TECHNISCHE UNIVERSITÄT DARMSTADT

der Bundeswehr Universität München

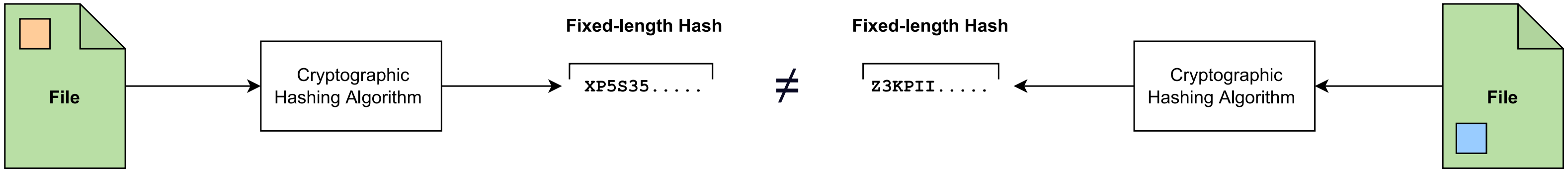
Combining AI and AM - Improving Approximate Matching through Transformer Networks

Digital Forensic Research Workshop (DFRWS) USA 2023

Frieder Uhlig, Technical University Darmstadt, Germany
Lukas Struppek, Technical University Darmstadt, Germany
Dominik Hintersdorf,Technical University Darmstadt, Germany
Thomas Göbel, Universität der Bundeswehr, München, Germany
Harald Baier, Universität der Bundeswehr, München, Germany
Kristian Kersting, Technical University Darmstadt, Germany

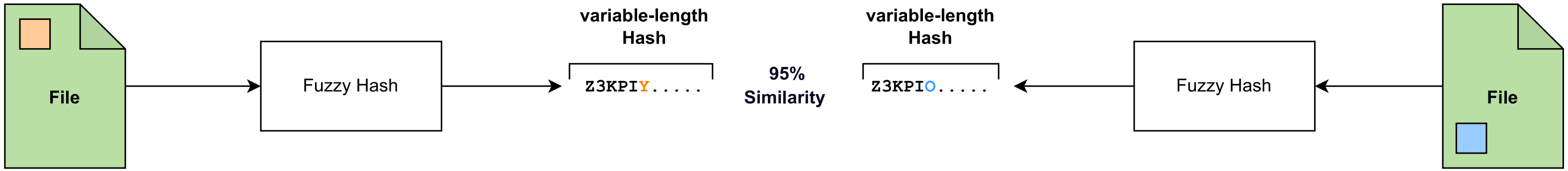
Cryptographic Hashes

- Deterministic, Collision resistant etc.
- Used to verify integrity
- Concise unique representation of a digital artifact



Fuzzy Hashes

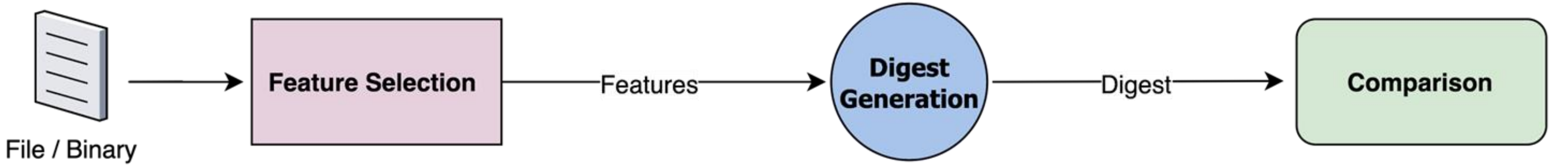
- ➤ Non-cryptographic hashes (not collision resistant etc.)
- ➤ Used to determine similarity
- ➤ Concise similarity preserving representation of a digital artifact



Background

Fuzzy Hashing Schemes / Approximate Matching

Simplified overview similar to Ren, Liwei [1] (DFRWS EU 2015)



ssdeep: sequences of chunks (strings)
mrsh-v2: sequences of chunks (extracted by PRF)
sdhash: bag of 64-byte blocks (selected by entropy)
TLSH: bag of triplets (selected from all 5-grams)
mvhash-b: bytewise majority voting into bit-sequences

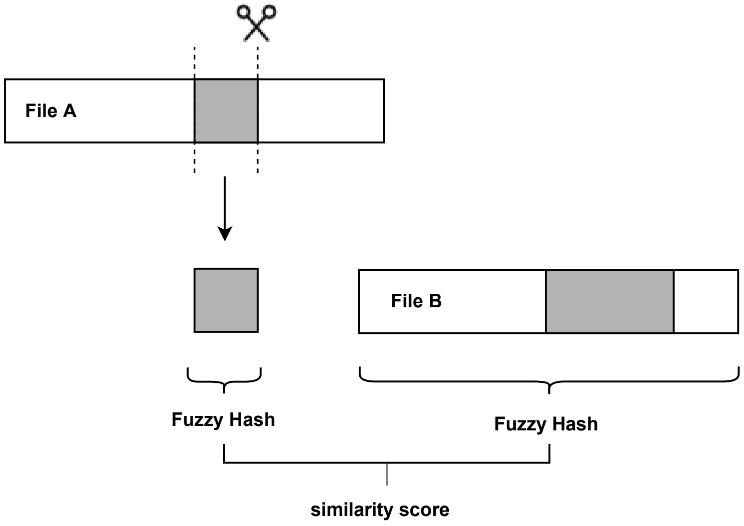
ssdeep: mapped chunks into 80 byte digests
mrsh-v2: Chunks hashed into Bloom filter
sdhash: Blocks mapped into Bloom filter
TLSH: mapped into 32-byte container
mvhash-b: bit groups mapped into Bloom filter

ssdeep: Levenshtein distance (0-100)
mrsh-v2: Hamming distance (0-100)
sdhash: Hamming distance (0-100)
TLSH: distance score (0-1000+)
mvhash-b: Hamming distance (0-100)

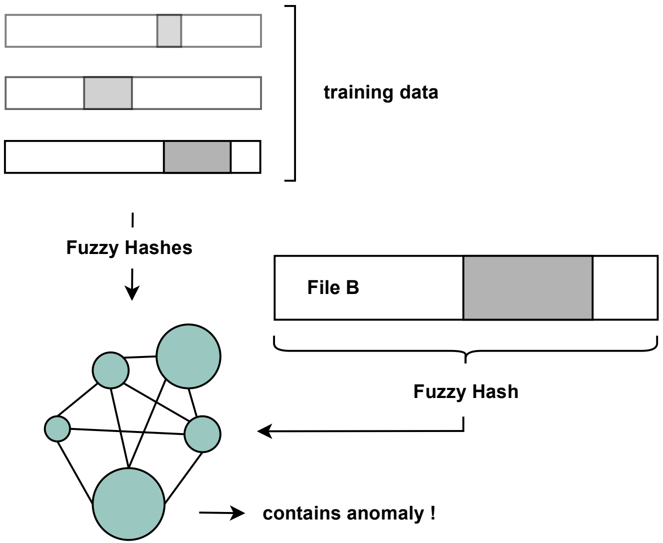
Background

Anomaly Detection

Approximate Matching



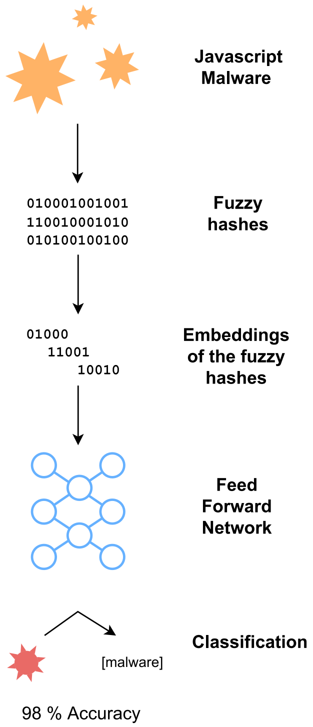
Deep Learning Approximate Matching (DLAM)



Related Work

Deep learning approximate matching (DLAM)

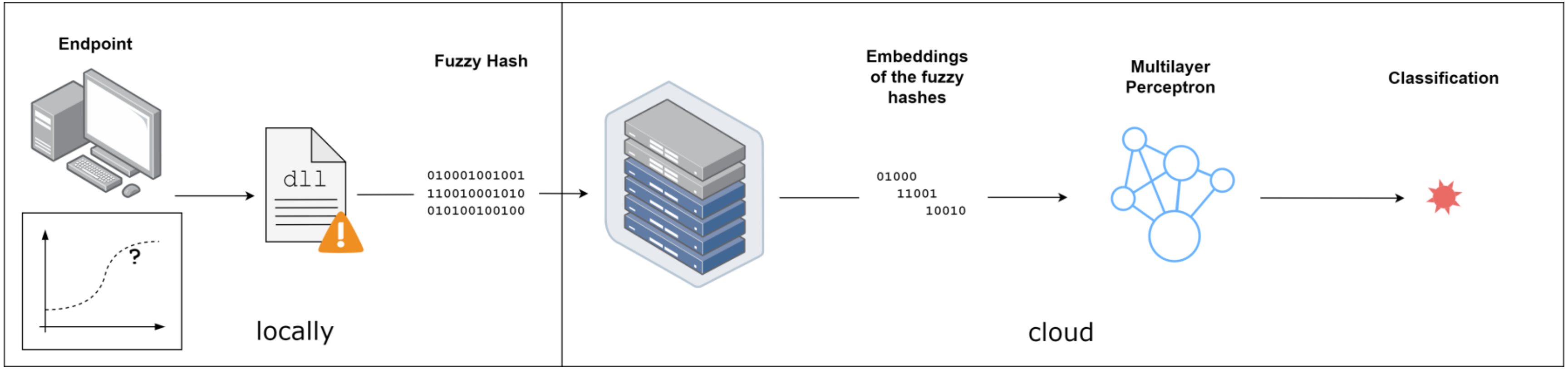
Current research impressions:
Peiser et al. [3]



Related Work

Deep learning approximate matching (DLAM)

Microsoft Defender



[2] (→ Google: combing through the fuzz)

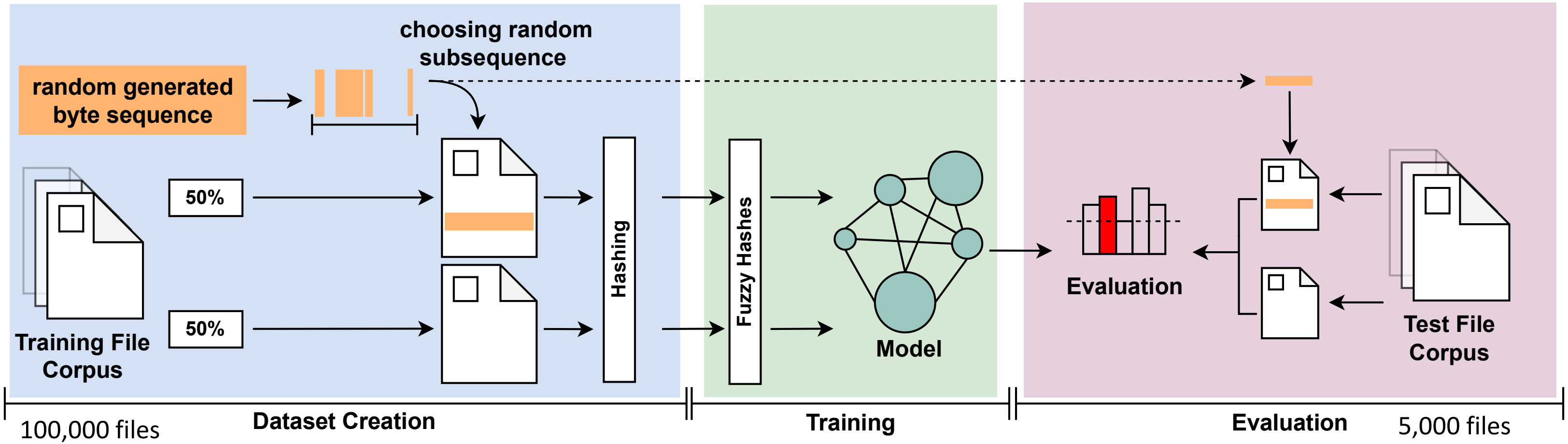
Background

Research Questions

- Is DLAM more effective than conventional approximate matching?
- Is the classification performance dependent on the file type?
- Are transformers better for DLAM?
- Is DLAM able to compensate for weaknesses in conventional approximate matching?

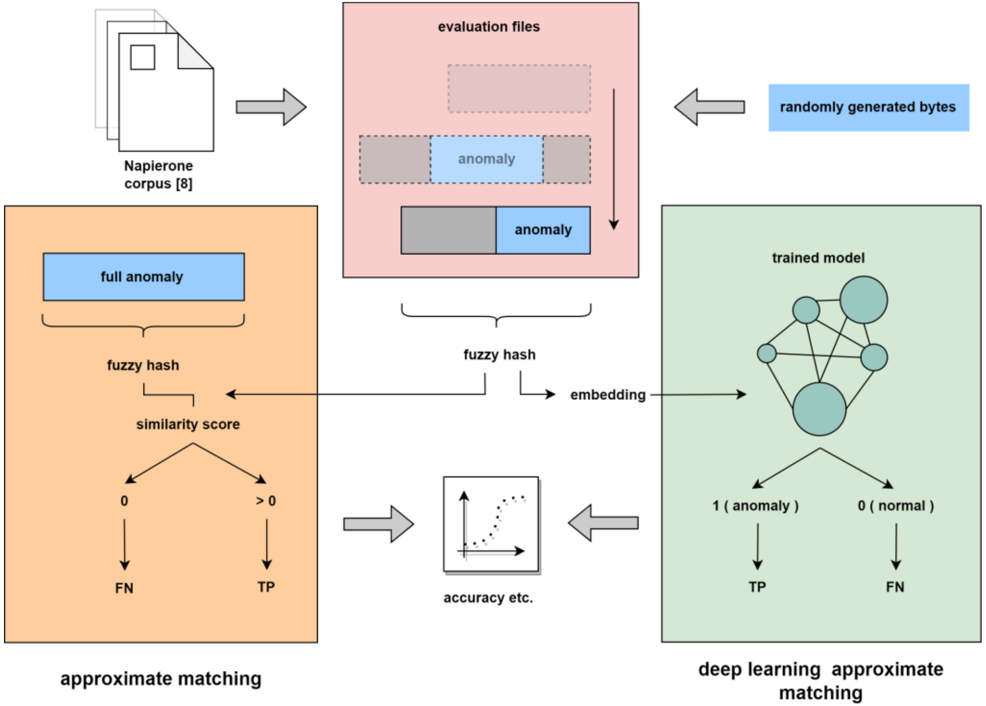
Setup

Training and Evaluation Pipeline



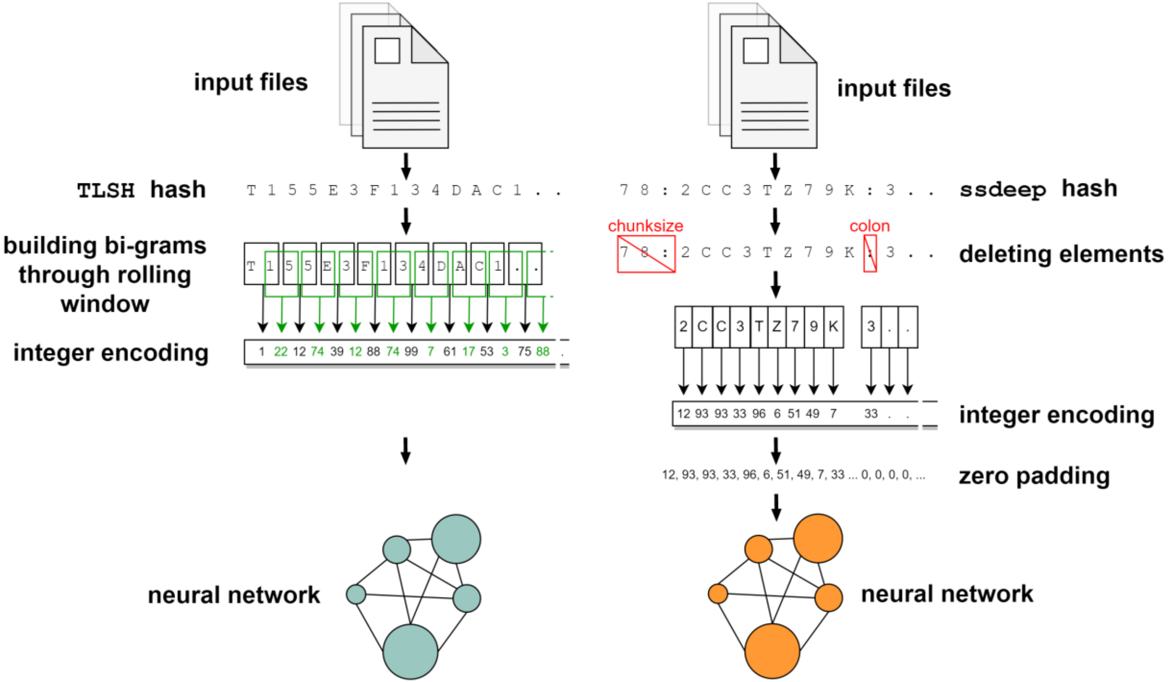
Setup

Evaluation Pipeline (simplified for FN's, TP's)



Background

Creating embeddings for deep learning



Classification

Results (Accuracies)

	conventional approximate matching				DLAM			
	Mrsh-cf	MRSH-v2	ssdeep	TLSH	ssdeep (**FF**)	TLSH (**FF**)	ssdeep (**TF**)	TLSH (**TF**)
JS	81.54	67.38	50.02	50.02	**92.70**	82.08	**92.70**	87.90
PDF	**97.3**	95.44	50.04	49.98	79.10	78.42	94.34	82.60
XLSX	96.78	88.22	52.54	51.17	93.80	90.28	**97.36**	90.74

FF: feed-forward network
TF: transformer model (small BERT)

Classification

Is DLAM more effective ? (Accuracy)

	Mrsh-cf	MRSH-v2	ssdeep	TLSH	ssdeep **(FF)**	TLSH **(FF)**	ssdeep **(TF)**	TLSH **(TF)**
JS	81.54	67.38	50.02	50.02	**92.70**	82.08	**92.70**	87.90
PDF	**97.3**	95.44	50.04	49.98	79.10	78.42	94.34	82.60
XLSX	96.78	88.22	52.54	51.17	93.80	90.28	**97.36**	90.74

FF: feed-forward network
TF: transformer model (small BERT)

Classification

Is the classification performance dependent on the file type? (Accuracy)

	Mrsh-cf	MRSH-v2	ssdeep	TLSH	ssdeep **(FF)**	TLSH **(FF)**	ssdeep **(TF)**	TLSH **(TF)**
JS	81.54	67.38	50.02	50.02	**92.70**	82.08	**92.70**	87.90
PDF	**97.3**	95.44	50.04	49.98	79.10	78.42	94.34	82.60
XLSX	96.78	88.22	52.54	51.17	93.80	90.28	**97.36**	90.74

FF: feed-forward network
TF: transformer model (small BERT)

Classification

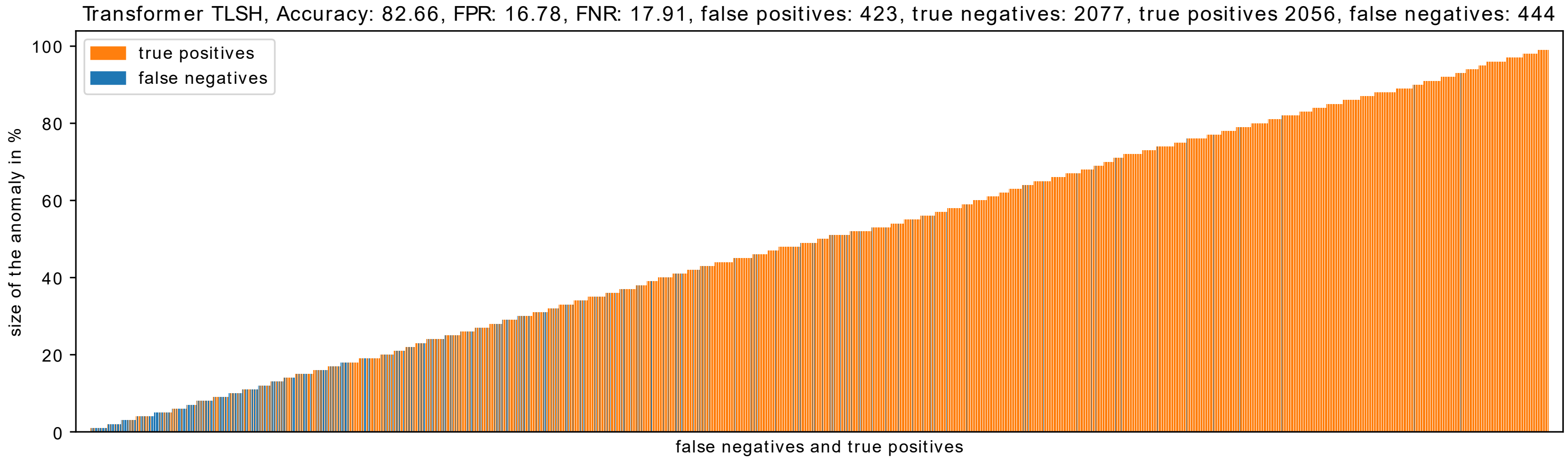
Are transformers better for DLAM ? (Accuracy)

	Mrsh-cf	MRSH-v2	ssdeep	TLSH	ssdeep **(FF)**	TLSH **(FF)**	ssdeep **(TF)**	TLSH **(TF)**
JS	81.54	67.38	50.02	50.02	**92.70**	82.08	**92.70**	87.90
PDF	**97.3**	95.44	50.04	49.98	79.10	78.42	94.34	82.60
XLSX	96.78	88.22	52.54	51.17	93.80	90.28	**97.36**	90.74

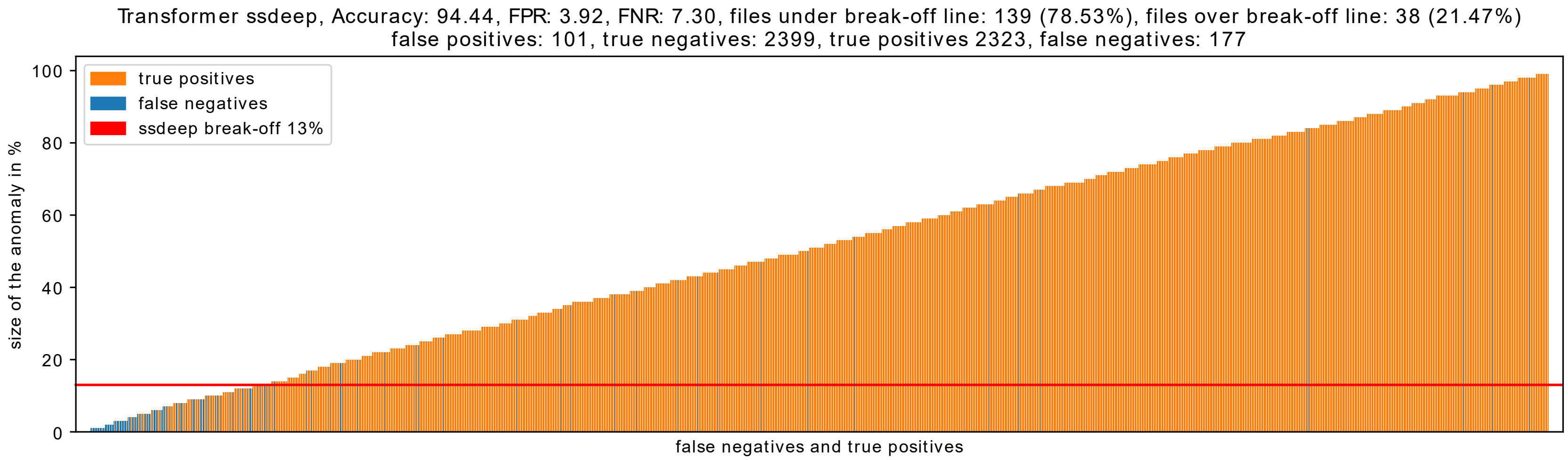
FF: feed-forward network
TF: transformer model (small BERT)

Classification

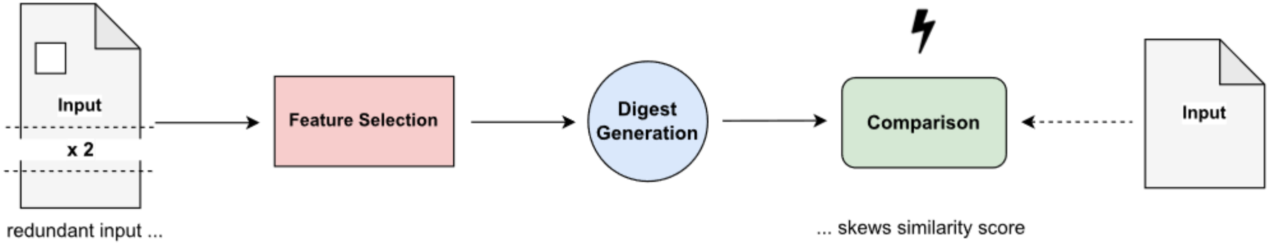
Deep learning assisted approximate matching - TLSH



Deep learning assisted approximate matching - ssdeep



Digest Comparison Impediment



Adversarial Resilience

Classification accuracy in face of repetition

Multiplication factor	mrsh-cf	MRSH-v2	ssdeep	TLSH	ssdeep (TF)	TLSH (TF)
x 1	97.54	76.13	50.0	50.0	91.09	94.94
x 2	92.19	74.37	50.0	50.0	81.68	93.57
x 4	91.99	72.67	50.0	50.0	63.56	93.25
x 8	91.69	76.47	50.0	50.0	53.55	93.07
x 16	91.49	73.73	50.0	50.0	53.55	93.95
x 32	88.21	71.11	50.0	50.0	50.0	93.59

Table 2: Prediction accuracy for anomaly detection. Per row, 5,000 files with a file size of 5,000 bytes were created. Half of them contained between 1% and 99% of anomaly bytes within them. All files were concatenated according to the respective multiplication factor and then had to be classified. The results are averaged over 10 test runs.

Conclusion

A new look on fuzzy hashing

- DLAM is a strong alternative to anomaly detection with conventional approximate matching.
  - 12 minutes for training on a conventional GPU (100,000 hashes.)
- DLAM is an enabler for anomaly detection with TLSH and ssdeep.
  - Transformers perform better than Feed Forward Networks
- ssdeep is preferable for DLAM over TLSH.
  - False Negatives are more predictable for ssdeep.
- DLAM is more robust in face of repetitive content.

Future Work

Future research

- **Are fuzzy hashes with variable size usable?**
  - MRSH-v2
- **Can DLAM be adapted to make more complex predictions?**
- **Can section-level hashing improve DLAM?**
- **How well does DLAM perform in practice?**
  - (Malware detection, data loss prevention)

Thank you.

Contact us:

frieder.uhlig@sec-defence.com

lukas.struppek@cs.tu-darmstadt.de

dominik.hintersdorf@cs.tu-darmstadt.de

thomas.goebel@unibw.de

harald.baier@unibw.de

kersting@cs.tu-darmstadt.de

is available via GitHub:
https://github.com/warlmare/DLAM

Registration sponsored by:

SEC Consult
an Eviden business

This work was supported by the Federal Ministry of Education and Research (BMBF) within the framework program “Research for Civil Security” of the German Federal Government, project KISTRA (reference no. 13N15343). It also benefited from the National Research Center for Applied Cybersecurity ATHENE, a joint effort of BMBF and the Hessian Ministry of Higher Education, Research, Science and the Arts (HMWK).

Bibliography

[1] Liwei Ren. A theoretic framework for evaluating similarity digesting tools, 03 2015. URL https://www.dfrws.org/file/127/download?token=5YOUdHpY. Visited on 2019-01-20.

[2] Lazo, E.G., 2021. Combing through the fuzz: using fuzzy hashing and deep learning to counter malware detection evasion techniques. URL: https://www.microsoft.com/en-us/security/blog/2021/07/27/combing-through-the-fuzz-using-fuzzy-hashing-and-deep-learning-to-counter-malware-detection-evasion-techniques/

[3] Peiser, S.C., Friborg, L., Scandariato, R., 2020. Javascript malware detection using locality sensitive hashing. In: ICT Systems Security and Privacy Protection, pp. 143e154.

Compression Efficiency

Algorithm	Digest file size (bytes)	Compression ratio (%)
MRSH-v2	28.67 MB	1.500%
sdhash	61.52 MB	3.218%
TLSH	394.11 KB	0.021%
FbHash	19.81 GB	1036.087%
mrsh-cf	33.55 MB	1.755%
ssdeep	485.45 KB	0,025%

Table 5: *Compression efficiency* of tested algorithms using *t5-corpus* with total size of 1911.81 MB.