamma_{\beta(v_i)}(v_i). \tag{4}$$

Note that since P is a shortest path with the minimal depth from s to v_i (a lexicographically minimal path), any prefix of P leading to any intermediate vertex is also a shortest path with the minimal depth. Indeed, if this was not the case, we could replace the prefix either with a shorter path or one with the same length and lesser depth, which would contradict the lexicographic minimality of P.

In P, $s \in S$, $v_i \notin S$. Let (x, y) denote the first edge in P such that $x \in S$ and $y \notin S$. Let $P^{(1)}$ denote the part of P from the start up to x, and $P^{(2)}$ denote the rest of the path after (x, y). From now on, we consider the variable values at the moment of the beginning of the ith iteration since the distance records of v_i do not change during the current iteration. In particular, $S = S_{i-1}$. Since $x \in S$, by the inductive hypothesis, $d(x) = \beta(x)$ and $\ell(x) = \gamma_{\beta(x)}(x)$. Moreover, by Lemma 1, these values remain unchanged during the current iteration.

$$d(x) = \beta(x), \tag{5}$$

$$\ell(x) = \gamma_{\beta(x)}(x). \tag{6}$$

When x was added to S, the algorithm considered updating $d(y)$ using $d(y) \leftarrow d(x) + w(x, y) \overset{(5)}{=} \beta(x) + w(x, y)$. Since then, the record of $d(y)$ can only decrease. Thus, after the current iteration,

$$d(y) \leq \beta(x) + w(x, y). \tag{7}$$

Since $\beta(x)$ is the shortest path distance from s to x, P is a lexicographically minimal path, and $P^{(1)}$ is its prefix:

$$\begin{aligned} \beta(x) &= w(P^{(1)}) \\ \beta(x) + w(x, y) &= w(P^{(1)}) + w(x, y) \\ \beta(x) + w(x, y) &\leq w(P^{(1)}) + w(x, y) + w(P^{(2)}). \\ \beta(x) + w(x, y) &\leq w(P). \end{aligned} \tag{8}$$

Since $\gamma_{\beta(x)}(x)$ is the minimal depth among the shortest paths from s to x (of weight $\beta(x)$), and $P^{(1)}$ is a shortest path of the minimal depth to x as a prefix of P:

$$\begin{aligned} \gamma_{\beta(x)}(x) &= |P^{(1)}| \\ \gamma_{\beta(x)}(x) + |w(x, y)| &= |P^{(1)}| + |w(x, y)| \\ \gamma_{\beta(x)}(x) + |w(x, y)| &\leq |P^{(1)}| + |w(x, y)| + |P^{(2)}| \\ \gamma_{\beta(x)}(x) + 1 &\leq |P| \end{aligned} \tag{9}$$

At the begining of the current iteration, `ExtractMin(Q)` picked v_i while y was a candidate. Given the ordering in the priority queue, we have:

$$d(v_i) \leq d(y), \tag{10}$$

Combining these results, we get:

$$\beta(v_i) \overset{(1)}{\leq} d(v_i) \overset{(10)}{\leq} d(y) \overset{(7)}{\leq} \beta(x) + w(x,y) \overset{(8)}{\leq} w(P) \overset{(3)}{=} \beta(v_i) \overset{(1)}{\leq} d(v_i).$$

Thus $\beta(v_i) = d(v_i)$, as we wanted to prove, and

$$d(v_i) = d(y). \tag{11}$$

$$d(v_i) = \beta(x) + w(x,y). \tag{12}$$

Equations (11) and (12) imply that:

$$d(y) = \beta(x) + w(x,y). \tag{13}$$

Recall that when x was added to S, the algorithm considered updating $d(y)$ using $d(y) \leftarrow d(x) + w(x,y) \overset{(5)}{=} \beta(x) + w(x,y)$. From (13) we know that $d(y)$ remain unchanged from the iteration where x inserted into S up to the begining of the current iteration. Thus, since $x \in S$,

$$\ell(y) \leq \gamma_{\beta(x)}(x) + 1. \tag{14}$$

Recall that `ExtractMin(Q)` picked v_i while y was a candidate. Given the ordering in the priority queue, and the knowlege that $d(v_i) = d(y)$ (equation 11):

$$\ell(v_i) \leq \ell(y), \tag{15}$$

$$\gamma_{\beta(v_i)}(v_i) = \gamma_{d(v_i)}(v_i) \overset{(2)}{\leq} \ell(v_i) \overset{(15)}{\leq} \ell(y) \overset{(14)}{\leq} \gamma_{\beta(x)}(x) + 1 \overset{(9)}{\leq} |P| \overset{(4)}{=} \gamma_{\beta(v_i)}(v_i).$$

Thus $\gamma_{\beta(v_i)}(v_i) = \ell(v_i)$, as required.

Now, we demonstrate that the path from s to v_i in the tree that `prev[]` restricted to $\{S_i\}$ defines has a weight of $\beta(v_i)$ and a depth of $\gamma_{\beta(v_i)}(v_i)$. In this iteration of the algorithm, the vertex v_i was added to the set S. Consequently, the tree that `prev[]` defines is updated to include the edge $(prev[v_i], v_i)$. Recall that we have already proved that $\beta(v_i) = d(v_i)$, $\gamma_{\beta(v_i)}(v_i) = \ell(v_i)$. Since the last vertex that updated $d(v_i)$ is $prev[v_i]$, we have the relations $d(prev[v_i]) + w(prev[v_i], v_i) = d(v_i)$, and $\ell(prev[v_i]) + 1 = \ell(v_i)$ because after $prev[v_i]$ updated them, these values remained unchanged. Thus,

$$\beta(v_i) \overset{\text{proved}}{=} d(v_i) = d(prev[v_i]) + w(prev[v_i], v_i) \overset{\text{I.H.}}{=} \beta(prev[v_i]) + w(prev[v_i], v_i),$$

$$\gamma_{\beta(v_i)}(v_i) \overset{\text{proved}}{=} \ell(v_i) = \ \ell(prev[v_i]) + 1 \overset{\text{I.H.}}{=} \gamma(prev[v_i]) + 1.$$

(I.H. stands for inductive hypothesis).

This completes the proof that the path from s to v_i in the tree that `prev[]` restricted to $\{s\}$ defines has weight of $\beta(v_i)$ and depth of $\gamma_{\beta(v_i)}(v_i)$.

Theorem 1. *At the completion of the algorithm, for every $v \in V$, we have $d(v) = \beta(v)$, $\ell(v) = \gamma_{d(v)}(v)$, and the array `prev[]` defines a Minimal Depths Shortest Path Tree (MDSPT).*

Proof. At the end of the algorithm, $Q = \emptyset$ and $S = V$. Therefore, by Lemma 3, the conditions of this theorem are satisfied.

5 Running Time Analysis

This algorithm differs from the original Dijkstra's algorithm only in the `relax` method and the `ExtractMin(Q)` operation. Since the running times of these operations remain unchanged, the overall running time of the algorithm is equivalent to that of the original Dijkstra's algorithm, which is $O(|E| + |V| \log |V|)$ when implemented with Fibonacci heap (see, e.g., [1]).

References

1. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms, 3rd Edition. MIT Press, 2009.
2. E. W. Dijkstra. A note on two problems in connexion with graphs. Numer. Math., 1(1):269–271, 1959.
3. Yefim Dinitz, Shlomi Dolev, and Manish Kumar. Polynomial time k-shortest multi-criteria prioritized and all-criteria-disjoint paths - (extended abstract). In Shlomi Dolev, Oded Margalit, Benny Pinkas, and Alexander A. Schwarzmann, editors, Cyber Security Cryptography and Machine Learning - 5th International Symposium, CSCML 2021, Be'er Sheva, Israel, July 8-9, 2021, Proceedings, volume 12716 of Lecture Notes in Computer Science, pages 266–274. Springer, 2021.

LLMs Hallucinations Prevention via Automated Reality Confrontations:*

Case of ChatGPT Knowledge in Dataset Feature Enhancement

(Preliminary Version)

Shlomi Dolev and Ilia Mashevitsky**

Department of Computer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel

Abstract. The problem of (re)generating features automatically to improve machine learning classification results has been extensively studied. Traditional approaches have used public knowledge in their analyses. In this work, we present an automated system that leverages the capabilities of the LLM ChatGPT, incorporating public knowledge through prompt engineering. This system has general-purpose potential and is demonstrated on a specific dataset, showcasing its novel and pioneering achievements. We use the above task to demonstrate our anti-hallucination framework based on repeated feedback from confrontations with (tested/examined) reality.

1 Introduction

As we all know, ChatGPT may give wrong answers. We refer to them as hallucinations. Techniques for coping with the hallucinations of LLMs are needed. Usually, an LLM user has to investigate the foundations of an answer, which frequently results in an apology" from the LLM "realizing that the answers were wrong and conflicting with reality. One may wish to make the verification of an LLM answer automatic, confronting the answer with reality checks.

We demonstrate such an automatic reality check in the scope of enhancement of datasets by common knowledge. Humanity accumulated knowledge that should contribute to machine learning, integrating features and enhancing features to gain better classifications.

We enhance a previous work [5] by introducing a loop of LLMs activity for enhancing a dataset. This loop includes a test of the result by activating machine learning over the enhanced dataset so that the quality of the classification results is fed back to the LLMs.

The goal of [5] was to demonstrate the usage of ChatGPT [7], in conjunction with other external knowledge sources like Wikipedia. The study aimed to showcase feature enhancement by manipulating dataset features and their corresponding data columns, ultimately improving classification tasks using well-known machine learning algorithms. Two specific datasets are considered in [5]. The public knowledge from ChatGPT is acquired manually, with the need for a lot of environment explanations about both specific dataset cases in [5]. The datasets chosen to be analyzed were more suitable for manual research towards feature engineering.

In [2], an automated system was developed for feature enhancement. While the dataset showcased is the same as what was analyzed in our paper, we are using the external knowledge organized in the ChatGPT LLM. As a result, the application developed in this paper [6] is original and comprehensive. Harari and Katz [4] compare various systems for feature generation from specific external knowledge sources. The study emphasizes that our synthetic dataset and the one used in [2] are part of their comparative analysis. Their study statistically analyzes approximately ten datasets.

Our experiments were conducted after the release of ChatGPT. However, it is possible that the DBpedia case study analyzed in their research may no longer be applicable in our context. The application of our system to other datasets, including those in [4], is not expected to yield satisfactory results due to the specific characteristics of those datasets. This prediction applies even to datasets that were not directly accessible to us. The unique nature of various data collections often requires tailored

* Partially supported by the Rita Altura trust chair in computer science, the Israeli Smart Transportation Research Center (ISTRC), and Israeli Science Foundation (Grant No. 465/22).

** Corresponding autor: mashevit@post.bgu.ac.il.

approaches, and our system's effectiveness may be limited when applied to datasets with significantly different properties or structures from those it was designed to handle. The scope of this research yields promising results for datasets exhibiting specific characteristics, which will be comprehensively detailed in the subsequent section. The analysis also noticed an overfitting of the original dataset.

The inner loop that checks for hallucinations also proves the unexplainable nature of ChatGPT. We notice that the same set of general sequences of prompts for ChatGPT API gives desirable results after a few runs. That raises the assumption that ChatGPT needs to focus between runs on different work cycles when iterating its proposed features. The natural language output underwent specialized processing to extract meaningful features. This process involved a series of systematic steps, culminating in generating a new feature column. This derived feature was subsequently utilized as input for evaluation using machine learning algorithms. In [8], the notion of *Informed learning* about incorporating general external aids of information to improve ML was coined.

2 Research Flow

This research work comes as a further step to [5]. The mission there was to automate the process of feature enhancement completely for any dataset using LLM APIs. The initial conceptualization involved integrating multiple LLM APIs with web connectivity, leveraging a distributed computing approach. This strategy was designed to harness comprehensive, publicly available knowledge resources. The proposed system envisioned inter-API communication to enhance result accuracy and reliability through collaborative information processing and cross-validation. However, the hands-on programming work yielded much more modest results. Nevertheless, we did have results. We would like to compare the achievements to programming prompt engineering in low-level programming. It is crucial to differentiate our specific implementation, which was developed as a proof of concept in Python. This paper aims to elucidate the system architecture and functionality, emphasizing the prompt engineering techniques employed for interaction with the ChatGPT 3.5 API model. The budget of iterations all along the way was negligible. Until now, the code designed to gather statistics concerning the non-determinism of ChatGPT has not been programmed. The data is stored in the file system rather than in an SQLite database.

3 Dataset Considerations

Prompt engineering is the art and science of crafting precise and effective instructions for AI language models. The pipeline of prompt engineering in our application is constrained to the Zoo dataset [1], which contains semantic columns of animals, along with features corresponding to them, like the presence of a tail or hair or a number of legs. It is a synthetic dataset from 1990, with classes represented by numbers in the range [1,7]. Each type represents a set of animals described also in a separate natural language file. The dataset is stored as a CSV (comma-separated values) file. Our system could work with datasets of similar characteristics, like explicit semantic columns and the same number of rows (100-1000). Another dataset that may be considered is the Titanic dataset [3]. In order to extend our application to handle this additional dataset, a decision branch by prompt should be inserted at the start of the application. If the result is the Titanic dataset –, a new prompt system should be used. Moreover, a substantially different set of engineered prompts should be used. ChatGPT, for example, can emit an answer of combining SibSp (siblings/spouses aboard) and Parch (parents/children aboard) to create a new family-size feature. The Zoo dataset, hereafter referred to as the primary dataset, resulted from a thorough exploration of various sources, and its selection was a critical component of the research process.

The following analysis and conclusions of this study are based on this dataset. The dataset has been thoroughly validated and reflects the conditions necessary for the experiments conducted in this paper.

Further work could be to automatically generate synthetic datasets like this primary dataset from 1990 [1] that we found. The system is proof to LLM hallucination non-determinism towards the various features it iterates on, both syntactically and semantically. Therefore, the syntactically wrong output of Not a Number is treated as a hallucination. Moreover, output with low accuracy quality is **also** treated as a hallucination.

We work in small steps of prompts engineered to bring to desired accuracy the machine learning result. A module to deal with offline correction of results is also present and is to be explained, along with the work done on low-level prompt engineering and the necessity of each small step of the prompts sent to API along with the variable text inside them. The queries themselves emitted to the ChatGPT server are long due to String variables embedded in the construction of each step, which sometimes are final requests from connection to the ChatGPT method.

4 Methodology

This section describes the feature enhancement process using the ChatGPT-3.5 API.

4.1 Initialization

We start by getting input files and initializing necessary variables.

Algorithm 1 Initialization Step

1: **Input:** *Dataset files:* `DatasetCsv`, `DatasetNames`
2: $Csv \leftarrow$ `DatasetCsv`$Path$
3: $Names \leftarrow$ `DatasetNames`$Path$

Algorithm 1 receives the dataset in CSV format *DatasetCsv*, and in addition, dataset feature names and class names in natural languages *DatasetNames* (line 1). The algorithm assigns the inputs to the variables *Csv* and *Names* (lines 2-3).

4.2 Getting Suggested Feature Array From API

Algorithm 2 Feature Aquisition

1: **Input:** *Names* ▷ A Natural Language Description of Dataset
2: *currentPrompt* ← Engineered prompt using *Names*
3: *currentFeatures* ← **CallAPI**(*currentPrompt*)
4: *processingPrompt* ← Engineered prompt using *CurrentFeatures*
5: *commaSaperatedAns* ← **CallAPI**(*processingPrompt*)
6: *answer* ← *commaSaperatedAns*
7: **Return** *answer* ▷ Return string of features for further work

Algorithm 2 is designed to acquire features for further processing. It takes a natural language description of a dataset as input, generates a series of engineered prompts, and calls an API twice to process these prompts and extract comma-separated features. Finally, the features are returned as a string for further work.

4.3 Getting Semantic Column in Comma Separated Format

Algorithm 3 Semantic Column Acquisition

1: **Input:** *Csv* ▷ Path to the main CSV dataset file
2: *file_path* ← *Csv* ▷ Set the file path (e.g., "../zoo/zoo.data")
3: **open** *file_path* **as** *file* ▷ Open the file for reading
4: *file_content* ← *file.read()* ▷ Read the entire file content as a single string
5: *prompt0* ← `"I have the following csv: "` + *file_content*
6: + `" output string with animal names (first column) separated by ', '"` ▷ Create the first prompt string
7: *prompt1* ← GET_ANS(*prompt0*) ▷ Generate the response to the prompt
8: **return** *prompt1*

Algorithm 3 provides a systematic method for extracting semantic information from a CSV dataset. Specifically, it focuses on retrieving the first column, assumed to contain meaningful entities, such as names, and outputs these values in a comma-separated format.

The following table (Table 1) describes part of the attributes of the primary dataset zoo with 3 of the semantic column animals, including their name, number of legs, presence of a tail, and type classification.

Table 1. Initial Features.

Name	Legs	Tail	Type
Bass	0	1	4
Bear	4	0	1
Boar	4	1	1

The following table (Table 2) describes the final result of one iteration with a new feature from LLM external knowledge - Habitat, saved to the file system. Note that the type classification is consistent with Table 1, but the habitat provides a new dimension of information. This results in a check of reality by an ML algorithm just to this feature of above 0.6 accuracy. It is worth pointing out that text categories like a forest or aquatic change to numerical values during iteration.

Table 2. Additional Feature.

Name	Habitat	Type
Bass	3	4
Bear	4	1
Boar	4	1

4.4 Iterative Feature Transformation and Analysis

This step outlines the main iterative process that transforms the dataset by generating new features, validating them against specified conditions, and applying machine learning or other analytical methods to evaluate the results.

Algorithm 4 Iterative Refinement with Feature Analysis

Input: Array A, size n, File Csv ▷ Initial inputs - Csv dataset, Array of emitted new features
for $i \leftarrow 1$ to n **do** ▷ Iterate over all elements of A
    Initialize $j \leftarrow 0$
    while $j \leq 3$ **do**
        $D' \leftarrow$ Transformation of Csv and $A[i]$ using steps T
        Test condition $T \leftarrow$ check with reality D'
        if T is **False then**
            Increment $j \leftarrow j + 1$
        else
            break ▷ Exit to the next 'for' element
        end if
    end while
end for
Apply **ML Classification** or **Analysis Algorithm** on Csv with *new features from* D'
Return Classification Results or Analysis Outcomes ▷ Return final analysis results

Algorithm 5 High-Level Steps $t_1, t_2, \ldots, t_k$

Input: Data D
Output: Processed data D'

t_1: Get categories for current feature D
t_2: Couple categories with semantic array
$\vdots$
t_k: Analyze the processed data offline D'
Return D' analizys and save to file system results

Example of Prompt Engineering for ChatGPT API

This part explains the steps involved in constructing a prompt for the ChatGPT API to improve classification datasets by generating features for a given list of semantic columns, which, in our case, are animals.

Prompt Construction

The goal of prompt engineering is to construct a sequence of queries that allow the model to generate meaningful features for a list of animals, considering constraints such as feature prevalence and relevance. Below is a step-by-step explanation of the process.

Step 1: Initial Prompt

We define the base structure of the prompt with a list of animals:

$$\texttt{prompt\_01\_01} = \text{“I have the following list of animals } \{\texttt{animal\_list}\}\text{”} \tag{1}$$

Here, `animal_list` is a variable containing the input list of animals.

Step 2: Contextual Extension

We introduce additional context to guide the model:

$$\begin{aligned}\texttt{prompt0\_cont} = \text{“for further improvement of classification dataset work.}\\ \text{Include not more than top } \{\texttt{num}\} \text{ most prevalent feature categories}\\ \text{for all list values. One feature per list value.”}\end{aligned} \tag{2}$$

The variable `num` specifies the maximum number of features to consider.

Step 3: Mid-level Prompt Assembly

The next step combines the contextual extension with specific feature extraction instructions:

$$\begin{aligned}\texttt{prompt\_mid} = &\text{“Output } \{\texttt{feature}\} \text{ features } \{\texttt{prompt0\_cont}\}.\\ &\text{Output original list value with matching feature name.”}\end{aligned} \tag{3}$$

The variable `feature` indicates the type of feature to extract.

Step 4: Comprehensive Prompt Construction

We merge the initial prompt and the mid-level prompt to create a comprehensive query:

$$\texttt{prompt2} = \texttt{prompt\_01\_01} + \texttt{prompt\_mid} \tag{4}$$

Step 5: Feature-Specific Refinement

A refinement step is added to focus on specific features:

$$\texttt{prompt\_only} = \text{“Include } \{\texttt{feature}\} \text{ only for animals in list.”} \tag{5}$$

Step 6: Final Prompt Assembly

The final prompt combines all components to create the input for the model:

$$\texttt{prompt\_tmp\_fin} = \texttt{prompt2} + \texttt{prompt\_only} \tag{6}$$

Step 7: Model Interaction

The constructed prompt is passed to the function `get_ans`, which interacts with the ChatGPT API to generate the desired output:

$$\texttt{prompt1} = \texttt{get\_ans}(\texttt{prompt\_tmp\_fin}) \tag{7}$$

5 Conclusion

This structured approach ensures that the prompt is detailed and specific, allowing the model to generate relevant and accurate features for the given list of animals. The process can be extended or adapted for similar tasks that require precise, prompt engineering. The feedback loop that we have constructed that prevents hallucinations can be useful in other scopes, such as validating programs or formulas.

Acknowledgment. We thank Dani Berend for the many hours of interactions and inputs.

References

1. Richard Forsyth. Zoo. UCI Machine Learning Repository, 1990. DOI: https://doi.org/10.24432/C5R59V.
2. Sainyam Galhotra, Udayan Khurana, Oktie Hassanzadeh, Kavitha Srinivas, Horst Samulowitz, and Miao Qi. Automated feature enhancement for predictive modeling using external knowledge. In *2019 International Conference on Data Mining Workshops (ICDMW)*, pages 1094–1097, 2019.
3. Yasser H. Titanic - machine learning from disaster dataset, 2017. Accessed: 2024-12-16.
4. Asaf Harari and Gilad Katz. Automatic features generation and selection from external sources: A dbpedia use case. *Information Sciences*, 582:398–414, 2022.
5. Arnon Ilani and Shlomi Dolev. Invited paper: Common public knowledge for enhancing machine learning data sets. In *Proceedings of the 5th Workshop on Advanced Tools, Programming Languages, and PLatforms for Implementing and Evaluating Algorithms for Distributed Systems*, ApPLIED 2023, New York, NY, USA, 2023. Association for Computing Machinery.
6. Ilia Mashevitsky. resnew: A github repository. https://github.com/mashevit/resnew, 2024. Accessed: 2024-12-09.
7. OpenAI. Chatgpt: Chat generative pre-trained transformer. https://www.openai.com/chatgpt, 2024. Accessed: 2024-07-14.
8. Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Michal Walczak, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, Jochen Garcke, Christian Bauckhage, and Jannis Schuecker. Informed machine learning - a taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge and Data Engineering*, PP:1–1, 05 2021.

Entrepreneurship Pitch Track chaired by: Yonah Alexandre Bronstein

Entrepreneurship Pitch Track chaired by: Yonah Alexandre Bronstein

The Hi-Tech industry and state-of-the-art research are getting ever closer, as shown by the overlap between the PhD track and Entrepreneurship track this year. The goal of the CSCML Pitch Track is to expose researchers to the world of entrepreneurs and vice versa, for the sake of creating mutual value and advancing the economy and society. All these entrepreneurs deserve all the encouragement that we in the community can give them, in whatever form is suitable. As was the case last year, the Entrepreneurship Pitch track at CSCML 2024 did an excellent job of fulfilling this objective and consequently was a great success. It received endorsement from leading VCs and corporations (IBM, Microsoft, Checkpoint…).

Overall, the quality and value of the start-ups who presented were quite impressive. I look forward to future CSCML conferences in the years to come, still on Zoom as IMHO, it makes it so much easier and cheaper for people around the world to present (and attend ☺).

Best regards,

Yonah Alexandre Bronstein

Entrepreneurship Pitch Track Chair

Development of Assistive Technologies for injured IDF veteran

Shlomi Arnon

This work presents the development of assistive technologies tailored to meet the unique needs of individuals with limb amputations, specifically focusing on IDF (Israel Defense Forces) veterans.

Two distinct projects are highlighted. The first project involves the design and construction of an adaptive PlayStation controller for individuals with upper limb amputations, enabling inclusive access to gaming and its associated cognitive, social, and therapeutic benefits.

The second project introduces a robotic system designed to assist individuals with lower limb amputations by facilitating the transportation of objects within home environments, enhancing their autonomy and quality of life.

These projects integrate innovative engineering, user-centered design, and cutting-edge robotics to address critical challenges faced by amputees, demonstrating significant potential for broader applications in assistive technologies.

StarDox

STARGO

Our Vision

Enabling the supply chain industry to operate with near 100% accuracy by applying our expertise in AI, MLOps, and LLMs to static, profit-draining freight and supply chain processes

Industry Pains

Unstructured data & data management automation are major pains for the transportation industry.

90% of the processing is still manual and 80% of the data is unstructured, resulting in:

x2 Human Error

30% Data management inaccuracies

50% Of employees time is wasted

The Stargo Solution

- Our platform leverages AI and ML technologies, transforming unstructured and semi-structured data into structured data from any source, format, and language.
- Combining Gen AI technology with one-click automation, we take tasks from hours to minutes and unlock hidden margins and revenue.
- We've built the world's first LLM that is fluent in freight and logistics for full data enhancement, correction, enrichment, and validation.
- Our LLM learns a customer's supply chain network and generates insights and recommendations for supply chain enhancement.

Use Cases

Function	Procurement	Sales	Operations	Compliance	Finance	Management	Insurance	Sustainability
Use Case	Carrier Allocation Buy Price RFQ Buy Price-Spot Buy Price-Contracts	Shipper's Quote/Proposal Shipper's Spot Price Request	Carrier Allocation PO Management Booking Request Shipping Instructions	Export Invoice Import Invoice	Carrier Invoice Audit	Predictions Analytics	Import Export	Carbon Footprint Tracking Route Optimization
Mode	✈ 🚢 🚚	✈ 🚢 🚚	✈ 🚢	✈ 🚢	✈ 🚢 🚚	✈ 🚢 🚚	✈ 🚢 🚚	✈ 🚢 🚚

Company Roadmap

2016-18	2021	2022	2023
Start of activity	**AI and ML** for freight use cases	Round B3 **US $20M**	Round B4 **US $2M**
Round A **US $5M**	**LLM** development start	**8 PoCs:** UPS #2-4, Rhenus, Nagel Group, Hellman and GCL	**6 PoCs:** UPS, Forto, EMO, GCL #2, Pilot and MSC
Round B1-B2 **US $12M**	**PoC #1: UPS**	**Commercial agreement with:** UPS	**Commercial agreement with:** Logistics.Cloud and GCL

Company Valuation

$12M Currently raising

SAFE round Pre-money valuation*

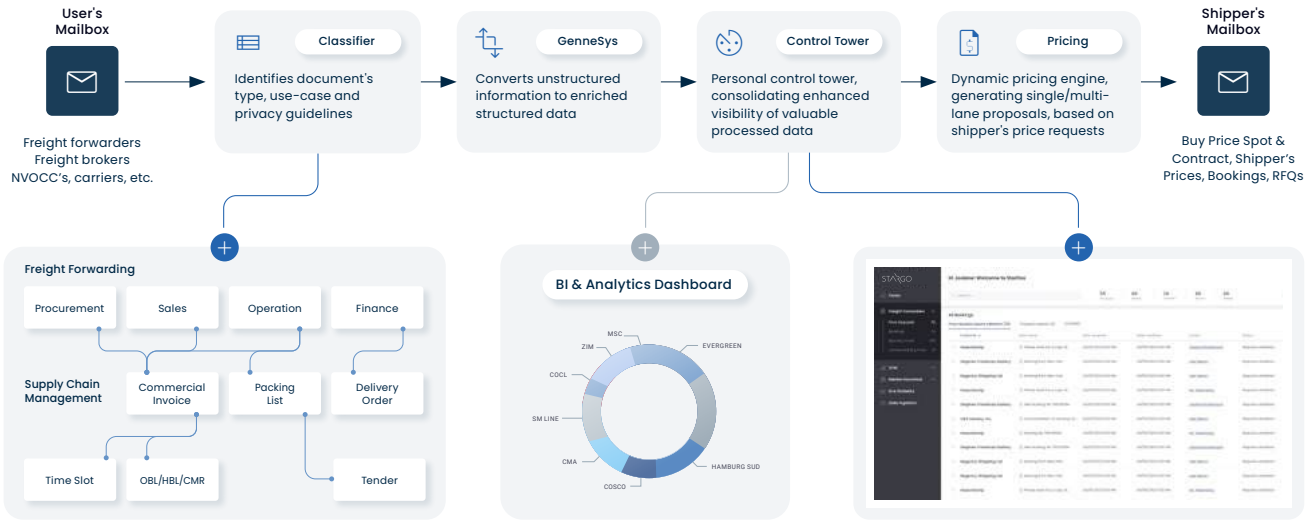
$35M+ Funding up to date

*Terms will be shared with interested parties

Cutting Edge Technology

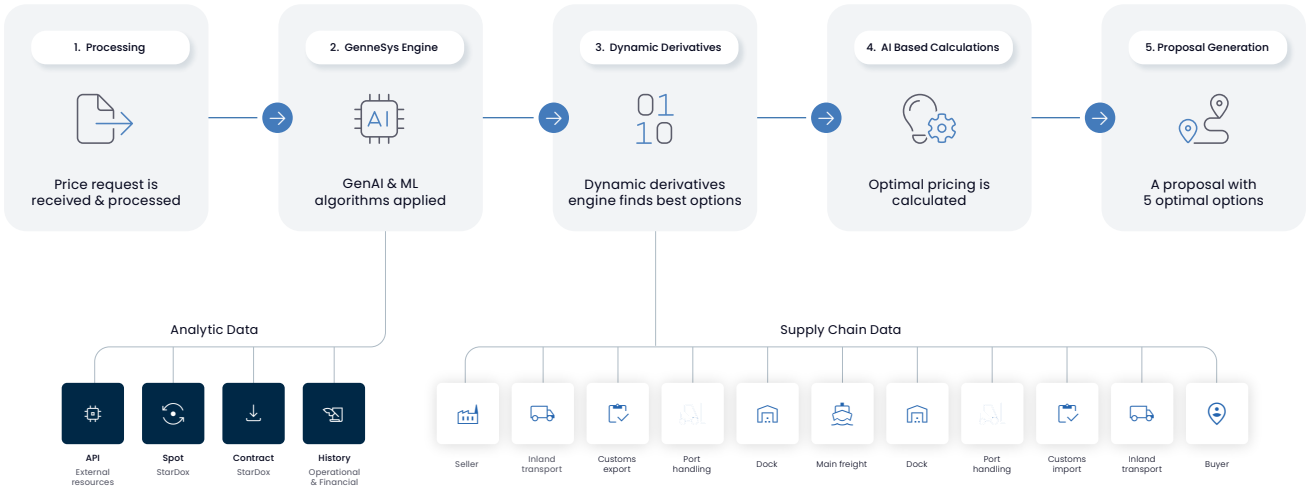
StarDox GenAI Automation and Digitization

Robust StarDox Classifier and GenneSys Engine, with the support of GenAI, our system automates the extraction, cleansing, enrichment, and structuring of all business data, contextualizing it seamlessly



Experience Innovation

StarDox Driven by GenAI for Dynamic Derivatives Pricing



Leading Clients and Partners

Lobster · SOVEREIGN · ups · mastercard · GCL · LOGO LOGISTICS | DIGITAL | SOLUTIONS

StarDox GenAI

Stargo Large Language Model (SLLM) is the world's first LLM trained exclusively on freight and supply chain data for advanced contextual understanding of supply chain operations

Harnessing retrieval augmented generation (RAG) and GenAI, SLLM augments all workflows across the supply chain lifecycle, from procurement and sales to operations and compliance, and business intelligence and analytics

ESG and Sustainability

Building more sustainable supply chains has become crucial to meet evolving regional ESG and compliance requirements.

Stargo's AI algorithms calculate CO2 emissions of all possible routes and modes in seconds, giving supply chain organizations the visibility they need to remain flexible and profitable.

Advanced GenAI flags high-emission risks across supply chains for mitigation and better supply chain management

Procurement

- Assess vendor shipping routes and modes to select lower-emission options.
- Partner with more ethical suppliers and become ESG compliant easily.

Sales

- Present real-time comparisons of CO2 emissions for all route and mode options for shipper price requests.
- Analyze and flag multi-lane, shipping proposals for hidden emissions before sending.

Operations

- Identify margin gaps and opportunities using historical and real-time market data.
- Use predictive modeling to flag and mitigate risks.
- Track and analyze supply chain CO2 emissions over time, along with other sustainability metrics.

Compliance

- Monitor current and emerging ESG and compliance regulations across regions.
- Generate accurate compliance reports, and correct and enrich existing reports.
- Unlock greener, more profitable routes while staying compliant.

Value Proposition

Increase productivity, improve conversion rates and maximize margins, with Stargo's GenAI

Sales

- Reconciling buy prices goes from 3 h. to <60 sec. for sales teams.
- Spot carriers buy price inaccuracies to prevent over and under charges.
- Shipping proposal creation costs go from $70 to $3.6.
- Shows you the most optimal rate margins for you and your customer.

Procurement

- Build better vendor relationships with historical data.
- Points out overcharge discrepancies, contract breaches, and rate errors before they harm vendor relationships.
- Identifies acquisition patterns across won proposals to boost procurement rates.

Operations

- Automatically completes missing or incorrect data across all operations.
- Control Tower provides full visibility over all vendor and customer contracts and processes.
- Build knowledge graphs of your existing supply chain networks and provides BI and analytics for revenue enhancement.

Accounting

- Automate data entry: Financial transactions.
- Automated reconciliation processes.
- Enhanced analysis.

Compliance

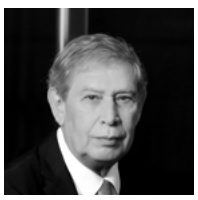
- Regulatory adherence: Automate freight regulations.
- Data Management.
- Comprehensive audit trails of freight movements.

Management

- BI Insights based on market data and supply chain network analysis.
- Demand forecasting, capacity planning and revenue optimization.
- Supply chain risk spotting and mitigation.

Management & Board

Innovators, experts in logistics and technology industries, solving complex challenges in global companies.



Tamir Pardo
Chairman of the Board

Joel Sellam
Founder, CEO & Board Member

Josiane Rosenbaum
COO & Board Member

Daniel Clayman
Board Member

Louis Mayberg
Board Member

David I. Beatson
Board Member

Development of an AI-Based Pain Measurement Monitor (PMM) Using TSSCI Technology

By Dr. Yoram Segal, yoramse@post.bgu.ac.il

Mentors: Dr. Alex Geftler and Prof. Ofer Hadar

Abstract

Our innovative project aims to develop an objective Pain Measurement Monitor (PMM) based on Time-Space single-color image (TSSCI) technology. The system uses cameras to capture patient movements and facial expressions, converting them into abstract TSSCI representations that preserve privacy while enabling accurate pain assessment. The system provides an objective measure of pain intensity by correlating these images with pain medication dosages.

The monitor leverages deep learning algorithms, including EfficientNet-B7 and Siamese neural networks, to analyze the TSSCI data and establish correlations between movement patterns and pain levels. A key advantage is the system's ability to assess pain in patients who cannot communicate effectively, such as those under anesthesia or with cognitive impairments.

We conducted the initial concept at Soroka Hospital with Dr. Alex Geftler, and the potential is promising for many clinical applications. The solution addresses a significant market need, with the pain management devices market projected to reach $5.68 billion by 2032.

The technology offers several unique benefits: contactless monitoring, complete patient privacy through abstract TSSCI representations, and integration with existing hospital camera systems. Our innovative contactless approach has the potential to revolutionize pain assessment in healthcare settings while ensuring patient comfort and dignity.

Reconstruction of The Attack Graph in DDoS Attacks – Algorithms, Simulations and Future Research

Dina Barak-Pelleg, Daniel Berend,
Thomas J. Robinson, and Itamar Zimmerman

December 2024

Outline

Outline

Denial-of-Service (DoS) Attack

- A denial-of-service (DoS) attack is a cyber attack on a computer (or a network resource).
- The attacker seeks to make the victim unavailable to its intended users.
- In the attack, the attacker floods the victim with fake data packets, until it collapses.

Distributed Denial-of-Service (DDoS) Attack

Distributed denial-of-service (DDoS) attack – same as DoS attack but with multiple attackers.

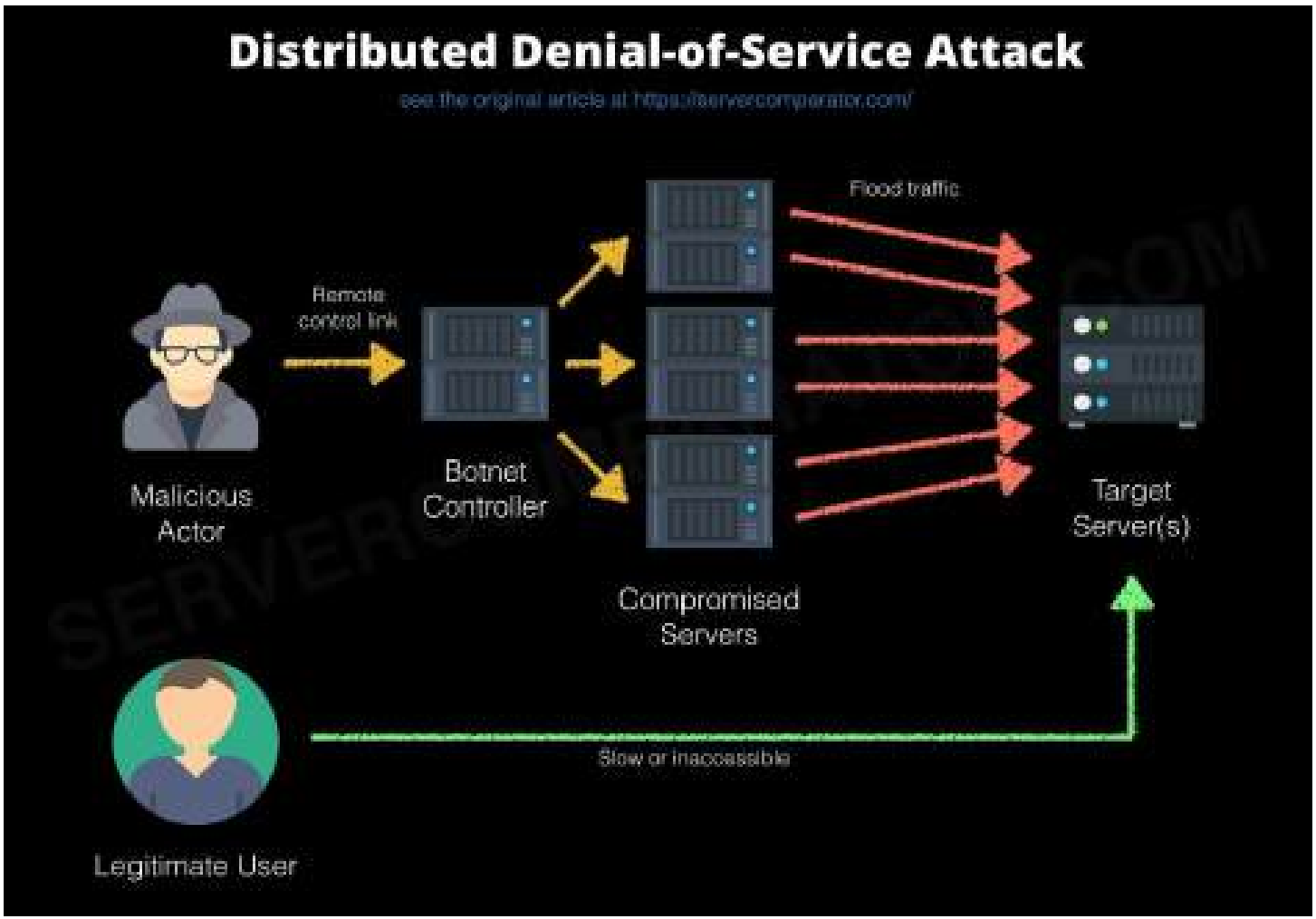
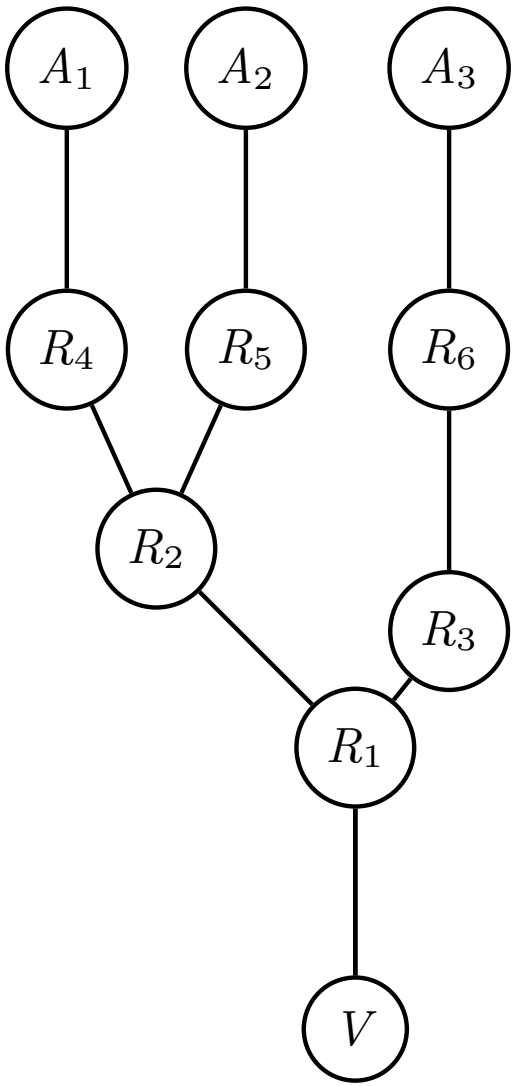


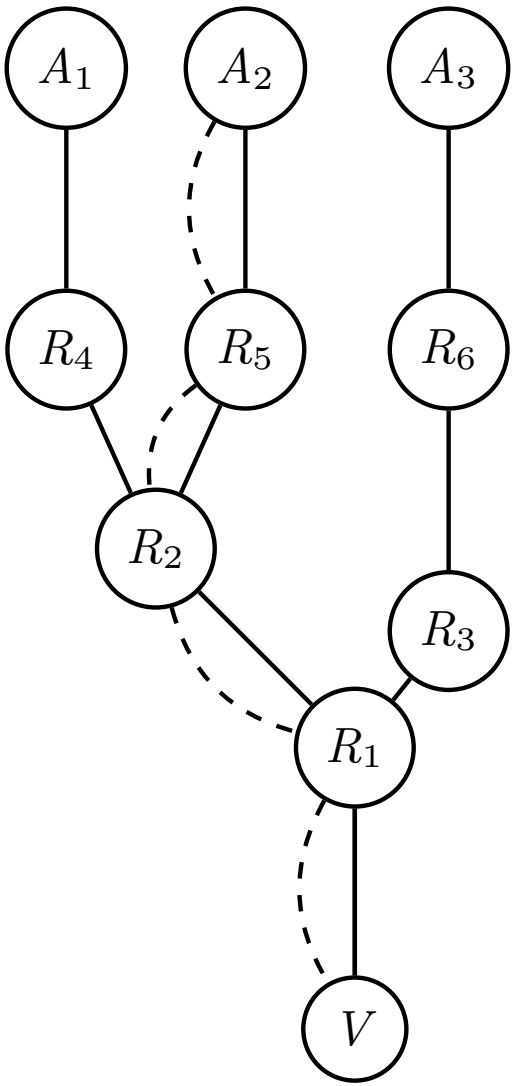
Figure 1: An illustration of a DDoS attack [5]

The So-Called Attack Graph (an Illustration)



- The attack graph is a tree-type graph. The root represents the victim, the leaves – the attackers, and the internal nodes – the routers on the network.
- One of the main defense techniques is as follows:
  - Detect the attackers on the network by reconstructing the attack graph (our focus here).
  - Mitigate the attack by filtering messages coming from the attackers.

Illustration: The Network As Seen by the Victim



- The victim is denoted by V.
- The routers are denoted by R_i, $1 \leq i \leq 6$.
- The potential attackers are denoted by A_i, $1 \leq i \leq 3$.

- Suppose that there is a single attacker, A_2.
- The attack graph is the path A_2, R_5, R_2, R_1, V.

Probabilistic Packet Marking (PPM)

- To deal with these attacks, we would like to get information on the source of the attack.
- It would be good if each packet could carry information of all the routers it had visited.
- It would require too much resources to record the full path each packet travels from each sender to each recipient.
- In Probabilistic Packet Marking, we record this information partially.
- Obviously, if we wait long enough, eventually we get the entire attack graph.

Edge Sampling

- Each router along the path marks each packets with probability $p < 1$.
- The mark consists of the identity of the marking router and the next one, as well as a distance field.
- Former marks, if any, are erased.
- Hence, each mark the victim receives consists of an edge of the graph and its distance from the victim.
- The probability of the victim to receive a packet marked by a router at distance j is

$$p(1-p)^{j-1}.$$

The Reconstruction Time

- Suppose there is a single attacker at distance n from the victim.
- An important question regarding the PPM algorithm is how much time (i.e., how many packets) we need to reconstruct the attack graph.
- We call this variable *the reconstruction time*, and denote it by T.

The Reconstruction Time – Related Work

- Savage et. al. [8] bounded from above the expected reconstruction time (for $p = 1/n$):

$$E(T) \leq \frac{\log n}{p(1-p)^{n-1}} \approx en \log n. \tag{1}$$

- In [1] we have found the distribution of this quantity for $p = \lambda/n$, for $\lambda > 0$. Here we present only its expectation for $\lambda = 1$:

$$E(T) = en\left(\log n - \log\log n + \gamma\right) + O\left(\frac{n \log\log n}{\log n}\right),$$

where $\gamma = 0.577\ldots$ is the Euler-Mascheroni constant.

The Reconstruction Time – Related Work (cont.)

- Saurabh and Sairam [7] suggested to stop the algorithm after receiving

$$\frac{\log n}{p(1-p)^{n-1}} + \frac{1}{3} \cdot \sqrt{\sum_{i=1}^{n} \frac{1 - \sum_{j=1}^{i} p(1-p)^{n-j}}{\left(\sum_{j=1}^{i} p(1-p)^{n-j}\right)^2}}.$$

  packets.

Outline

A Basic Algorithm

- Denote the attack path by $P = e_1, e_2, \ldots, e_n$, $e_i = \{v_{i-1}, v_i\}$.
- At each point in time, he has several edges of P, and perhaps some edges relating to packets received from legitimate users.
- Denote the random subgraph, induced by the edges he has, by G.
- We may assume that most edges he has belong to P.

A Basic Algorithm (cont.)

- Suppose the victim has at some point a maximal subpath $P' = e_1, e_2, \ldots, e_i$.
- (We assume here that $i \geq 2$; an attack path will not consist of a single edge.)
- P' is a *full subpath* if:
  - P' is a subpath $e_1, e_2, \ldots, e_i$, $i \geq 2$.
  - $e_j \notin G$, $i + 1 \leq j \leq n$.
- Suppose the victim has a full subpath P'. What should the victim do next?
- Algorithm 1 simply returns P' as the attack path.

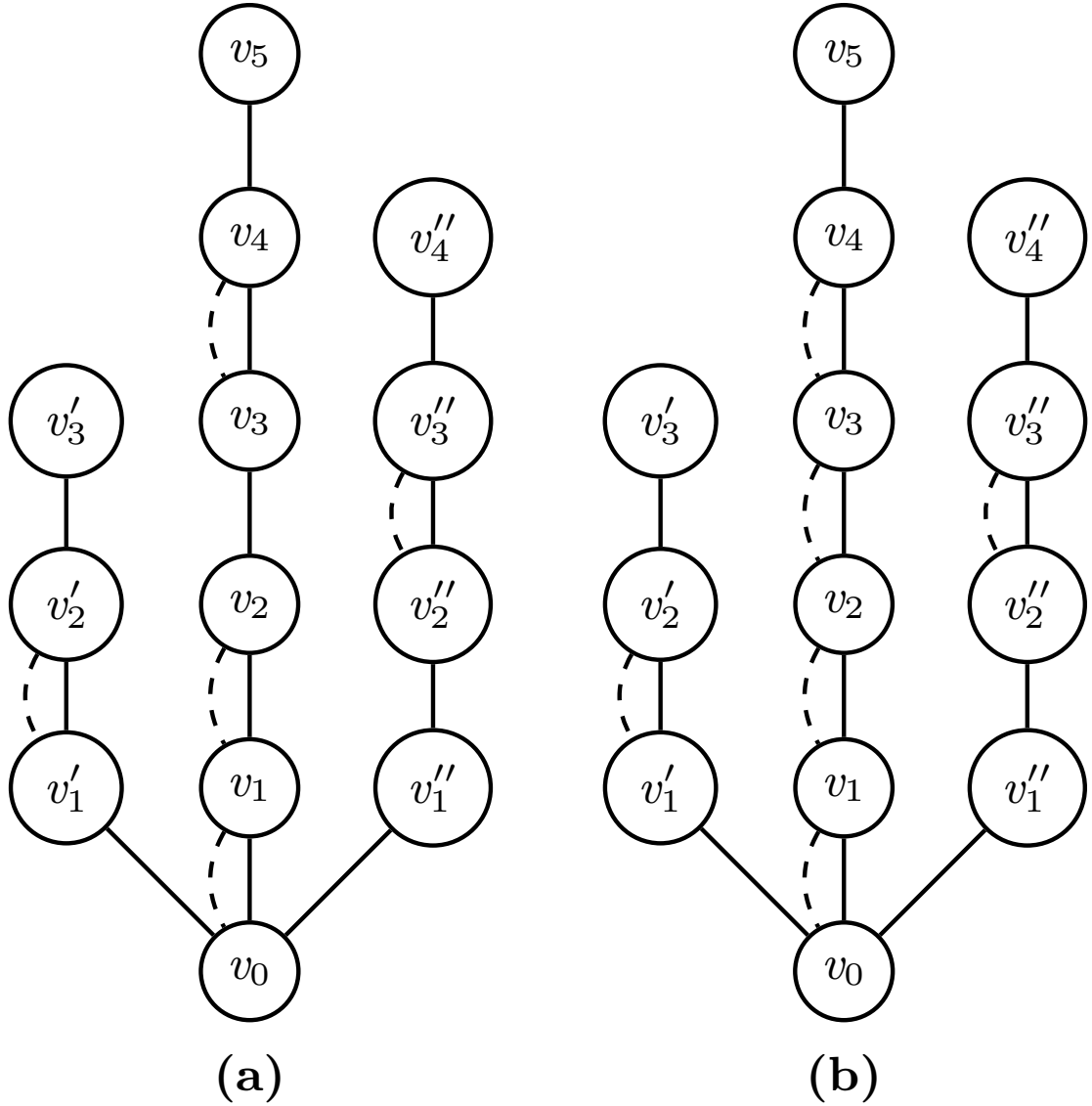


Figure 2: Illustrations of: (a) A maximal subpath, and (b) A full subpath.

Algorithm 1: Basic Main Algorithm

Input: Incoming flow of packets
Output: Attack path

```
G ← ∅ ;                          /* G:random graph from collected edges */
while G does not contain a full subpath P' of length ≥ 2 do
    Receive the next packet;
    if the packet is marked by some edge e_i then
        G ← G ∪ e_i;
return P'
```

The Success Rate of Algorithm 1

In [2], we have proved

Theorem 1

Let $p = 1/n$. The probability that Algorithm 1 does not return the full attack path is

$$O\left(\frac{(\log n)^{1/(e-1)}}{n^{1/e-1/e^2}}\right) = o(1)$$

as $n \to \infty$.

- Our simulations show very good results for the success rate of Algorithm 1 for common values of n as well.

Algorithm 2: A More Accurate Heuristic Algorithm

Input: Marking probability p,
incoming flow of packets,
ε (small parameter controlling the accuracy)

Output: Attack path

$\ell \leftarrow 0$; `/* ` ℓ`:number of collected packets */`

$G \leftarrow \emptyset$; `/* ` G`:random graph from collected edges */`

while G does not contain an ε-full timed subpath P' of length ≥ 2 **do**
    Receive the next packet;
    ℓ ++;
    if the packet is marked by some edge e_i **then**
        $G \leftarrow G \cup e_i$;

return P'

- Let $\varepsilon > 0$ be some small parameter.
- Suppose that we have collected ℓ packets and P' is a full subpath of length j. If
$$(1 - p(1-p)^j)^\ell < \varepsilon,$$
then P' is an ε-*full timed* subpath.

Outline

Simulation – Objective

- We have simulated the process of collecting the markings in a DoS attack.
- The objective – to compare the following five algorithms:
  Algorithm 1,
  Algorithm 2 with $\varepsilon = 0.1$,
  Algorithm 2 with $\varepsilon = 0.05$,
  SWKA [8] – to be explained in the following slides.
  S & S [6] – to be explained in the following slides.

Simulation – Objective (cont.)

Algorithm SWKA [8]:

- Savage et al. [8] have not explicitly stated an algorithm.
- They showed, as presented in (1), that the expected reconstruction time is bounded above (for $p = 1/n$) by

$$\frac{\log n}{p(1-p)^{n-1}} \approx en \log n.$$

- This may be interpreted, and has been indeed interpreted by some [3, 6], to mean that the victim should try to reconstruct the attack path after receiving this number of packets.

Simulation – Objective (cont.)

Algorithm S & S [6]:

- Saurabh and Sairam suggested to collect

$$\frac{\log n}{p(1-p)^{n-1}} + \frac{1}{3} \cdot \sqrt{\sum_{i=1}^{n} \frac{1 - \sum_{j=1}^{i} p(1-p)^{n-j}}{\left(\sum_{j=1}^{i} p(1-p)^{n-j}\right)^2}}.$$

  packets before the reconstruction.

Simulation Description

- In our experiment, the length of the path, the marking probability, and the number of iterations are:

$$n = 25, \quad p = 1/25, \quad N = 10^7.$$

- In each iteration, we collect the marked packets arriving via the path P until we get the full attack path P.

Simulation Description (cont.)

- At the end of each iteration we have recorded the number of packets actually collected until we have obtained all the edges of the attack graph.
- For each of the five algorithms we have recorded:
  1. The number of packets collected.
  2. Whether or not the algorithm has been successful in obtaining P.

Table 1 – Performance of the Algorithms

Algorithm		Average Number of Received Packets	Success Rate
Algorithm 1		167	0.87
Algorithm 2	$\varepsilon = 0.1$	191	0.97
	$\varepsilon = 0.05$	222	0.99
SWKA [8]		214	0.77
S & S [6]		241	0.86

Table 1: Performance of the Algorithms for $n = 25, p = 1/25$.

- Average number of received packets until collecting all edges is 178.

Outline

An Alternative PPM

In edge sampling:

- The probability of receiving a marking from the farthest and nearest routers to the victim are

$$p(1-p)^{n-1} \approx \frac{1}{en}, \quad p = \frac{1}{n},$$

  respectively, for $p = 1/n$.
- Each router, upon receiving a packet, marks it with probability p.
- This probability is the same for packets which are already marked and for those that are not.

An Alternative PPM (cont.)

- The idea of not overriding a marked packet has been suggested by Ma [4].
- Here, each router may mark only unmarked packets. Such packets are marked with probability p.
- This method is a mirror image of the original method.

An Alternative PPM (cont.)

- Consider the following marking model:
- Let $0 < q_1, q_2 < 1$.
- When a packet arrives at a router:

  If the packet is unmarked, then the router will mark it with probability q_1.

  Otherwise, it will mark it with probability q_2.

An Alternative PPM – An Example

- Suppose that $q_1 = 2/n$ and $q_2 = 1/(2n)$.
- The probability of receiving the markings from the farthest router to the victim:

$$q_1(1-q_2)^{n-1} = \frac{2}{n}\left(1-\frac{1}{2n}\right)^{n-1} \approx \frac{2}{\sqrt{e}n}.$$

- The probability of receiving the markings from the nearest router to the victim:

$$\begin{aligned}
&(1-q_1)^{n-1}\cdot q_1 + (1-(1-q_1)^{n-1})\cdot q_2 \\
&\quad = \left(1-\frac{2}{n}\right)^{n-1}\cdot\frac{2}{n} + \left(1-\left(1-\frac{2}{n}\right)^{n-1}\right)\cdot\frac{1}{2n} \\
&\quad \approx \frac{2}{e^2 n} + \frac{1-e^{-2}}{2n}.
\end{aligned}$$

An Alternative PPM – Research Question

- Find q_1 and q_2 such that the expected reconstruction time will be minimal.

Our Algorithms' Performance in DDoS

- As we have shown, our algorithms' success rates are very good for a DoS attack.
- Extend Algorithm 1 and Algorithm 2 to the case of multiple attackers.

Improving Algorithm 2

Our waiting time in Algorithm 2 depends on the number ℓ of packets we have received so far in a very simple way. Namely, based on the length of the full path we have and the accuracy ε, we calculate a required number M of packets we want to have received before making a decision. If $M \geq \ell$, we do not wait anymore. If $M < \ell$, we wait for $M - \ell$ packets, and then, assuming no further edges have been found in the meantime, decide.
We believe that the waiting time can be defined in a better way if we allow it to depend in a less trivial way on ℓ.

Outline

Bibliography I

[1] Dina Barak-Pelleg and Daniel Berend, *The Time for Reconstructing the Attack graph in DDoS Attacks*, Submitted to Advances in Applied Probability.

[2] Dina Barak-Pelleg, Daniel Berend, Thomas J. Robinson, and Itamar Zimmerman, *Algorithms for Reconstructing DDoS Attack Graphs Using Probabilistic Packet Marking*, preprint.

[3] Ankunda R. Kiremire, Matthias R. Brust, and Vir V. Phoha, *Using Network Motifs To Investigate the Influence of Network Topology on PPM-Based IP Traceback Schemes*, Computer Networks 72 (2014), 14–32.

[4] Miao Ma, *Tabu Marking Scheme to Speedup IP Traceback*, Computer Networks 50 (18) (2006), 3536–3549.

Bibliography II

[5] *https://techcommunity.microsoft.com/t5/sql-server/understanding-server-traffic-logs-and-detecting-denial-of/ba-p/385529?lightbox-message-images-385529=98595i2E1A70BBD37EAF04*

[6] Ashok Singh Sairam and Samant Saurabh. *Increasing Accuracy and Reliability of IP Traceback for DDoS Attack Using Completion Condition.*International Journal of Network Security, Vol.18, No.2, 224–234, Mar. 2016.

[7] Ashok Singh Sairam and Samant Saurabh. *A More Accurate Completion Condition for Attack-Graph Reconstruction in Probabilistic Packet Marking Algorithm.* in Proc. IEEE National Conference on Communications (NCC), 2013.

Bibliography III

[8] Stefan Savage, David Wetherall, Anna Karlin, and Tom Anderson, *Practical Network Support for IP Traceback*, Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (2000), 295–306.

Thank you for your attention

Predicting Clinical Trial Readiness of Drug Candidates Using AI

BiolomiX

Drug development is a high-risk endeavor. Nearly 90% of potential drug candidates that enter phase I trials fail to reach the clinic. This problem isn't new to the pharma world. Every year, seemingly endless amounts of resources are wasted on drug candidates of which the majority never reach the market.

Our platform will transform the pharma industry. Capable of ingesting preclinical data of all modalities across multiple species, our platform aggregates all data to identify common and unique biological features. Our solution applies proprietary technology, harnessing the power of network analysis and machine learning approaches.

BiolomiX's unique method harmonizes between data-driven and knowledge-based approaches. Proof of concept studies have shown that our approach is more accurate and effective than other methods. At the core of our platform sits a patented approach – our secret sauce - determining the biological activity of a biological feature. This approach overcomes a key caveat of conventional methods by emphasizing biological activity, as opposed to presence, generating a new data type, we call “activity score”.

Our platform accurately predicts the efficacy and safety of a drug candidate and suggests validation experiments. As a result, pharma companies can significantly de-risk the decision of moving a candidate from preclincal to clinical trials.

BioSOC: Cybersecurity-Inspired Biosecurity Platform for Synthetic DNA Screening

Entrepreneurship Pitch Submission

Contact: Vladislav Kogan, koganv@bgu.ac.il, Beer-Sheva, Israel.

Overview

Screening synthetic DNA orders for select agents and toxins is a critical biosecurity measure to reduce the risks of bioterrorism and reduce the risk of creating dangerous pathogens intentionally or accidentally. This process parallels digital cybersecurity, where human analysts analyze alerts from intrusion detection systems. However, existing biosecurity tools lack the scalability, flexibility, and resilience of digital cybersecurity systems, exposing synthetic biology to potential misuse and insecure designs.

Team

* **Prof. Rami Puzis** - Associate Professor and Pioneer of Biosecurity Research at Ben-Gurion University of the Negev. Inventor of the Gene Editing Distance (GED) approach.[1]
* **Prof. Isana Veksler Lublinsky** - Associate Professor in the Department of Software Engineering and Information Systems at Ben Gurion University. Bioinformatician known for publications in the field of biosecurity. [2]
* **Vladislav Kogan** – Biomedical (B.Sc.) and Information Systems Engineer (M.Sc.), Developer of recombination-based DNA obfuscation and alert correlation algorithm for detection of distributed toxins.
* **Anna Kulakovsky** - Software Engineer (SISE) Developer of the BioSOC academic prototype.

Product

The concept of a Security Operation Center (SOC) with a bioinformatics toolbox enables the detection, triaging and follow-up on suspicious DNA orders. This system integrates SIEM/SOC technology to correlate alerts and detect cases where dangerous DNA sequences may reach the wrong hands even if they are spread across multiple orders. Applying cybersecurity practices strengthens biosecurity by identifying hidden threats in seemingly benign DNA. The system can alert on dangers when recombining DNA products may create significant risks. In cases where dangerous DNA sequences are ordered with a permit the system will streamline the customer experience by retrieving the relevant permits while the order is being placed.

---

https://faramirp.wixsite.com/puzis[1]

linkedin isana-veksler-lublinsky[2]

The platform, designed for Biology Equipped Security Operations Centers (BioSOC), enhances bio-security analysts' efficiency by detecting adversarially manipulated purchase orders. It utilizes a patent-pending algorithm (WO2021165961A1) to assess the probability of converting a DNA sequence into one that produces a select agent or toxin using common DNA editing tools, alongside proprietary technology for correlating selected agent alerts.

Market

The global DNA synthesis market size was USD 4.04 billion in 2023. The market is expected to grow from USD 4.56 billion in 2024 to USD 16.08 billion by 2032, exhibiting a CAGR of 17.1% during the forecast period.[3]

In recent years, the DNA synthesis market has seen a significant trend, with leading companies focusing on the development of advanced products and technologies to synthesize DNA. The rising need for cost-efficient and sustainable synthetic products has led companies to emphasize synthesizing faster, cheaper, and more accurate DNA.

The rising prevalence of chronic diseases and increasing awareness among people for personalized treatment options have led to increasing usage and production of synthetic DNA products and services across the globe.

High market growth can be attributed to the advancements in synthetic biology and gene editing technologies such as CRISPR, coupled with the growing adoption of precision medicine.

Competitors

Aclid Bio (founded 2021) raised $3.3M in a Seed Round on October 12, 2023.[4] The company offers a biosecurity platform that manages the lifecycle of synthetic DNA and RNA production. It screens orders for pathogenic or toxic elements, automates compliance processes, and verifies customer credentials to ensure regulatory standards are met. Key services include screening DNA/RNA sequences and automating compliance for governments, foundries, and biotech providers.[5]

Battelle is a global research organization involved in biosecurity and synthetic biology. It provides biosecurity solutions to screen, manage, and mitigate the misuse of biological agents, advancing genomic research and national security projects globally.[6]

IBBIS is an independent organization with a primary mission dedicated to reducing emerging biological risks associated with technology advances.[7] They developed a tool – an

---

fortune business insights - DNA Synthesis Market[3]
pitchbook- aclid[4]
aclid.bio[5]
battelle[6]
IBBIS[7]

international Common Mechanism for DNA synthesis screening – that DNA synthesis providers can use to screen DNA orders and customers.

IP/PATENTS:

February 2021: Submission of a patent application for the "Gene Edit Distance" (GED) technology.

September 2022: The patent was formally assigned to Ben-Gurion University of the Negev (BGN).

March 2023: Patent applications are pending across several jurisdictions including the European Patent Office (EP), Japan (JP), the United States (US), and under the International Patent Cooperation Treaty (WO).

Planned for 2024: An additional patent filing is anticipated for the advanced alert correlation system.

Regulation

In 2023, the U.S. tightened regulations on synthetic DNA production, specifically targeting shorter sequences. Previously, only sequences of 200 base pairs and above were scrutinized, but now even sequences as short as 50 base pairs are subject to screening. This change aims to prevent the creation of harmful DNA fragments that could be pieced together to produce toxins or dangerous pathogens. These rules apply to federally funded entities and aim to ensure stricter biosecurity in synthetic DNA manufacturing.[8]

In Europe, the regulation of synthetic DNA production focuses on ensuring biosafety by preventing the release of genetically modified organisms and controlling the creation of harmful or toxic DNA sequences. These regulations prioritize environmental protection and the safe use of synthetic biology technologies within contained settings.[9]

According to regulations, only verified entities such as research institutions, universities, biotechnology companies, and government agencies can purchase synthetic DNA. These entities must pass screening to ensure the DNA is used for legitimate purposes and not for creating harmful organisms or toxins.[10]

SynNA-Guidance-2023[8]
regulators [9]
cser - synthetic biology regulation[10]

NErlNet

Distributed ML on the Edge IoT

David Leon, Guy Peretz
Yehuda Ben-Shimol

Distributed ML (DML) on the Edge

Machine learning on the edge, where data processing happens on local devices rather than centralized servers, presents challenges:

Limited Resources

Energy Consumption

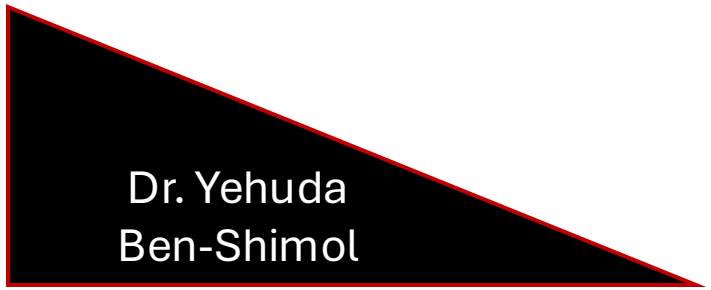
Model Update

Latency and Realtime Processing

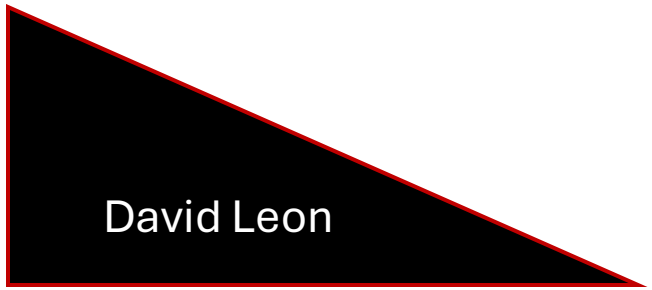
DML – Questions

- How to conduct experiments of ***real-time*** ML models running on IoT devices as a distributed system?
- How to ***collect information*** and statistics of throughput and model performance?
- How to ***control parameters*** of both ML and distributed system?
- How to ***simplify*** the execution of distributed ML cluster that runs on IoT devices?
- How ***data flow*** in the Network influences the capability of online models at the edge to train?

NErlNet Group



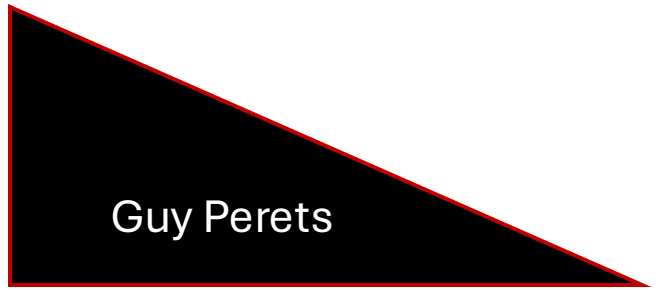
Academic Advisor and Founder of NErlNet



Founder of NErlNet

M.Sc. in CSE at Ben Gurion University

Staff AI Engineer at Analog Devices



R&D of NErlNet

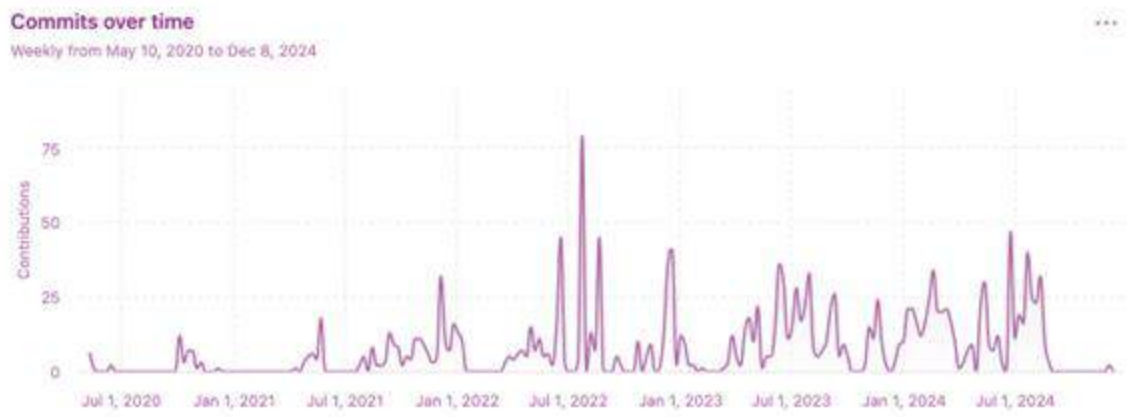
M.Sc. Student at Ben-Gurion University

NErlNet for DML

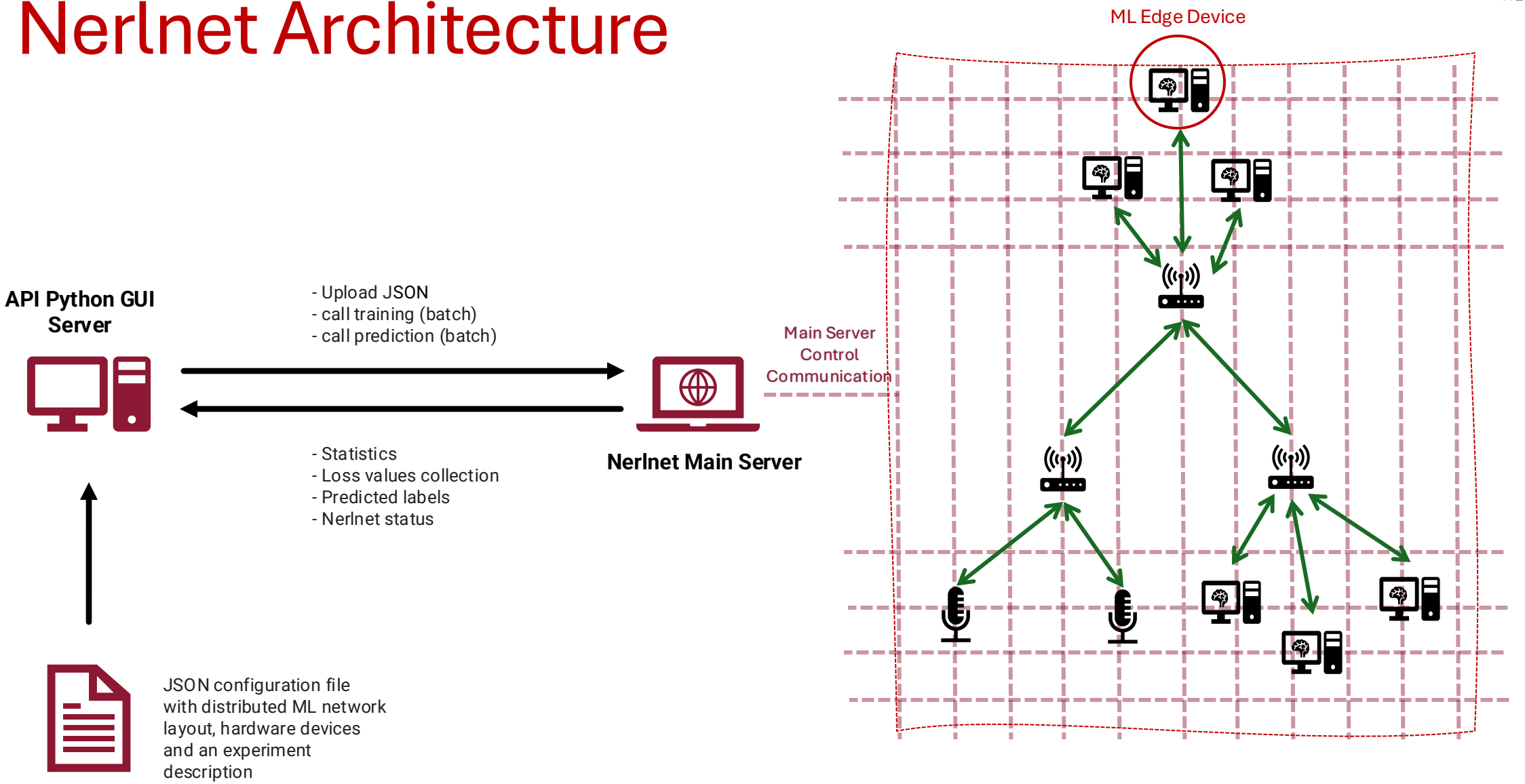
- The main problem is exploring DML models in real-time.
- Edge devices have different architectures and lack support for known Python ML packages..
- Nerlnet connects the design of concurrent systems with ML models at the edge, where data is generated.

The Innovation of Nerlnet

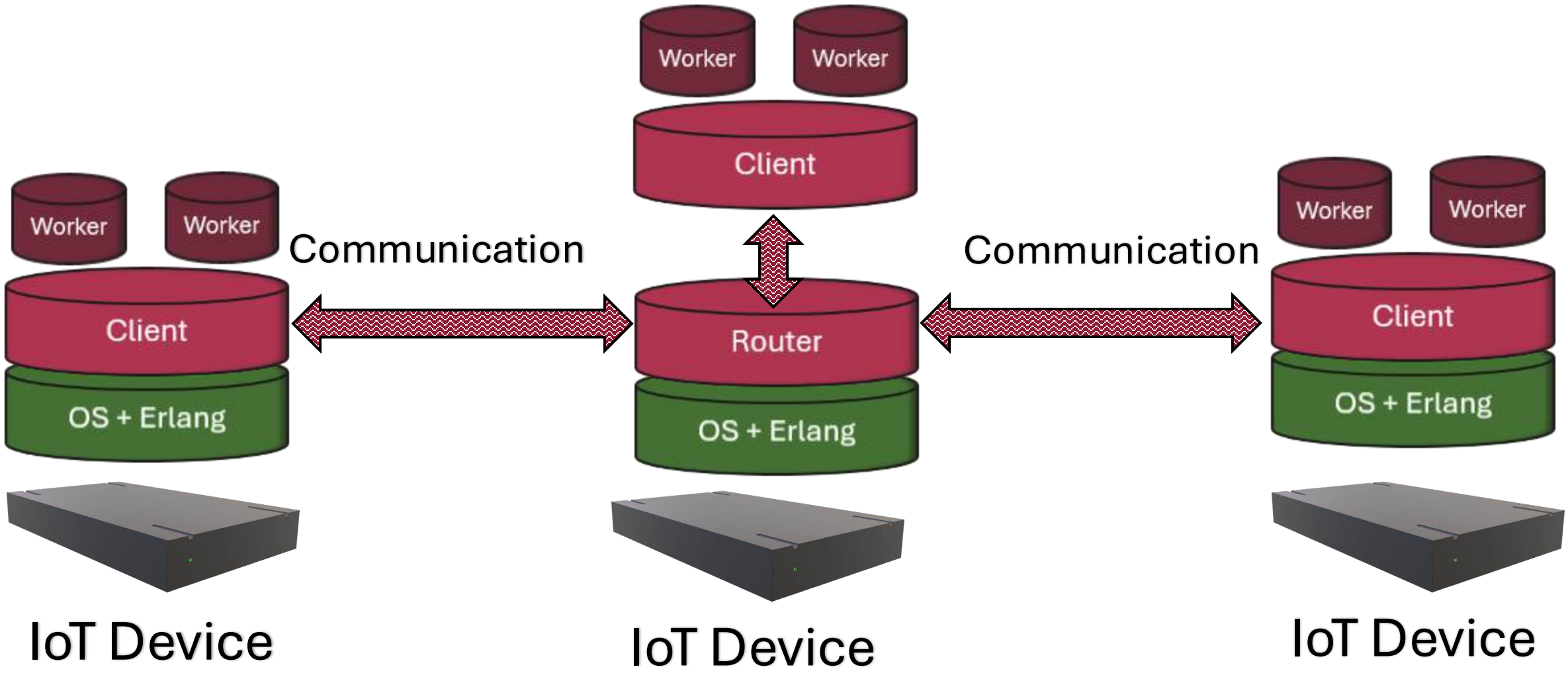
- The Nerlnet project was initiated as a final task in the functional programming in concurrent systems course..
- How can Erlang VM be used for executing ML models concurrently.
- Nerlnet environment treats ML model as an event-driven state machine, a process that can communicate with other processes.



Nerlnet Architecture



NErlNet's Entities



Available Solutions

Feature	Nerlnet	Flower	Scaleout Systems
Open Source	Yes	Yes	Yes
Programming Language API	Python	Python	Python
Programming Language Infra	Erlang / Cpp	Python	Python
Integration	OpenNN, (libtorch upcoming)	Tensorflow / Pytorch	Tensorflow / Pytorch
Requires Python on IoT device	No	Yes	Yes
Federated ML	Limited	Support	Support

Achievments so far

- Yielded two academic (in process of writing).
- Azure credits sponsorships for selected open-source projects – Nerlnet was selected twice, 2023 and 2024.
- AWS credits sponsorship – selected open source project for 2023.

Useful Links

Nerlplanner

GitHub Repo

Experiment Example