



MATHEMATICS FOR
ARTIFICIAL REASONING
IN SCIENCE (MARS)
@PNNL

Security by the Analytic: A Framework for Automatic Mapping of Vulnerabilities to Attack Patterns using Artificial Intelligence

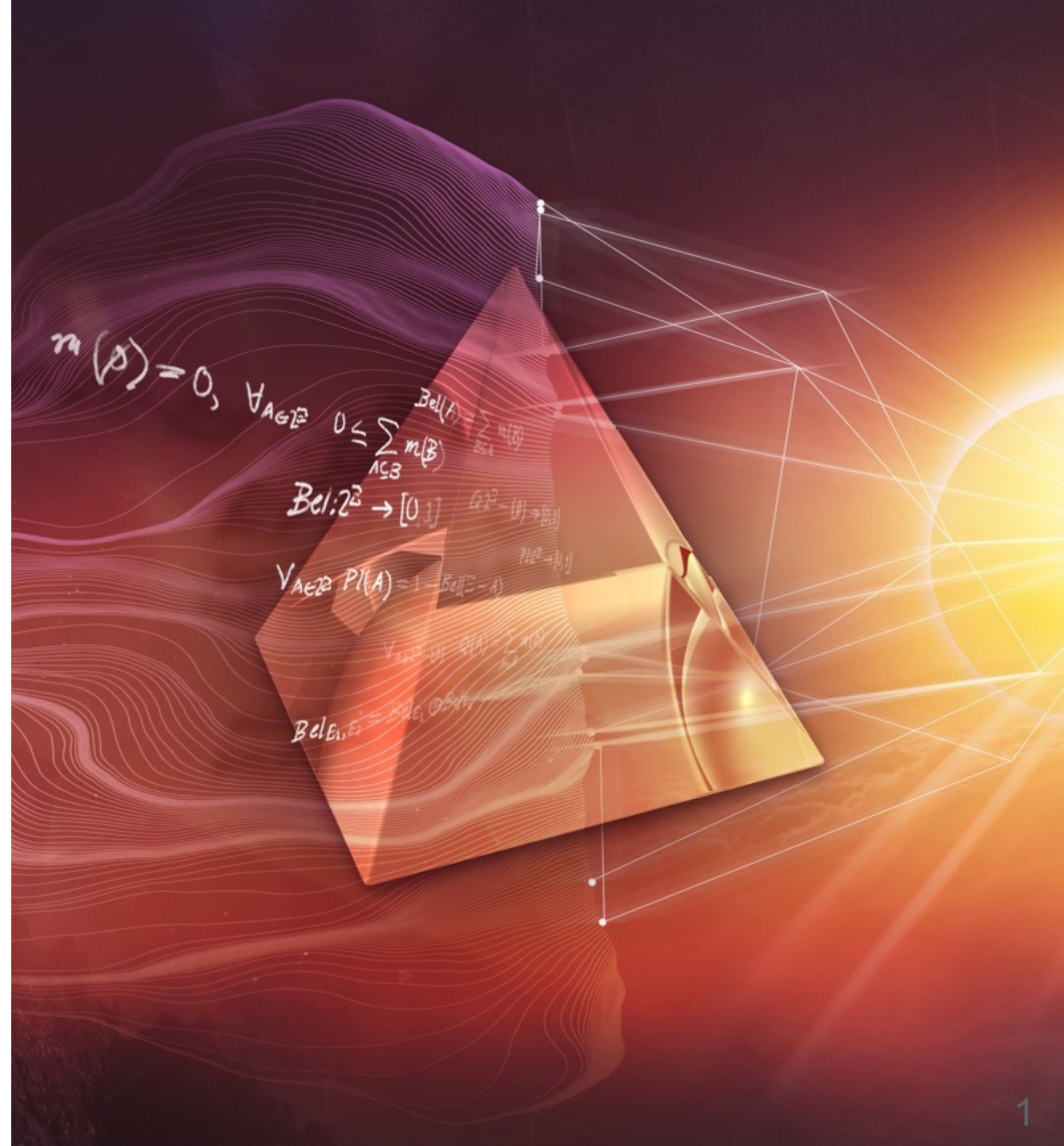
Mahantesh Halappanavar[†], Siddhartha Das^{‡†},
Edoardo Serra^{*†}, Ashutosh Dutta^{†ε},
Antonino Tumeo[†], Maxwell Levint[†],
Alex Pothen[‡], Ehab Al-Shaer^{‡†}

[Antonino Tumeo, Sumit Purohit, Kathy Panchal,
Luis de la Torre, John Miller]

[†]Pacific Northwest National Lab; [‡]Purdue University;
^{*}Boise State University; [&]Carnegie Mellon University;
^ε Amazon



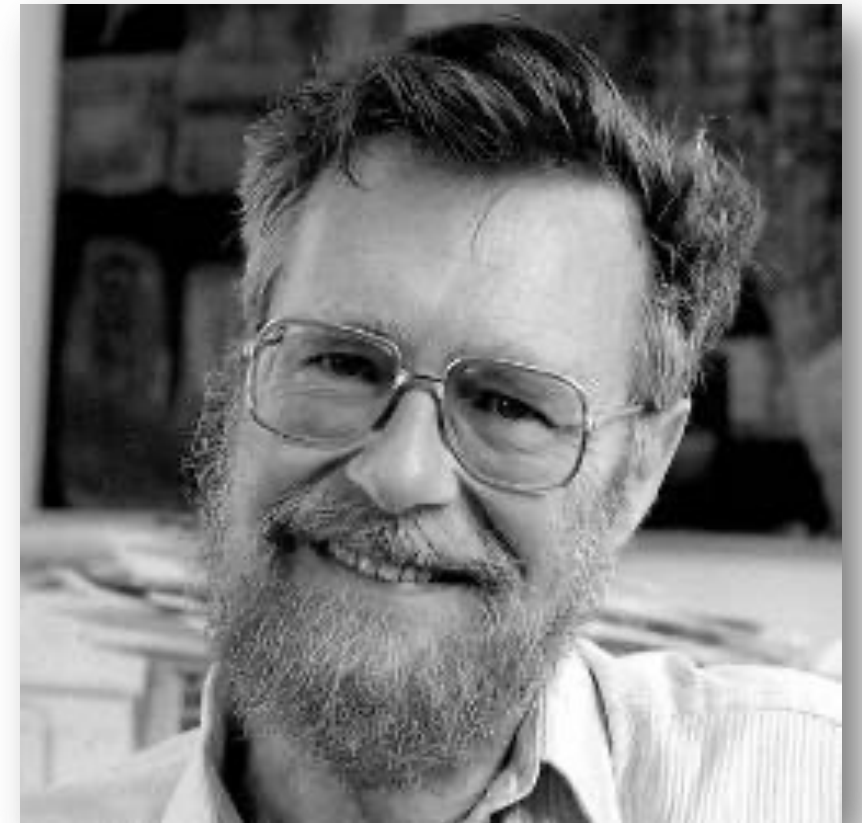
PNNL is operated by Battelle for the U.S. Department of Energy



Programming must be...

*“If **debugging** is the process of removing software bugs, then **programming** must be the process of putting them in.”*

#1 Rule in cybersecurity: *“Treat Everything Like It's Vulnerable”*



1972 ACM A.M. Turing Award
winner Edsger W. Dijkstra.

**Photo Credit: The University
of Texas at Austin**

Our Goal: Feed the Good Wolf

- Minimize the **time** between vulnerability **discovery** and **mitigation** by **linking** vulnerabilities to mitigation **actions**
- Enhance threat intelligence and exploratory research
- Eg., The “**Log4Shell**” flaw was widely exploited -- ~**40,000** attempted attacks within **two hours** of it becoming public, and >**830,000** attempts within the first **three days**.*

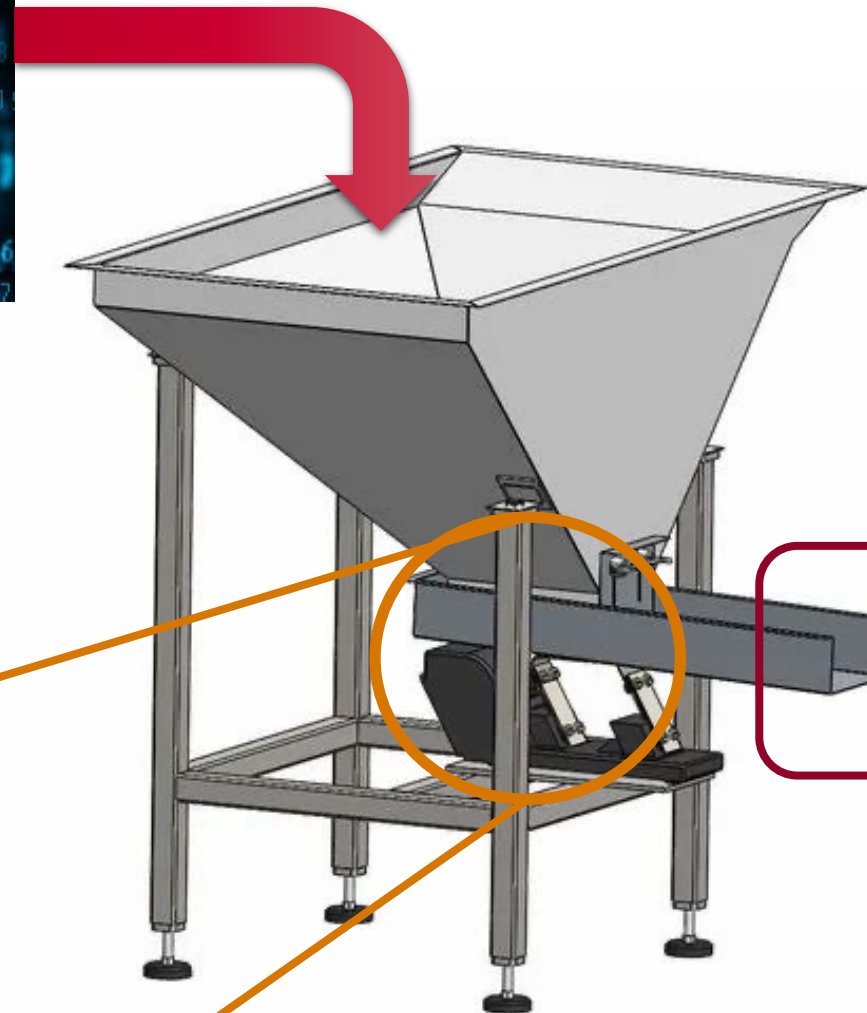


* <https://www.checkpoint.com/cyber-hub/cyber-security/what-is-cybersecurity/biggest-cybersecurity-challenges-in-2022/>

Overview

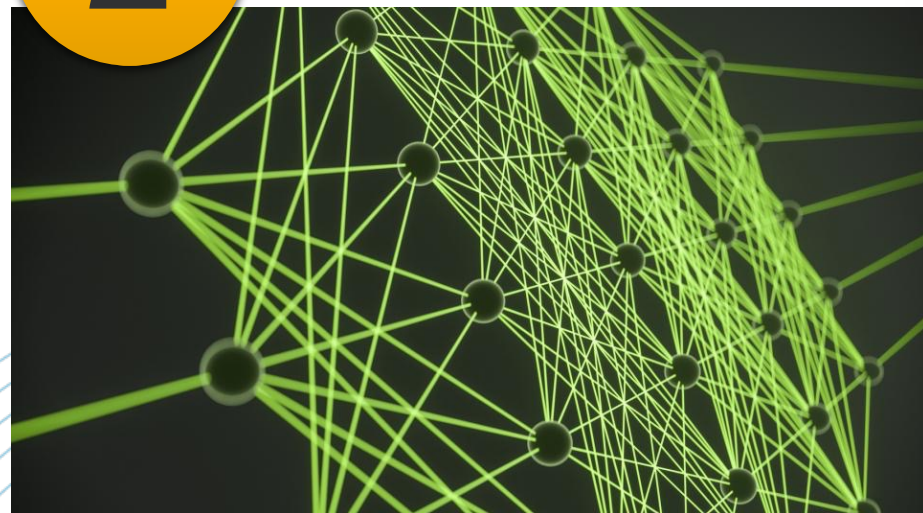
Security by the Analytic

1



```
int num = 50;  
int[] x = new int[num];  
int[] y = new int[num];  
  
void setup() {  
  size(100, 100);  
  noStroke();  
  fill(255, 102);  
}  
  
void draw() {  
  background(0);  
  // Shift the values to the right  
  for (int i = num-1; i > 0; i--) {  
    x[i] = x[i-1];  
    y[i] = y[i-1];  
  }
```

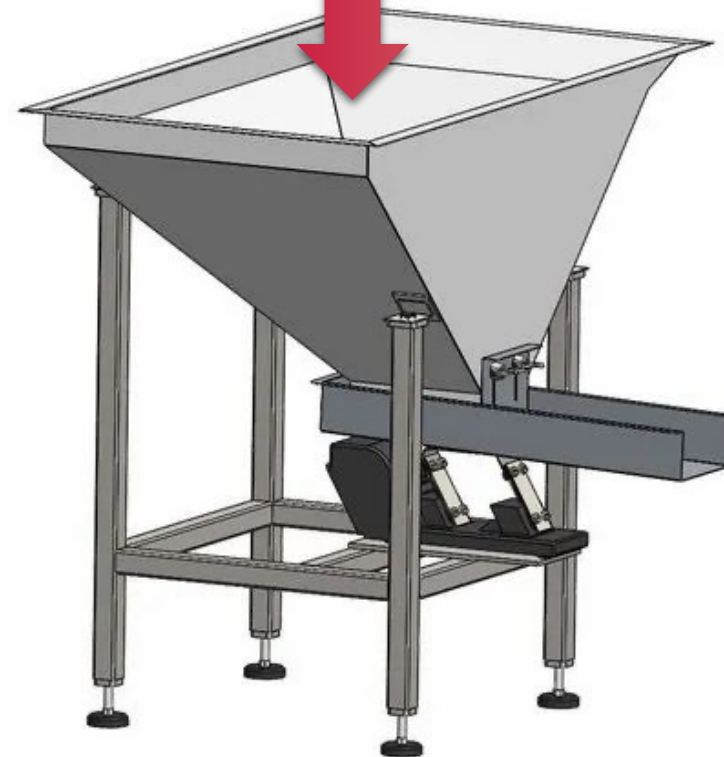
2



3

1. Data & Problem Setting

1



```
int num = 50;  
int[] x = new int[num];  
int[] y = new int[num];
```

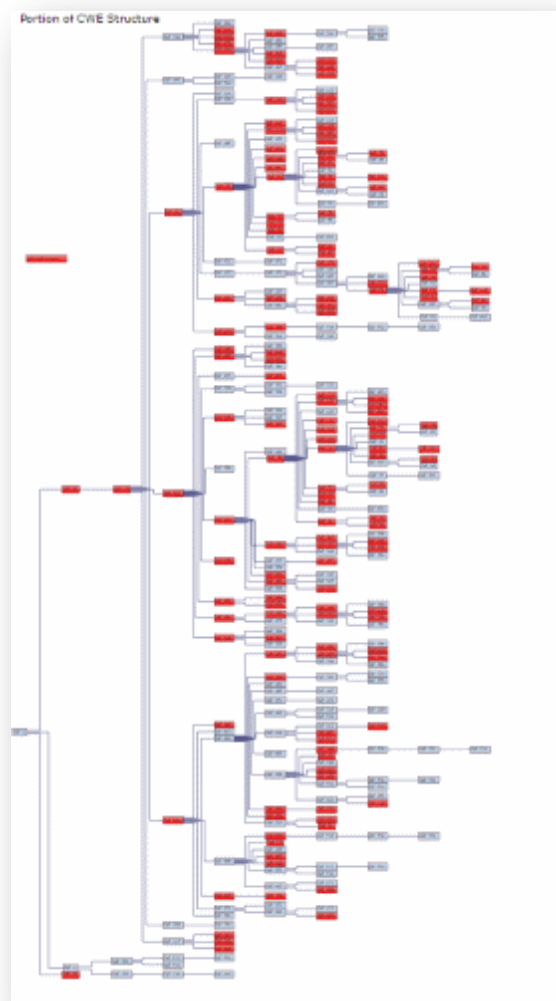
```
void setup() {  
  size(100, 100);  
  noStroke();  
  fill(255, 102);  
}
```

```
void draw() {  
  background(0);  
  // Shift the values to the right  
  for (int i = num-1; i > 0; i--) {  
    x[i] = x[i-1];  
    y[i] = y[i-1];  
  }
```

```
  // Add the new values to the beginning of the array  
  mouseX = mouseX;  
  mouseY = mouseY;  
  // Draw the circles  
  for (int i = 0; i < num; i++) {  
    ellipse(x[i], y[i], v/2.0, v/2.0);  
  }
```

Common Weakness Enumeration (CWE)

A **blueprint** for understanding software flaws and their impact through a **hierarchically** designed **dictionary** of software **weaknesses** (934 Weaknesses)



- [-] [P] Improper Check or Handling of Exceptional Conditions - (703)
- [-] [P] Improper Neutralization - (707)
 - [-] [G] Improper Encoding or Escaping of Output - (116)
 - [-] [G] Improper Neutralization of Special Elements - (138)
 - [-] [B] Improper Null Termination - (170)
 - [-] [G] Encoding Error - (172)
 - [-] [G] Improper Input Validation - (20)
 - [-] [G] Improper Handling of Syntactically Invalid Structure - (228)
 - [-] [B] Improper Handling of Inconsistent Structural Elements - (240)
 - [-] [B] Deletion of Data Structure Sentinel - (463)
 - [-] [G] Improper Neutralization of Special Elements in Output Used by a Downstream Component ('Injection') - (74)
 - [-] [B] Improper Neutralization of Formula Elements in a CSV File - (1236)
 - [-] [G] Failure to Sanitize Special Elements into a Different Plane (Special Element Injection) - (75)
 - [-] [G] Improper Neutralization of Special Elements used in a Command ('Command Injection') - (77)
 - [-] [B] Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') - (79)
 - [-] [B] XML Injection (aka Blind XPath Injection) - (91)
 - [-] [B] Improper Neutralization of CRLF Sequences ('CRLF Injection') - (93)
 - [-] [B] Improper Control of Generation of Code ('Code Injection') - (94)
 - [-] [G] Improper Neutralization of Special Elements in Data Query Logic - (943)
 - [-] [B] Improper Neutralization of Data within XPath Expressions ('XPath Injection') - (643)
 - [-] [B] Improper Neutralization of Data within XQuery Expressions ('XQuery Injection') - (652)
 - [-] [B] Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') - (89)
 - [-] [V] SQL Injection: Hibernate - (564)



Common Vulnerabilities & Exposures (CVE)

- **Bugs:** Mistakes happen in the process of building and coding a system
- **Vulnerabilities:** Bugs that can be **exploited** to induce **unintended** behavior from software/protocol/hardware
- A bug is determined to be a vulnerability is registered by **MITRE** as a **CVE**
 - **Publicly known** information-security vulnerabilities and exposures
 - TOTAL CVE Records: **215,715**. (<https://cve.mitre.org/>) on 10/31/2023
- Some Examples:
 - Broken Authentication
 - SQL Injection
 - Cross-Site Scripting



[330 CVE Numbering Authorities](#)

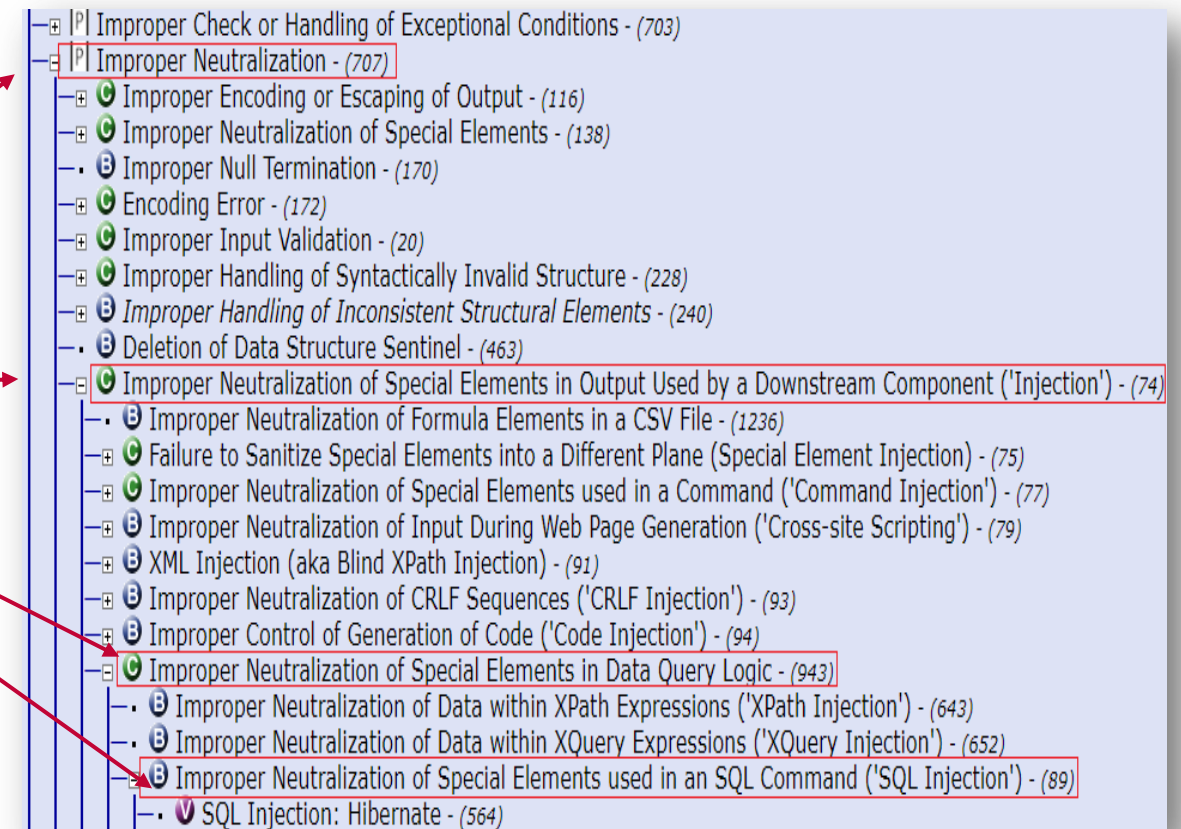
Example description: “*CVE-2004-0366: SQL injection vulnerability in the libpam-pgsql library before 0.5.2 allows attackers to execute arbitrary SQL statements.*”

Mapping CVEs to CWEs

- CVE reports are uniquely identified computer security vulnerabilities, where a **vulnerability** is defined as a **set of one or more weaknesses** in a specific **product** or protocol that allows an attacker to exploit the behaviors or resources to compromise a system
- Example: “*CVE-2004-0366: SQL injection vulnerability in the libpam-pgsql library before 0.5.2 allows attackers to execute arbitrary SQL statements.*”

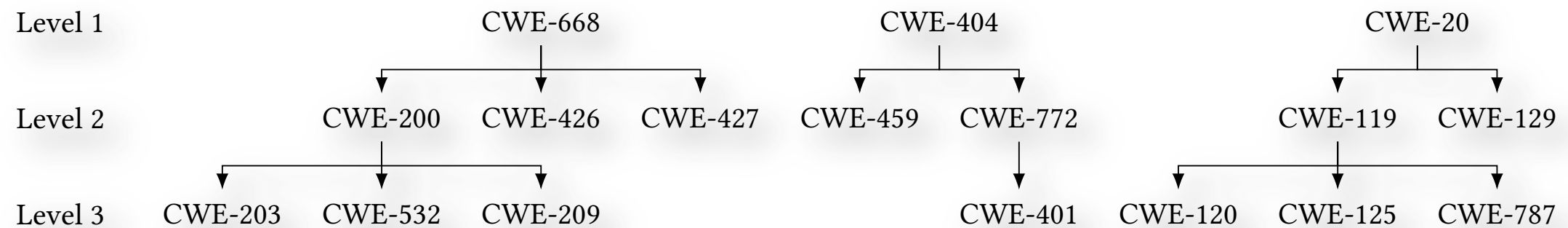
“*CVE-2004-0366: SQL injection vulnerability in the libpam-pgsql library before 0.5.2 allows attackers to execute arbitrary SQL statements.*”

<https://cve.mitre.org>



CVE to CWE Mapping

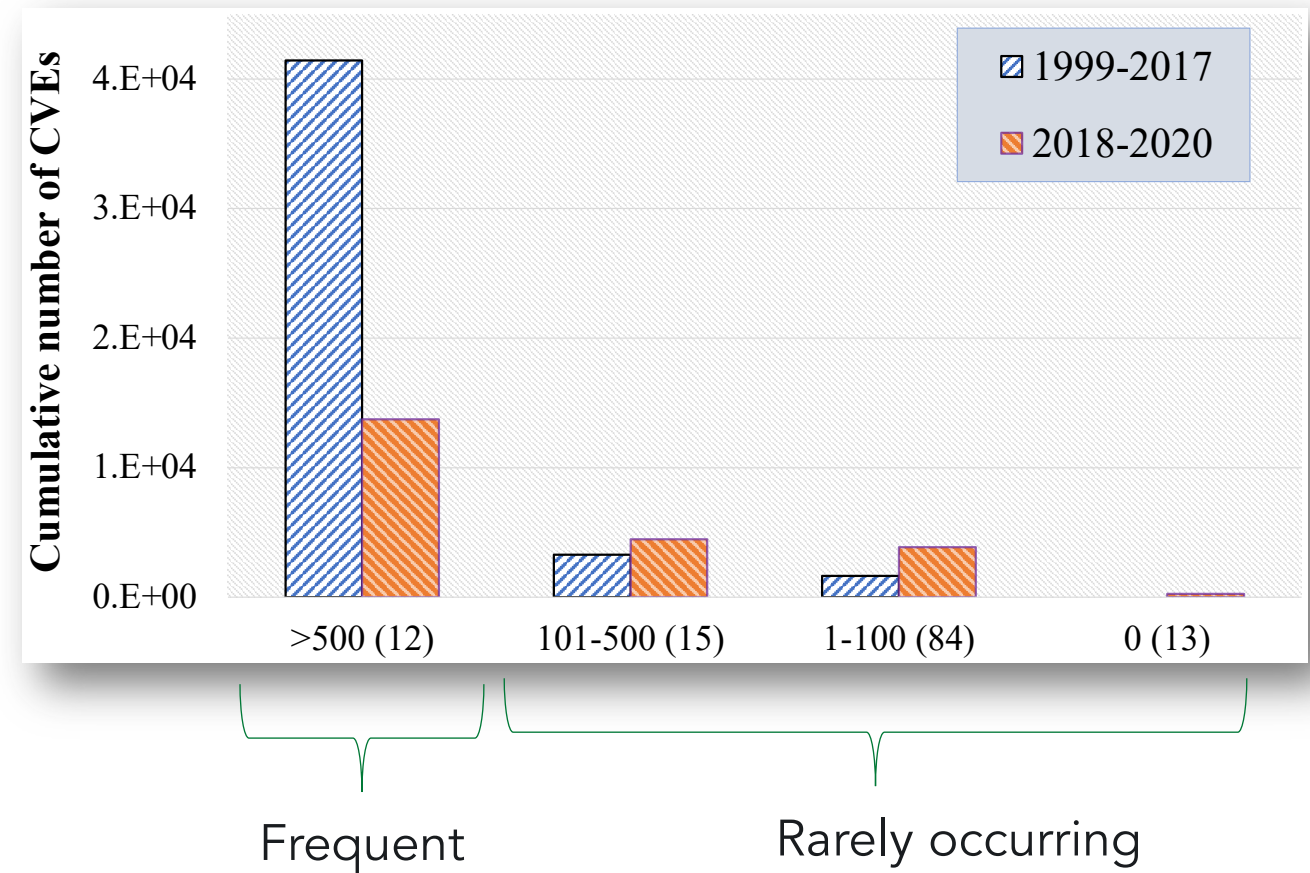
- We considered **933** CWE classes from MITRE (circa 2021)
- About **124** types of CWEs are classified in National Vulnerability Dataset (NVD)
- CVE to CWE mapping is done manually, requires **human expertise**, and error prone



Partial hierarchy of CWE extracted from MITRE to demonstrate how precise and relaxed predictions are performed. We consider **124** CWEs that are distributed in three levels in the hierarchy, with **34** in the first level, **78** in the second level, and **16** in the third level.

CVE to CWE Mapping: Challenges

- **Distribution** of the number of CVEs per CWE in NVD, bucketed into four categories
- Data partitioned into two time periods to simulate testing for CVEs observed in the future
- 1999-2017 (used for training) and 2018-2020 (used for testing).



Common Attack Pattern Enumeration and Classification (CAPEC)



Pacific Northwest
NATIONAL LABORATORY

- An “**exploit**” makes a “weakness” a “vulnerability”
- “**Attack Patterns**” are descriptions of the common attributes and approaches employed by adversaries to **exploit** known weaknesses in cyber-enabled capabilities
- CAPEC is a **dictionary** of common identifiers for attack patterns

1000 - Mechanisms of Attack

- [-] **C** Engage in Deceptive Interactions - (156)
 - [+] **M** Content Spoofing - (148)
 - [+] **M** Identity Spoofing - (151)
 - [+] **S** Fake the Source of Data - (194)
 - [+] **S** Principal Spoof - (195)
 - [+] **S** Signature Spoof - (473)
 - **S** Pharming - (89)
 - [+] **S** Phishing - (98)
 - [+] **M** Resource Location Spoofing - (154)
 - [+] **M** Action Spoofing - (173)
 - [+] **M** Manipulate Human Behavior - (416)
- [-] **C** Abuse Existing Functionality - (210)
 - [+] **M** Interface Manipulation - (113)
 - [+] **M** Flooding - (125)
 - [+] **M** Excessive Allocation - (130)
 - **M** Resource Leak Exposure - (131)
 - [+] **M** Functionality Misuse - (212)
 - [+] **M** Communication Channel Manipulation - (216)
 - [+] **M** Sustained Client Engagement - (227)
 - [+] **M** Protocol Manipulation - (272)
 - [+] **M** Functionality Bypass - (554)
- [-] **C** Manipulate Data Structures - (255)
- [-] **C** Manipulate System Resources - (262)
- [-] **C** Inject Unexpected Items - (152)
- [-] **C** Employ Probabilistic Techniques - (223)
- [-] **C** Manipulate Timing and State - (172)
- [-] **C** Collect and Analyze Information - (118)

Some Well-Known Attack Patterns:

- HTTP Response Splitting ([CAPEC-34](#))
- Session Fixation ([CAPEC-61](#))
- Cross Site Request Forgery ([CAPEC-62](#))
- SQL Injection ([CAPEC-66](#))
- Cross-Site Scripting ([CAPEC-63](#))
- Buffer Overflow ([CAPEC-100](#))
- Clickjacking ([CAPEC-103](#))
- Relative Path Traversal ([CAPEC-139](#))
- XML Attribute Blowup ([CAPEC-229](#))



Mapping CVE to CWE to CAPEC

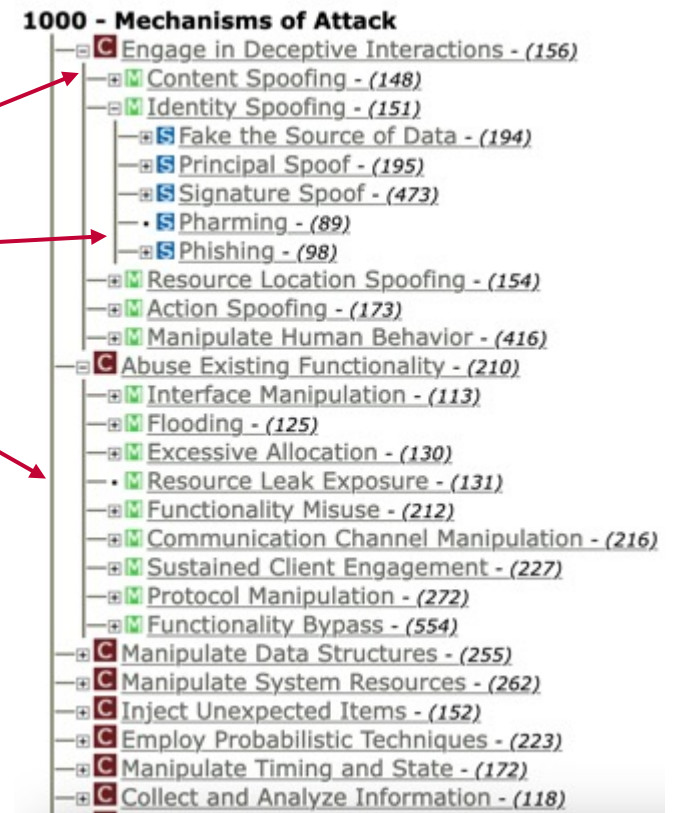
- We want to know what attack sequences can be taken taken given CVE descriptions
- CVE-CWE mapping ---- CWE-CAPEC mapping

“CVE-2004-0366: SQL injection vulnerability in the libpam-pgsql library before 0.5.2 allows attackers to execute arbitrary SQL statements.”



Improper Neutralization of Special Elements in Output Used by a Downstream Component ('Injection') - (74)

Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') - (89)



1000 - Mechanisms of Attack

Engage in Deceptive Interactions - (156)

Abuse Existing Functionality - (210)

<https://cve.mitre.org>

Adversarial Tactics, Techniques & Common Knowledge (ATT&CK)

ATT&CK™

Adversarial Tactics, Techniques & Common Knowledge



Pacific Northwest NATIONAL LABORATORY



Tactics

MITRE ATT&CK

Matrices Tactics Techniques Mitigations Groups Software Resources

Initial Access 9 techniques	Execution 10 techniques	Persistence 18 techniques	Privilege Escalation 12 techniques	Defense Evasion 34 techniques	Credential Access 14 techniques	Discovery 24 techniques	Lateral Movement 9 techniques	Collection 16 techniques	Command and Control 16 techniques
Drive-by Compromise	Command and Scripting Interpreter (7)	Account Manipulation (4)	Abuse Elevation Control Mechanism (4)	Abuse Elevation Control Mechanism (4)	Brute Force (4)	Account Discovery (4)	Exploitation of Remote Services	Archive Collected Data (3)	Application Layer Protocol (4)
Exploit Public-Facing Application	Exploitation for Client Execution	BITS Jobs	Access Token Manipulation (5)	Access Token Manipulation (5)	Credentials from Password Stores (3)	Application Window Discovery	Internal Spearphishing	Audio Capture	Communication Through Removable Media
External Remote Services	Inter-Process Communication (2)	Boot or Logon Autostart Execution (11)	Boot or Logon Autostart Execution (11)	BITS Jobs	Exploitation for Credential Access	Browser Bookmark Discovery	Lateral Tool Transfer	Automated Collection	Data Encoding (2)
Hardware Additions	Native API	Boot or Logon Initialization Scripts (5)	Boot or Logon Initialization Scripts (5)	Deobfuscate/Decode Files or Information	Forced Authentication	Cloud Service Dashboard	Remote Service Session Hijacking (2)	Clipboard Data	Data Obfuscation (3)
Phishing (3)	Scheduled Task/Job (5)	Browser Extensions	Boot or Logon Initialization Scripts (5)	Direct Volume Access	Input Capture (4)	Cloud Service Discovery	Replication Through Removable Media	Data from Cloud Storage Object	Dynamic Resolution (3)
Replication Through Removable Media	Shared Modules	Compromise Client Software Binary	Create or Modify System Process (4)	Execution Guardrails (1)	Man-in-the-Middle (1)	Domain Trust Discovery	Software Deployment Tools	Data from Information Repositories (2)	Encrypted Channel (2)
Supply Chain Compromise (3)	Software Deployment Tools	Create Account (3)	Event Triggered Execution (15)	Exploitation for Defense Evasion	Modify Authentication Process (3)	File and Directory Discovery	Taint Shared Content	Data from Local System	Fallback Channels
Trusted Relationship	System Services (2)	Create or Modify System Process (4)	Exploitation for Privilege Escalation	File and Directory Permissions Modification (2)	Network Sniffing	Network Service Scanning	Use Alternate Authentication Material (4)	Data from Network Shared Drive	Ingress Tool Transfer
Valid Accounts (4)	User Execution (2)	Event Triggered Execution (15)	Group Policy Modification	Group Policy Modification	OS Credential Dumping (8)	Network Share Discovery		Data from Removable Media	Multi-Stage Channels
	Windows Management Instrumentation	External Remote Services	Hijack Execution Flow (11)	Hide Artifacts (5)	Steal Application Access Token	Network Sniffing		Data from Removable Media	Non-Application Layer Protocol
		Hijack Execution Flow (11)	Impair Defenses (6)	Hijack Execution Flow (11)	Steal or Forge Kerberos Tickets (3)	Password Policy Discovery		Data Staged (2)	Non-Standard Port
		Process Injection (11)	Indicator Removal on Host (6)	Impair Defenses (6)	Steal Web Session Cookie	Peripheral Device Discovery		Email Collection (3)	Protocol Tunneling
		Scheduled Task/Job (5)	Indirect Command Execution	Indicator Removal on Host (6)	Two-Factor Authentication Interception	Permission Groups Discovery (3)		Input Capture (4)	Proxy (4)
		Valid Accounts (4)	Masquerading (6)	Indirect Command Execution	Unsecured Credentials (2)	Process Discovery		Man in the Browser	Remote Access Software
			Modify Authentication Process (3)	Masquerading (6)		Query Registry		Man-in-the-Middle (1)	Traffic Signaling (1)
			Modify Cloud Compute	Modify Authentication Process (3)		Remote System Discovery		Screen Capture	Web Service (3)
				Modify Authentication Process (3)		Software Discovery (1)		Video Capture	
				Modify Cloud Compute		System Information Discovery			

Techniques

- Describes **operational phases** in adversary's lifecycle (e.g., Persistence, Lateral Movement, Exfiltration)
- Details specific tactics, techniques, and procedures (**TTPs**) used in advanced persistent threats (APT)
- Attack patterns enumerated by CAPECs are used in specific ATT&CK techniques

Problem Setting: Summary



CVSS



CWE

- Abstract
- ~1k

CAPEC

- Abstract
- ~600

ATT&CK

- Specific
- 196 tech.; 411 sub

CVE

- Specific
- >200k



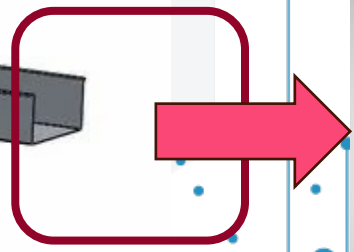
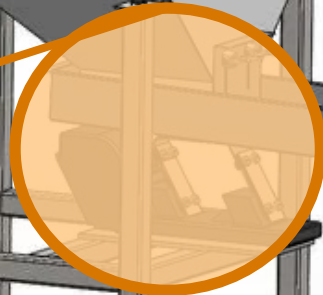
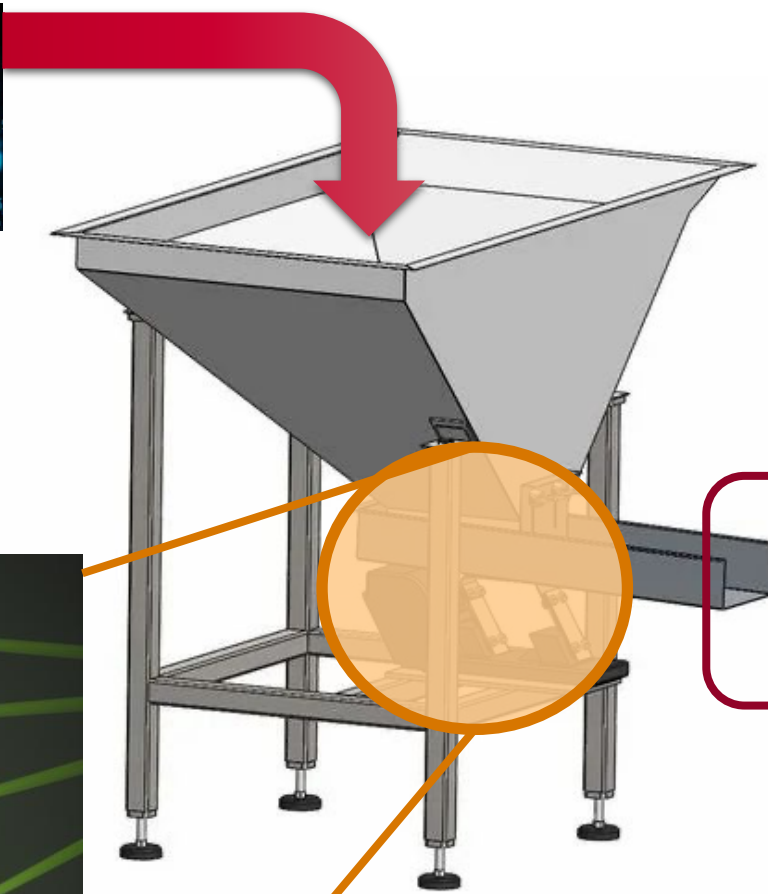
Specific

Abstract/Generalization

Specific

2. Approach: V2W-BERT & VWC-MAP

1

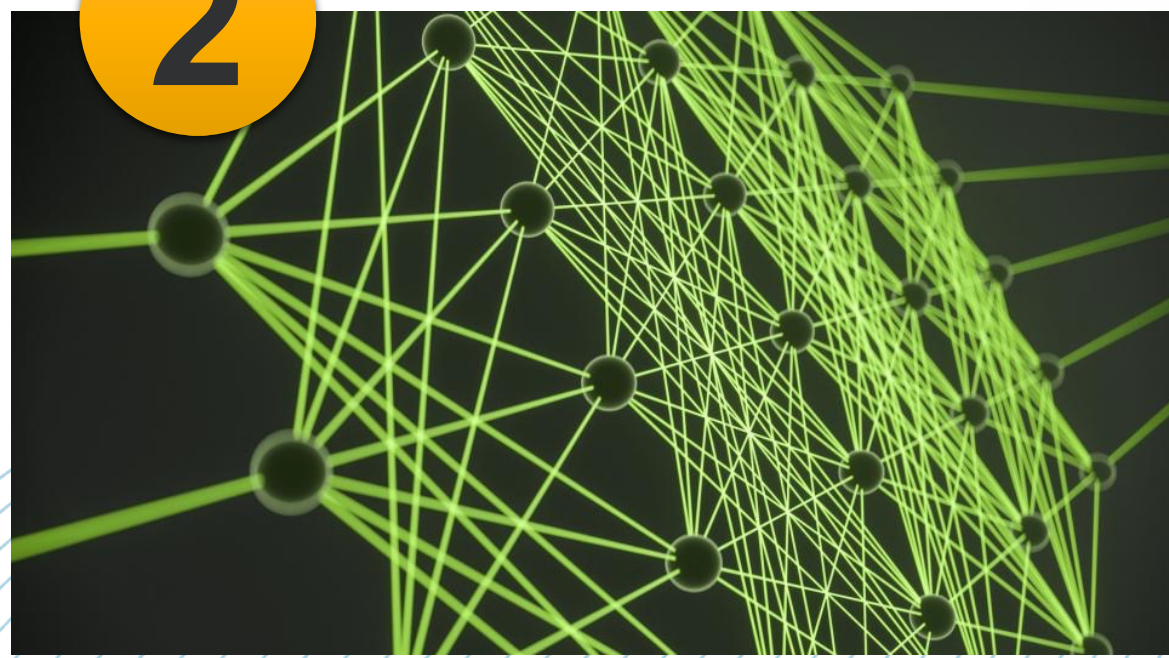


```
int num = 50;  
int[] x = new int[num];  
int[] y = new int[num];  
  
void setup() {  
  size(100, 100);  
  noStroke();  
  fill(255, 102);  
}  
  
void draw() {  
  background(0);  
  // Shift the values to the right  
  for (int i = num-1; i > 0; i--) {  
    x[i] = x[i-1];  
    y[i] = y[i-1];  
  }  
  // Add the new values to the beginning of the array  
  x[0] = mouseX;  
  y[0] = mouseY;  
}
```



3

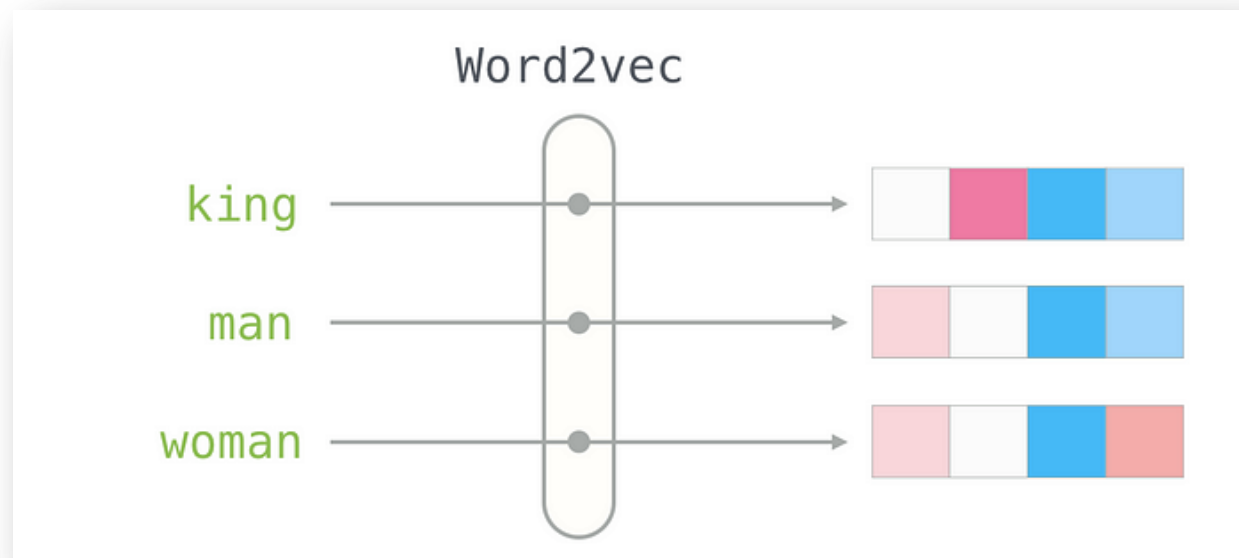
2



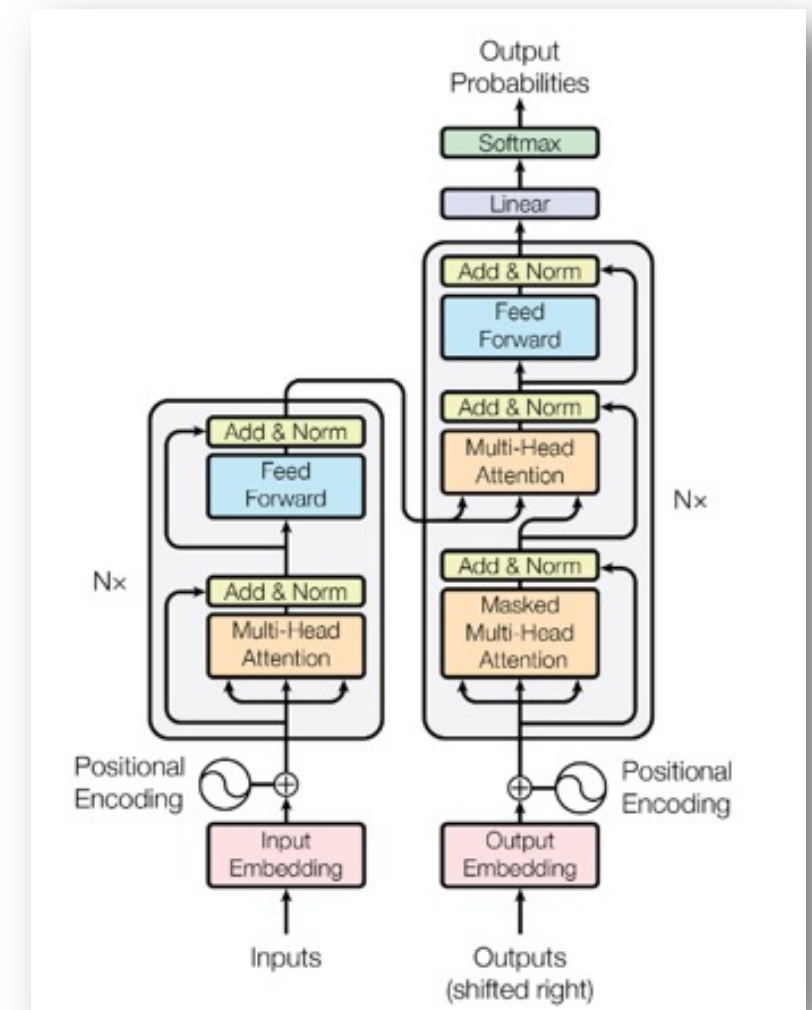
Key Ideas

“You shall know a *word* by the *company* it keeps”
 – Firth (1957)

- Meaning \approx Location in **semantic** space
 - Latent Semantic Analysis (LSA) – *state-of-the-art*
 - ✓ Vector Based on SVD of Word by Document Matrix



Source: <http://jalammar.github.io/illustrated-word2vec/>

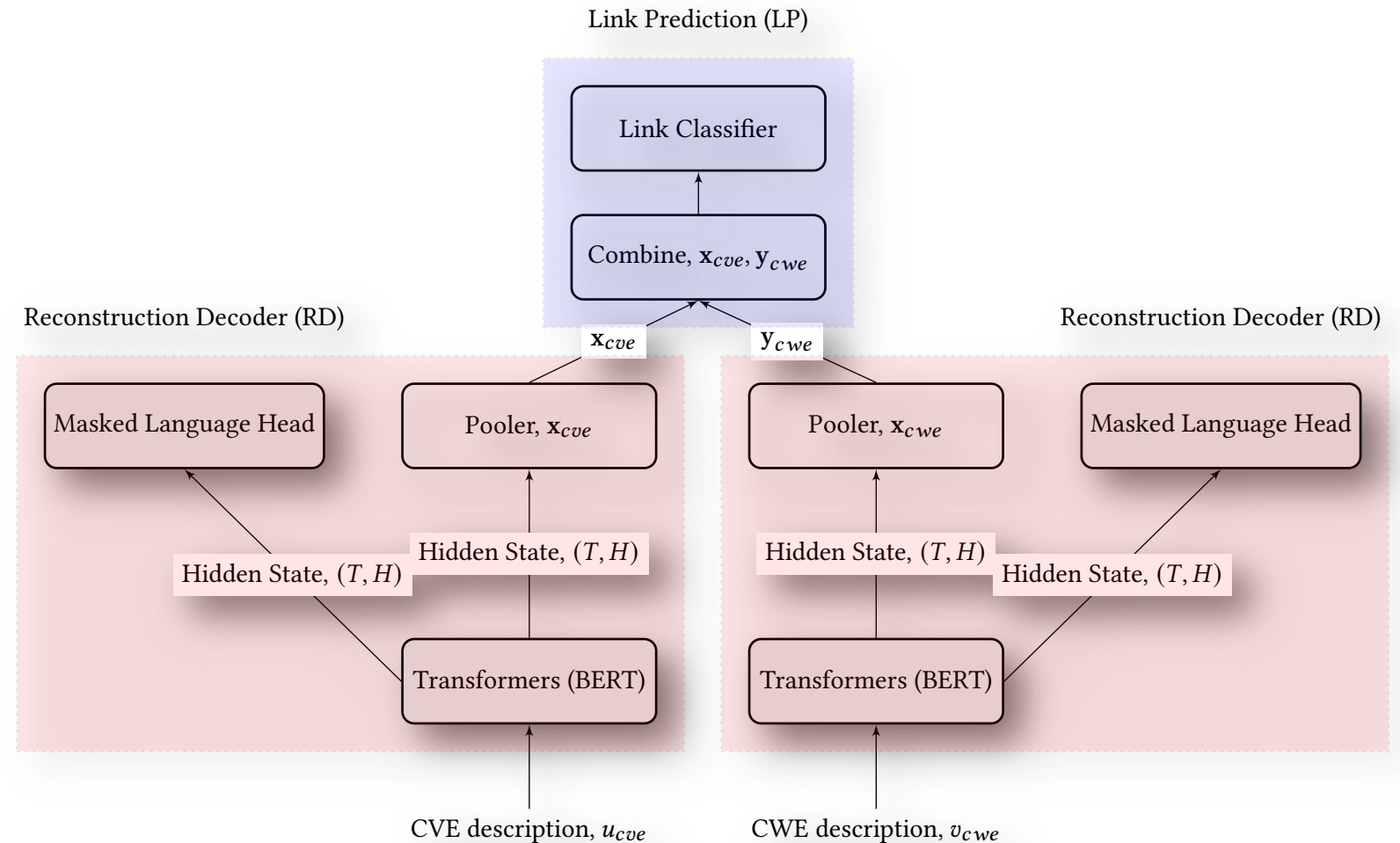


From “Attention is all you need” paper by Vaswani, et al., 2017

CVE-CWE: V2W-BERT Framework

- The Transformer component including the **Reconstruction-Decoder (RD)** is highlighted in pink, and the **Link Prediction (LP)** component is highlighted in blue
- Pre-training is done only on the **Masked Language Model**, and during link prediction the entire framework is considered
- Learnable function:

$$l = F_{\theta}(v, w)$$



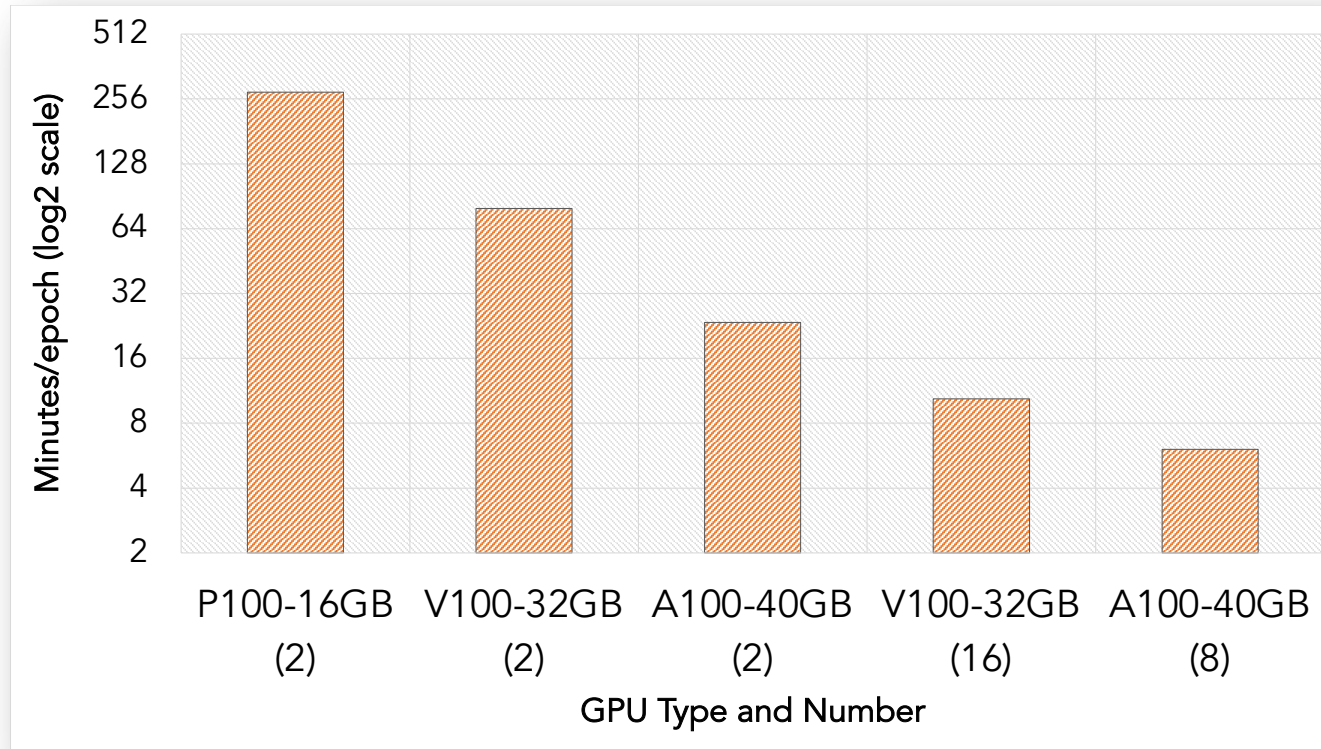
Architectural overview of V2W-BERT framework

S Das, E. Serra, M. Halappanavar, A. Pothan, and E. Al-Shaer. "V2W-BERT: A Framework for Effective Hierarchical Multiclass Classification of Software Vulnerabilities." In proceedings of the *8th IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. Porto, Portugal. October 2021. **[Best Application Paper Award]**

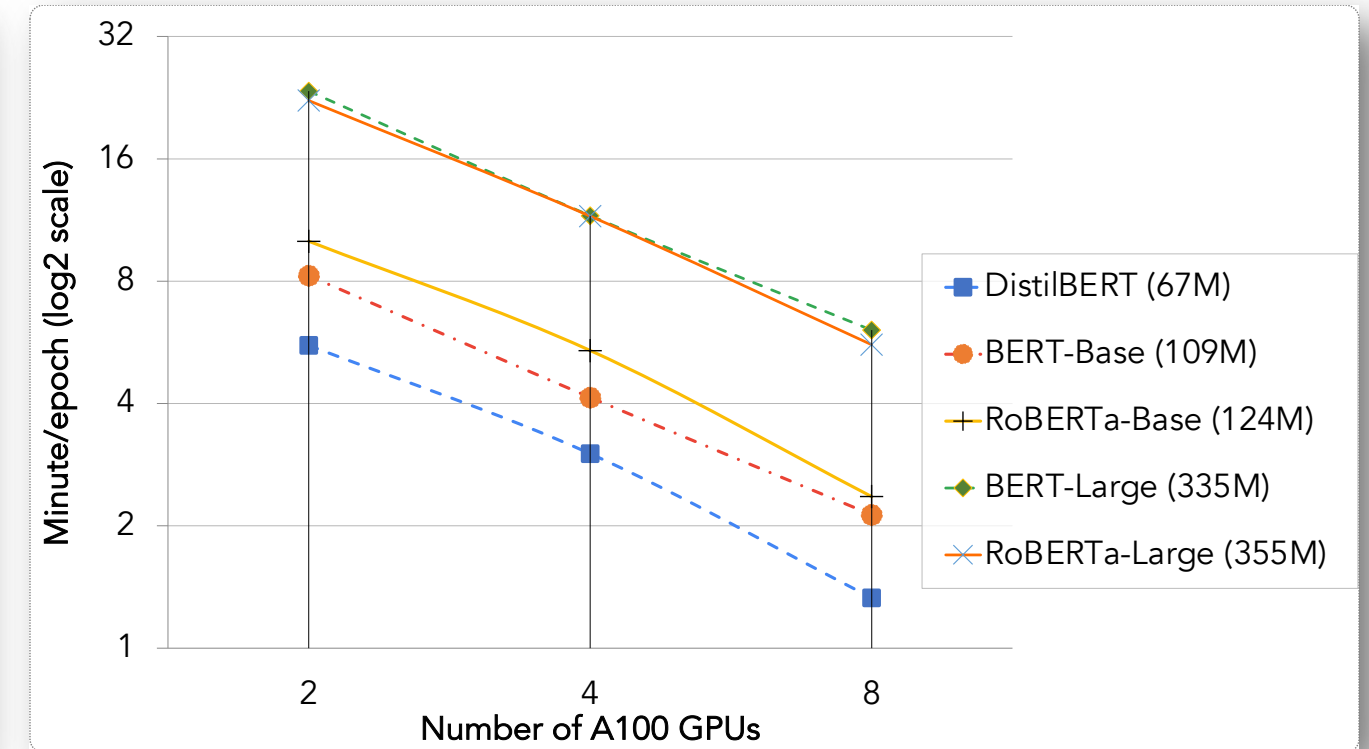
Implementation details

- Huggingface and PyTorch Lightning :
 - Huggingface provides a wide range of transformer-based models
 - PyTorch Lightning helps organize and parallelize PyTorch implementations efficiently (DP, DDP)
- Step 1: V2W-BERT pretraining with CVE/CWE descriptions
 - Retraining BERT model with domain specific CVE descriptions
 - Take a batch of CVEs and update
- Step 2: V2W-BERT linking CVEs with CWEs
 - Take batch of CVEs, batch of CWEs, process them parallel
 - Create training links process them parallel
 - Process reconstruction loss of the batch of CVEs and CWEs parallel

Strong Scaling of V2W-BERT: GPUs vs. #parameters



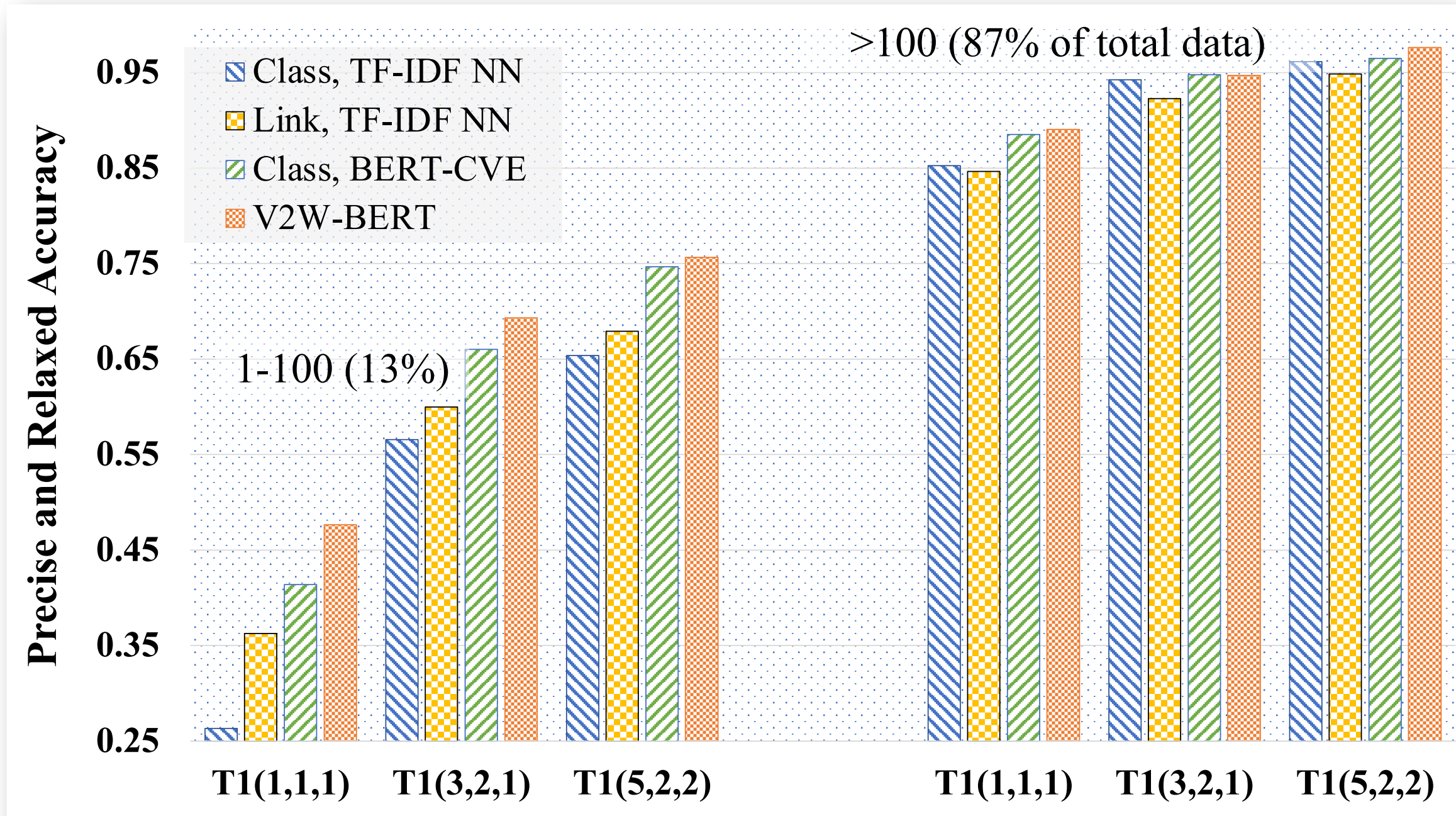
Pre-training time for BERT-Large across different generations of Nvidia accelerators with different number of GPUs



Scaling of different language models on A100 system with a batch size of 16

Das S., M. Halappanavar, A. Tumeo, E. Serra, A. Pothan, and E. Al-Shaer. "VWC-BERT: Scaling Vulnerability–Weakness–Exploit Mapping 59 60 3 on Modern AI Accelerators." In *IEEE International Conference on Big Data (IEEE BigData 2022)* December 17-20, 2022. Osaka, Japan.

A Summary of Key Results



Relative Performance (temporal partition)

Table 5: Performance comparison of V2W-BERT

	Model	Test 1 (k_1, k_2, k_3)			Test 2 (k_1, k_2, k_3)		
		(1,1,1)	(3,2,1)	(5,2,2)	(1,1,1)	(3,2,1)	(5,2,2)
1-100	Class, TF-IDF NN	0.2631	0.5656	0.6537	0.2519	0.4838	0.5739
	Link, TF-IDF NN	0.3626	0.5998	0.6791	0.3395	0.564	0.659
	Class, BERT _{CVE}	0.4138	0.6602	0.7466	0.2914	0.6105	0.6902
	Link, V2W-BERT	0.4765	0.6933	0.7564	0.4072	0.6293	0.7179
>100	Class, TF-IDF NN	0.8524	0.9425	0.9616	0.7815	0.8953	0.9404
	Link, TF-IDF NN	0.8463	0.9227	0.9485	0.7604	0.8738	0.9153
	Class, BERT _{CVE}	0.8852	0.9479	0.9649	0.8067	0.9064	0.9414
	Link, V2W-BERT	0.8905	0.947	0.9763	0.8113	0.9123	0.9492
All	Class, TF-IDF NN	0.775	0.893	0.9298	0.6886	0.8231	0.8761
	Link, TF-IDF NN	0.7828	0.8803	0.9132	0.6863	0.8196	0.8706
	Class, BERT _{CVE}	0.8232	0.9101	0.9363	0.7163	0.8578	0.9038
	Link, V2W-BERT	0.8362	0.914	0.9442	0.7345	0.8594	0.9151

Randomly Partitioned (across years)

Table 4: Performance with randomly partitioned dataset

Model	Test Set (k_1, k_2, k_3)		
	(1,1,1)	(3,2,1)	(5,2,2)
Class, TF-IDF NN	0.8606	0.9464	0.9668
Link, TF-IDF NN	0.8642	0.9502	0.9693
Class, BERT _{CVE}	0.8812	0.9503	0.9689
Link, V2W-BERT	0.8916	0.9523	0.9723

Quality Across Models

- Best performance from RoBERTa
- Hierarchical Mapping

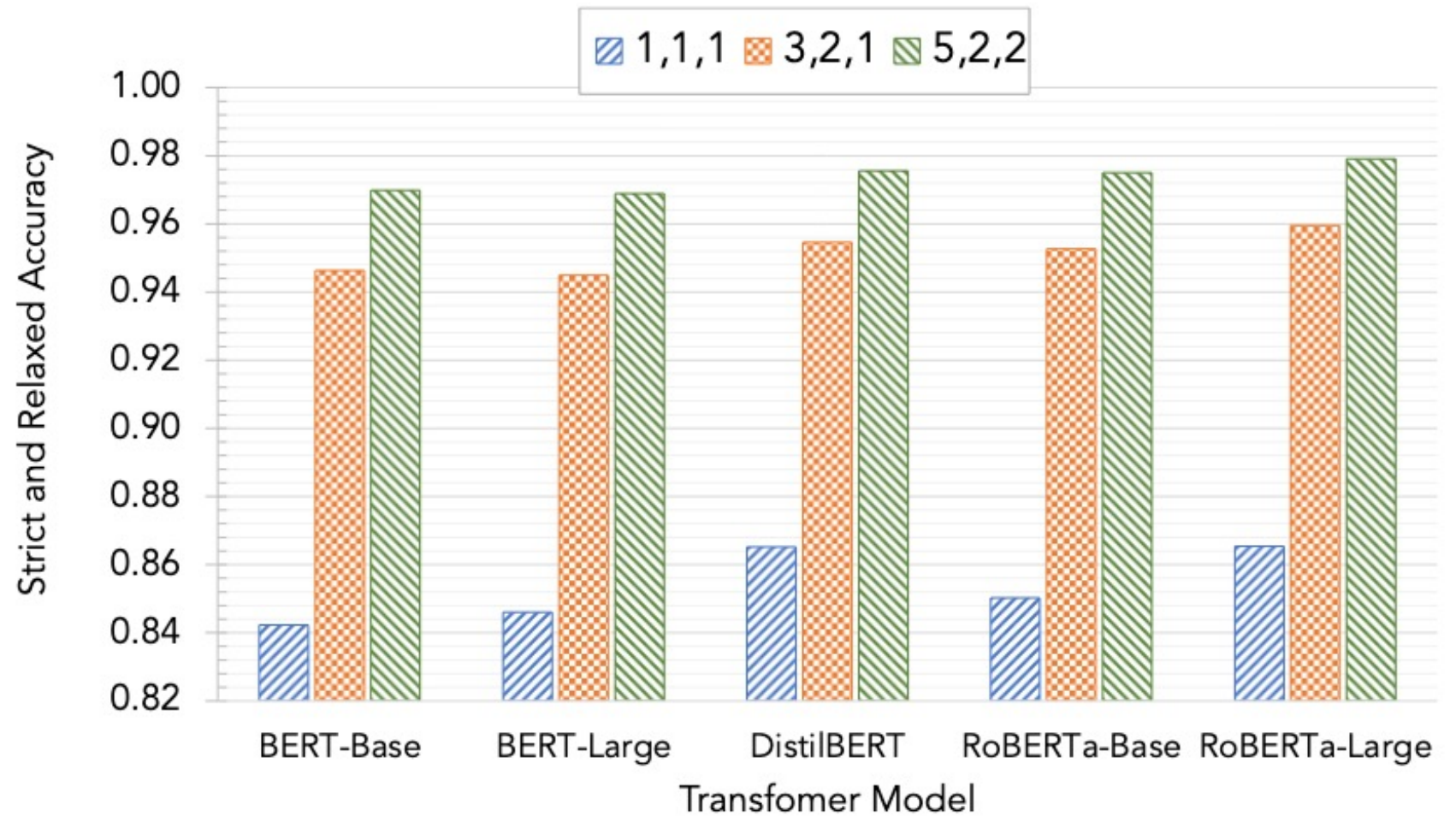
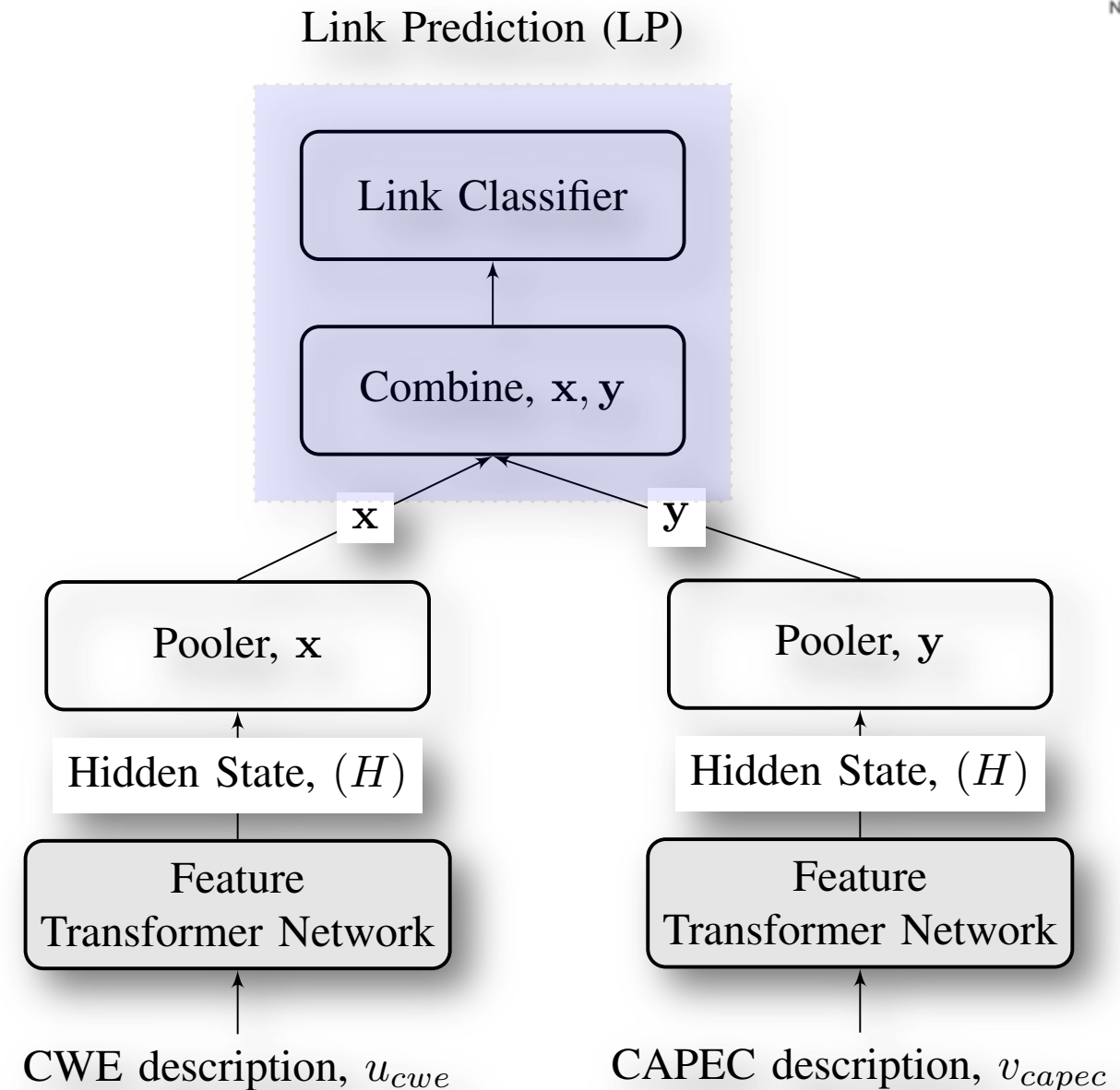


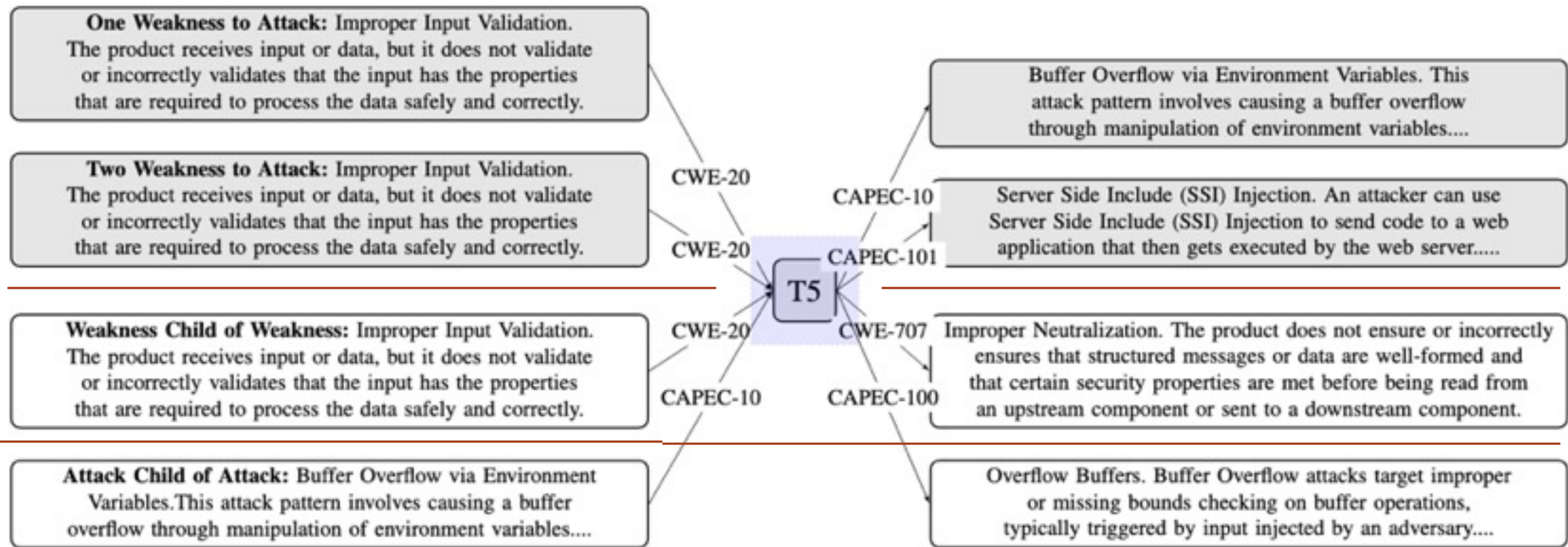
Fig. 3: Prediction accuracy of CVEs to CWEs using different language models. The (3,2,1) Refers to taking top 3 predictions from root, 2 from their children and 1 from leaf nodes of the CWE hierarchy.

CWE-CAPEC: VWC-MAP Framework: Approach1: Link prediction

Architectural overview of the Link Prediction network. The Feature Transformer components have shared weights. The model takes CWE-CAPEC feature information and transforms and combines them for prediction



CWE-CAPEC: VWC-MAP Framework Approach2: Using LLMs (Google T5)



Text-2-Text Mapping: Training process for VWC-MAP framework

CWE-CAPEC Mapping Result: CWE-131

CWE	Link Prediction		T5-model	
	CAPEC	Rating	CAPEC	Rating
CWE-131: Incorrect Calculation of Buffer Size	CAPEC-100: Overflow Buffers*	10	CAPEC-100: Overflow Buffers*	10
	CAPEC-47: Buffer Overflow via Parameter Expansion*	10	CAPEC-47: Buffer Overflow via Parameter Expansion*	10
	CAPEC-14: Client-side Injection-induced Buffer Overflow	8	CAPEC-14: Client-side Injection-induced Buffer Overflow	8
	CAPEC-24: Filter Failure through Buffer Overflow	10	CAPEC-24: Filter Failure through Buffer Overflow	10
	CAPEC-256: SOAP Array Overflow	10	CAPEC-67: String Format Overflow in syslog()	3
	CAPEC-45: Buffer Overflow via Symbolic Links	5	CAPEC-45: Buffer Overflow via Symbolic Links	5
	CAPEC-46: Overflow Variables and Tags	10	CAPEC-46: Overflow Variables and Tags	10
	CAPEC-8: Buffer Overflow in an API Call	3	CAPEC-8: Buffer Overflow in an API Call	3
* - Ground truth				

Predictions from our Link Prediction model for CWE-131. Both models predict the ground truth of two CAPECs perfectly. The additional predictions were evaluated manually and we have found them to be highly relatable. It's also interesting to see both of these give almost same predictions with one difference.

CWE-CAPEC Mapping Result: CWE-22

CWE	Link Prediction		T5-model	
	CAPEC	Rating	CAPEC	Rating
CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	CAPEC-126: Path Traversal*	10	CAPEC-126: Path Traversal*†	×
	CAPEC-64: Using Slashes and URL Encoding	×	CAPEC-64: Using Slashes and URL Encoding	10
	Combined to Bypass Validation Logic*†		Combined to Bypass Validation Logic	
	CAPEC-76: Manipulating Web Input to File System Calls*	10	CAPEC-76: Manipulating Web Input to File System Calls*	10
	CAPEC-78: Using Escaped Slashes in Alternate Encoding*†	×	CAPEC-78: Using Escaped Slashes in Alternate Encoding	10
	CAPEC-79: Using Slashes in Alternate Encoding*†	×	CAPEC-79: Using Slashes in Alternate Encoding*†	×
	CAPEC-597: Absolute Path Traversal	10	CAPEC-80: Using UTF-8 Encoding to Bypass Validation Logic	10
	CAPEC-139: Relative Path Traversal	10	CAPEC-139: Relative Path Traversal	10
		CAPEC-3: Using Leading 'Ghost' Character Sequences to Bypass Input Filters	5	

* - Ground truth † - not predicted

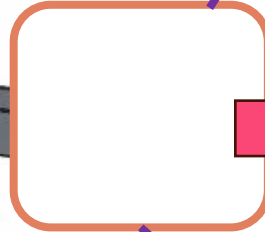
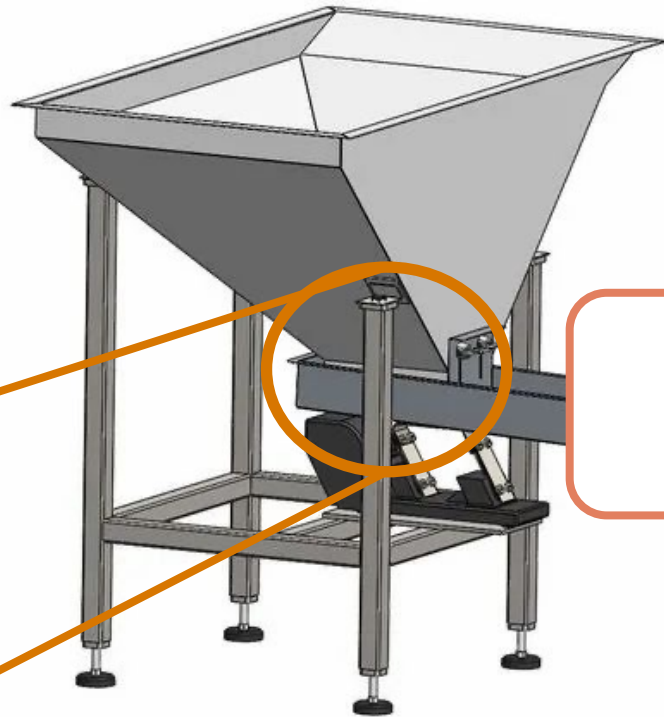
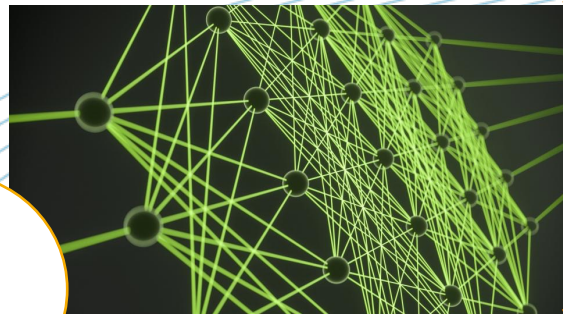
Predictions from our Link Prediction model for CWE-22. We kept the original CWE-CAPEC mapping hidden during training time. We can see Link prediction model predicted two out of five CAPECs successfully, and the suggested two CAPECs match the context. The T5-model predicted three CAPECs successfully, and among the suggestions, two of the three CAPECs match contexts.

3. Insights and Next Steps

1



2



```
int num = 50;  
int[] x = new int[num];  
int[] y = new int[num];  
  
void setup() {  
  size(100, 100);  
  noStroke();  
  fill(255, 102);  
}  
  
void draw() {  
  background(0);  
  // Shift the values to the right  
  for (int i = num-1; i > 0; i--) {  
    x[i] = x[i-1];  
    y[i] = y[i-1];  
  }  
  // Add the new values to the beginning of the array
```

3



Vulnerability Exploration (<http://enigma.pnl.gov:8501/>)

CVE Home

SOM Cluster

CAPEC Cluster

CVSS Metric

cvss_v2 cvss_v3

Time range

2010 2023

2010 2023

CVE Lookup

CVE-2017-11882 ▼

CVE-2017-11882

2017-11-15 03:29:00 UTC

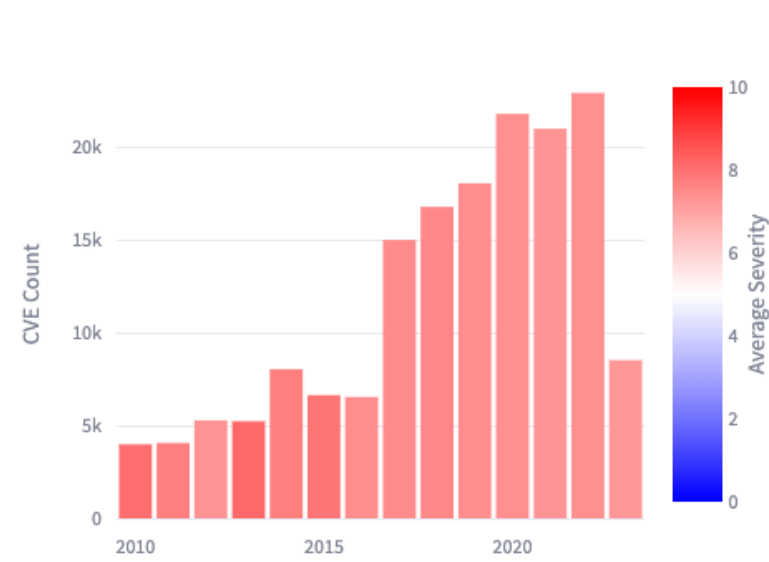
Score: 7.8

Microsoft Office 2007 Service Pack 3, Microsoft Office 2010 Service Pack 2, Microsoft Office 2013 Service Pack 1, and Microsoft Office 2016 allow an attacker to run arbitrary code in the context of the current user by failing to properly handle objects in memory, aka "Microsoft Office Memory Corruption Vulnerability". This CVE ID is unique from CVE-2017-11884.

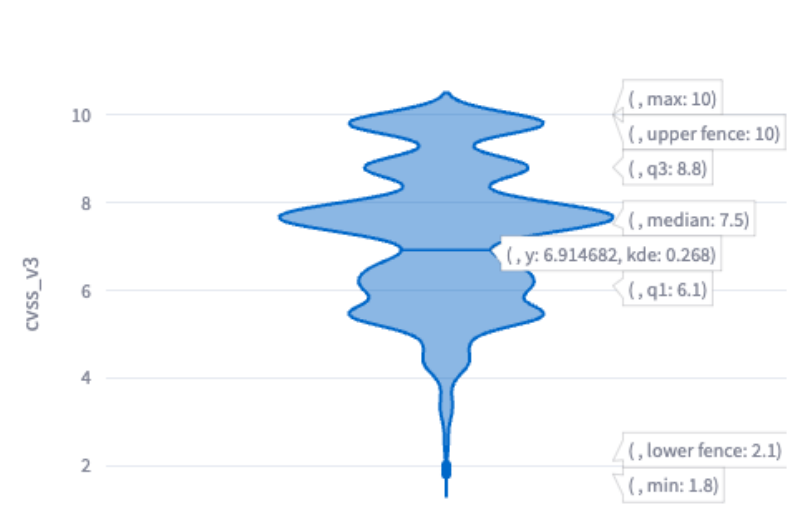
Vector: 'CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:+

Annual Monthly 2023 2022 2021 2020 2019 2018 2017 2016 2015 2014 2013

CVE Histogram



CVE Severity Score



	publishedDate	CVE ID	CVE Description	impact.baseM
1	2017-02-13 21:59:00+00:00	CVE-2017-5145	An issue was discovered in Carlo Gavazzi VMU-C EM prior to firmware Version A11_U0	CVSS:3.0/AV:N,
2	2018-08-20 19:31:00+00:00	CVE-2018-1000652	JabRef version <=4.3.1 contains a XML External Entity (XXE) vulnerability in MsBibImp	CVSS:3.0/AV:N,
3	2018-08-20 19:31:00+00:00	CVE-2018-1000651	Stroom version <5.4.5 contains a XML External Entity (XXE) vulnerability in XML Parser	CVSS:3.0/AV:N,
4	2018-08-20 19:31:00+00:00	CVE-2018-1000644	Eclipse RDF4j version < 2.4.0 Milestone 2 contains a XML External Entity (XXE) vulneral	CVSS:3.0/AV:N,
5	2018-11-07 05:29:00+00:00	CVE-2018-19047	** DISPUTED ** mPDF through 7.1.6, if deployed as a web application that accepts ar	CVSS:3.0/AV:N,
6	2019-07-02 17:15:00+00:00	CVE-2019-7268	Linear eMerge 50P/5000P devices allow Unauthenticated File Upload.	CVSS:3.1/AV:N,

CWE Lookup

CWE-1188

CWE-1188

Description

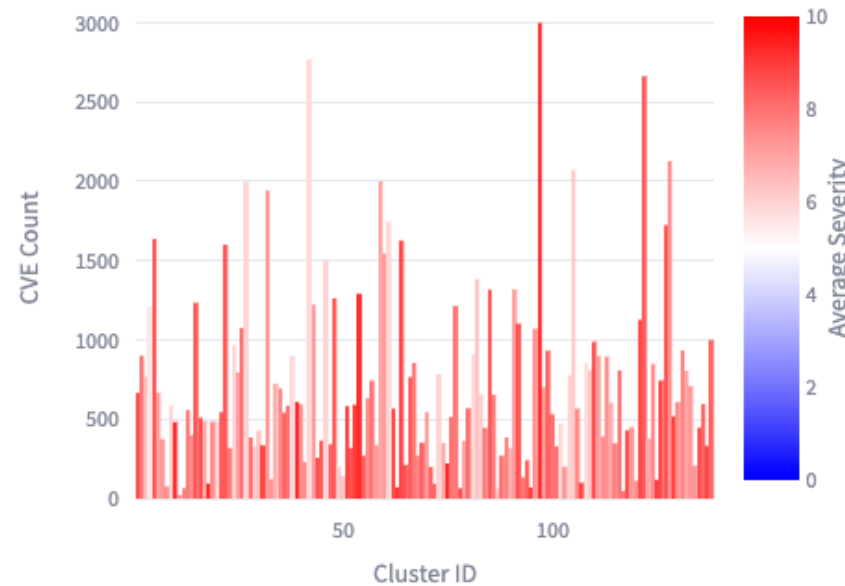
Insecure Default Initialization of Resource The software initializes or sets a resource with a default that is intended to be changed by the administrator, but the default is not secure. Developers often choose default values that leave the software as open and easy to use as possible out-of-the-box, under the assumption that the administrator can (or should) change the default

SOM Cluster

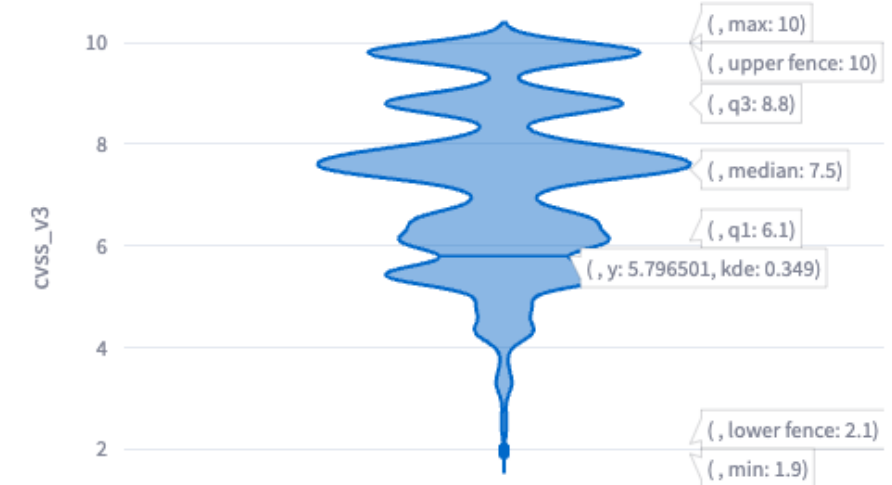
CVE Count Histogram Average Severity Histogram

Overall Cluster 1 Cluster 2 Cluster 3

CVE Count by Cluster



CVE Severity Score



CWEs for All Clusters

CWEs for Cluster 1

CWEs for Cluster 2

CWEs for Cluster 3

Other CWE-434 CWE-20 CWE-200

Case Studies

- Microsoft Office Memory Corruption (**CVE 2017-11882**):
 - This Microsoft Office software bug allows attackers to execute arbitrary code on the user's system by convincing the user to open a malicious file. It was patched in a later version of Office.
 - **CWE-119** (from V2W-BERT): Improper Restriction of Operations within the Bounds of a Memory Buffer --- The software performs operations on a memory buffer, but it can read from or write to a memory location that is outside of the intended boundary of the buffer.
- Citrix Netscaler Directory Traversal (CVE-2019-19781):
 - An issue was discovered in Citrix Application Delivery Controller (ADC) and Gateway 10.5, 11.1, 12.0, 12.1, and 13.0. They allow Directory Traversal.
 - CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') The software uses external input to construct a pathname that is intended to identify a file or directory that is located underneath a restricted parent directory, but the software does not properly neutralize special elements within the pathname that can cause the pathname to resolve to a location that is outside of the restricted directory

CISA CYBERSECURITY ADVISORY: Top Routinely Exploited Vulnerabilities
August 20, 2021 -- Alert Code AA21-209A

CVE-2017-11882

CVE-2017-11882

2017-11-15 03:29:00 UTC

Score: 7.8

Microsoft Office 2007 Service Pack 3, Microsoft Office 2010 Service Pack 2, Microsoft Office 2013 Service Pack 1, and Microsoft Office 2016 allow an attacker to run arbitrary code in the context of the current user by failing to properly handle objects in memory, aka "Microsoft Office Memory Corruption Vulnerability". This CVE ID is unique from CVE-2017-11884.

Vector: 'CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H'

More information: <https://www.cvedetails.com/cve/CVE-2017-11882>

CWEs

['CWE-119']

The CVE to CWE mappings are provided by employing the V2W-BERT tool using the DistilBERT model. These mappings have not been validated by subject matter experts and should be used only as a suggestion.

CWE Lookup

CWE-119

CWE-119

Description

Improper Restriction of Operations within the Bounds of a Memory Buffer The software performs operations on a memory buffer, but it can read from or write to a memory location that is outside of the intended boundary of the buffer. Certain languages allow direct addressing of memory locations and do not automatically ensure that these locations are valid for the memory buffer that is being referenced. This can cause read or write operations to be performed on memory locations that may be associated with other variables, data structures, or internal program data. As a result, an attacker may be able to execute arbitrary code, alter the intended control flow, read sensitive information, or cause the system to crash.

::SCOPE: Integrity:SCOPE: Confidentiality:SCOPE: Availability:IMPACT

Unauthorized Code or Commands:IMPACT

Memory:NOTE

the memory accessible by the attacker can be effectively controlled, it may be possible to execute arbitrary code, as with a standard buffer overflow. If the attacker can overwrite a pointer's worth of memory (usually 32 or 64 bits), they can redirect a function pointer to their own malicious code. Even when the attacker can only modify

CWE Lookup

CWE-119

CWE-119

Description

CWEs

Linked Clusters: 2, 4, 5, 8, 10, 13, 14, 15, 17, 21, 22, 23, 25, 26, 28, 31, 32, 34, 35, 36, 37, 43, 44, 45, 46, 48, 56, 57, 58, 62, 63, 64, 66, 70, 71, 72, 77, 80, 81, 89, 92, 94, 95, 96, 98, 99, 100, 101, 105, 106, 107, 109, 110, 111, 112, 113, 114, 115, 116, 120, 122, 125, 127, 128, 129, 131, 132, 133, 134, 135, 136

[View linked clusters](#)

- Cluster 122 x
- Cluster 125 x
- Cluster 127 x
- Cluster 128 x
- Cluster 129 x
- Cluster 131 x
- Cluster 132 x
- Cluster 133 x
- Cluster 134 x
- Cluster 135 x

CWE Lookup

CWE-119

CWE-119

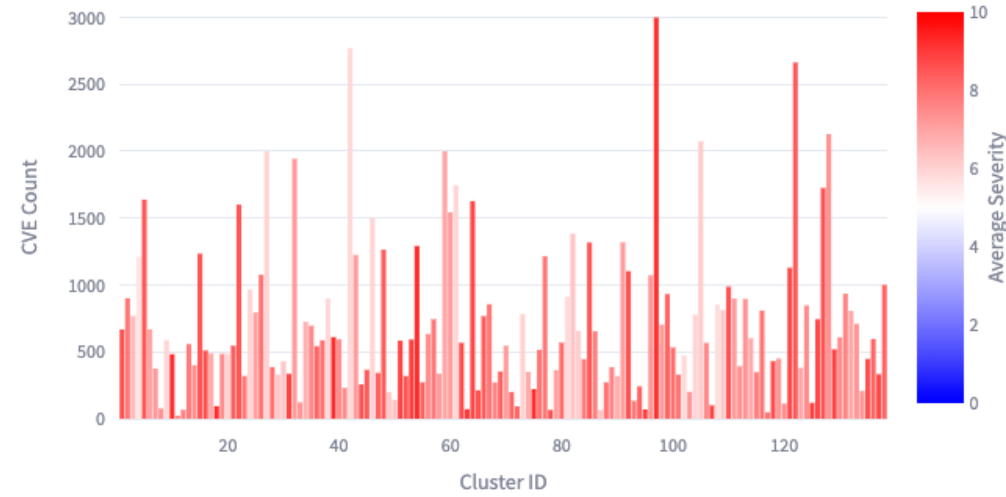
Description

CVEs

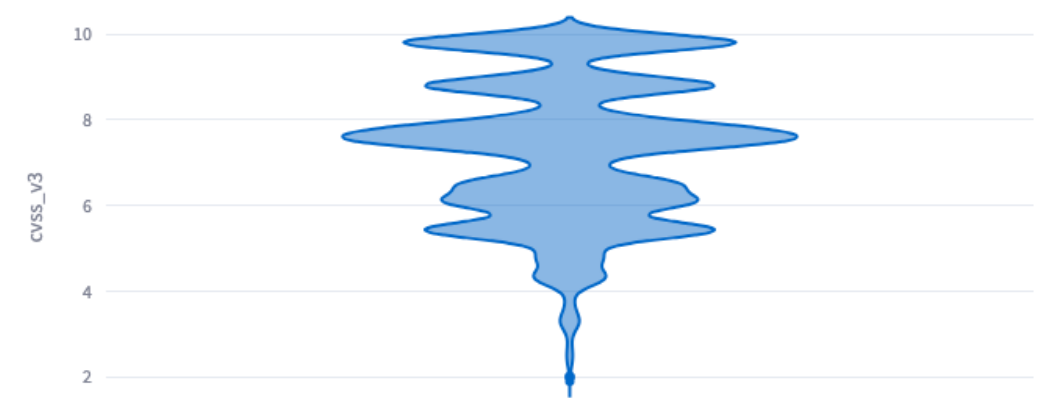
Linked Clusters: 2, 4, 5, 8, 10, 13, 14, 15, 17, 21, 22, 23, 25, 26, 28, 31, 32, 34, 35, 36, 37, 43, 44, 45, 46, 48, 56, 57, 58, 62, 63, 64, 66, 70, 71, 72, 77, 80, 81, 89, 92, 94, 95, 96, 98, 99, 100, 101, 105, 106, 107, 109, 110, 111, 112, 113, 114, 115, 116, 120, 122, 125, 127, 128, 129, 131, 132, 133, 134, 135, 136

View linked clusters

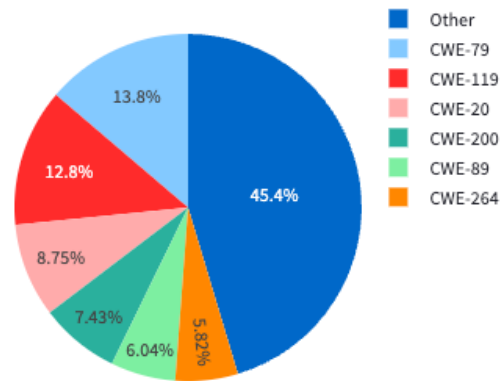
CVE Count by Cluster



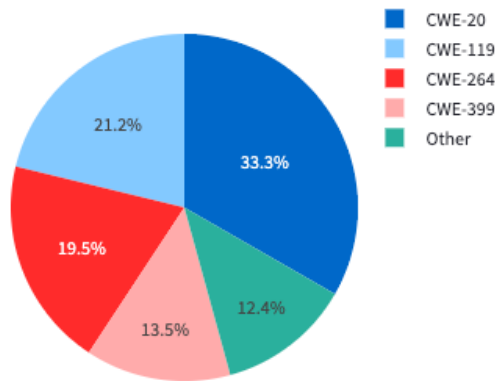
CVE Severity Score



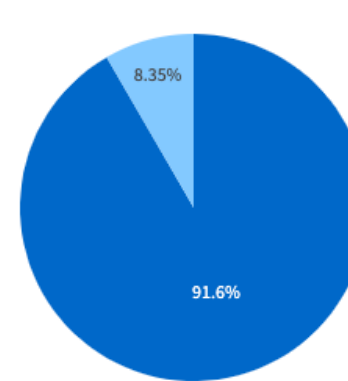
CWEs for All Clusters



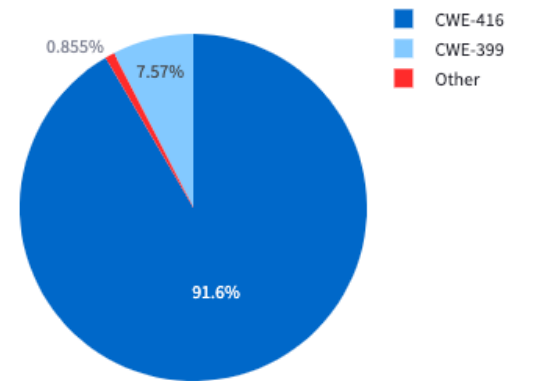
CWEs for Cluster 2



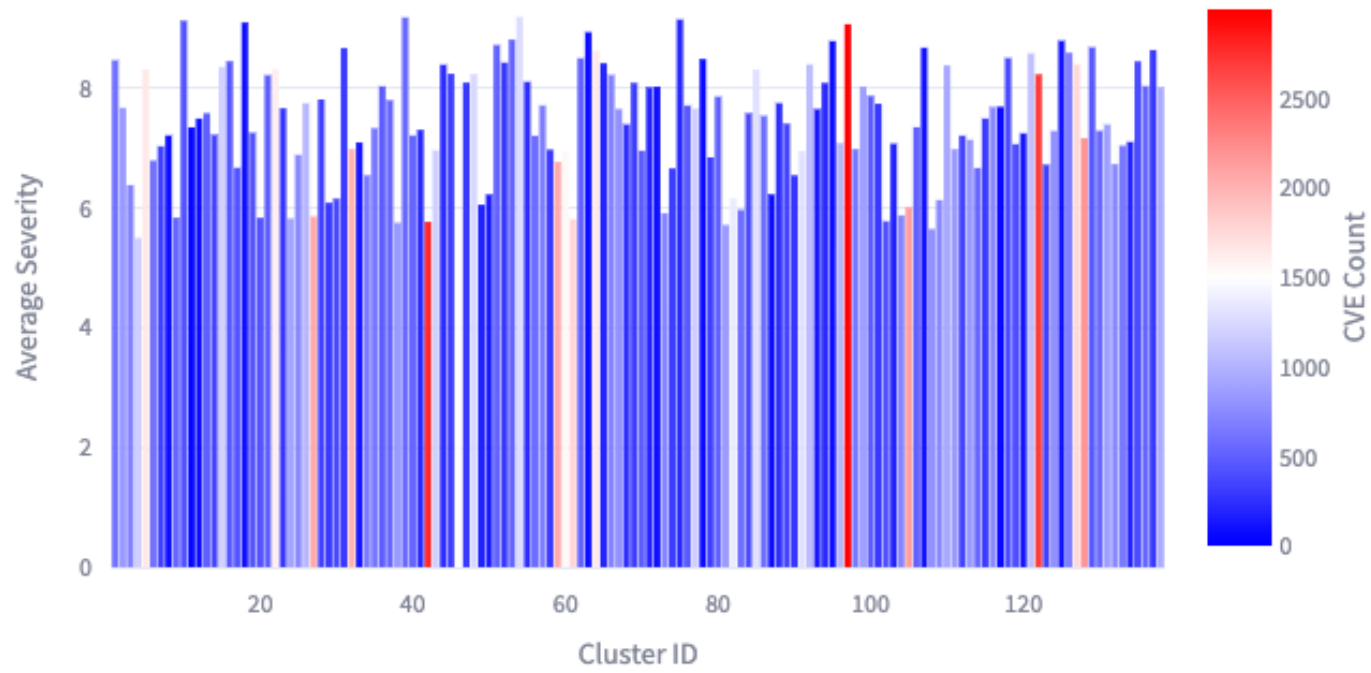
CWEs for Cluster 4



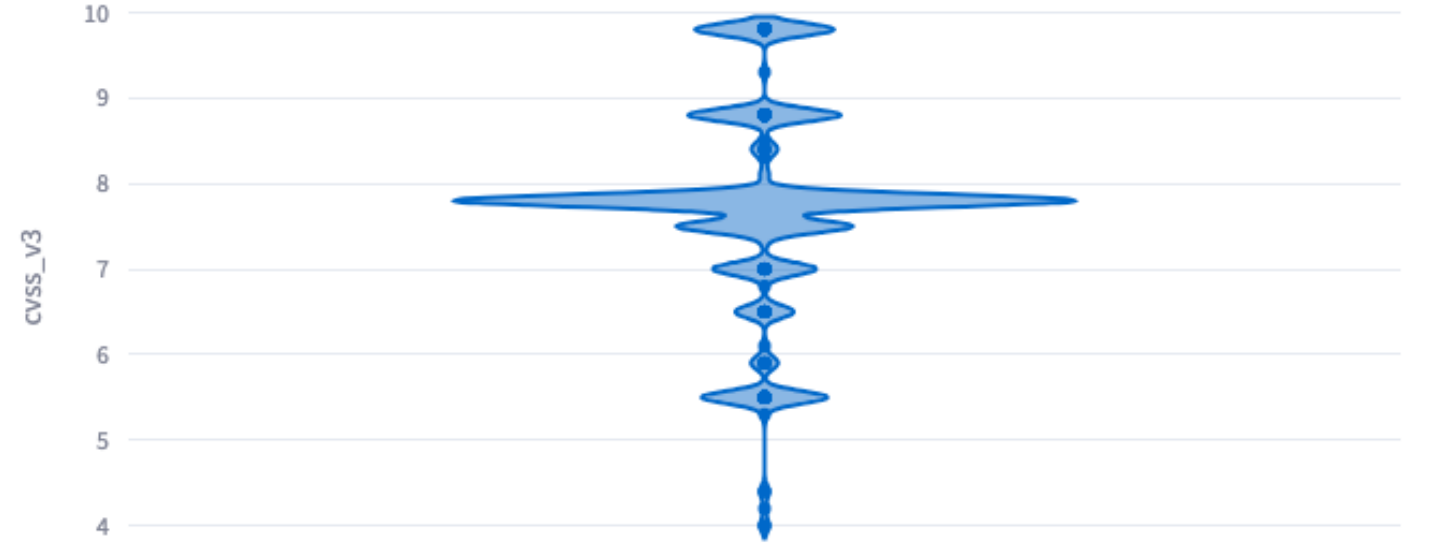
CWEs for Cluster 5



Average Severity by Cluster

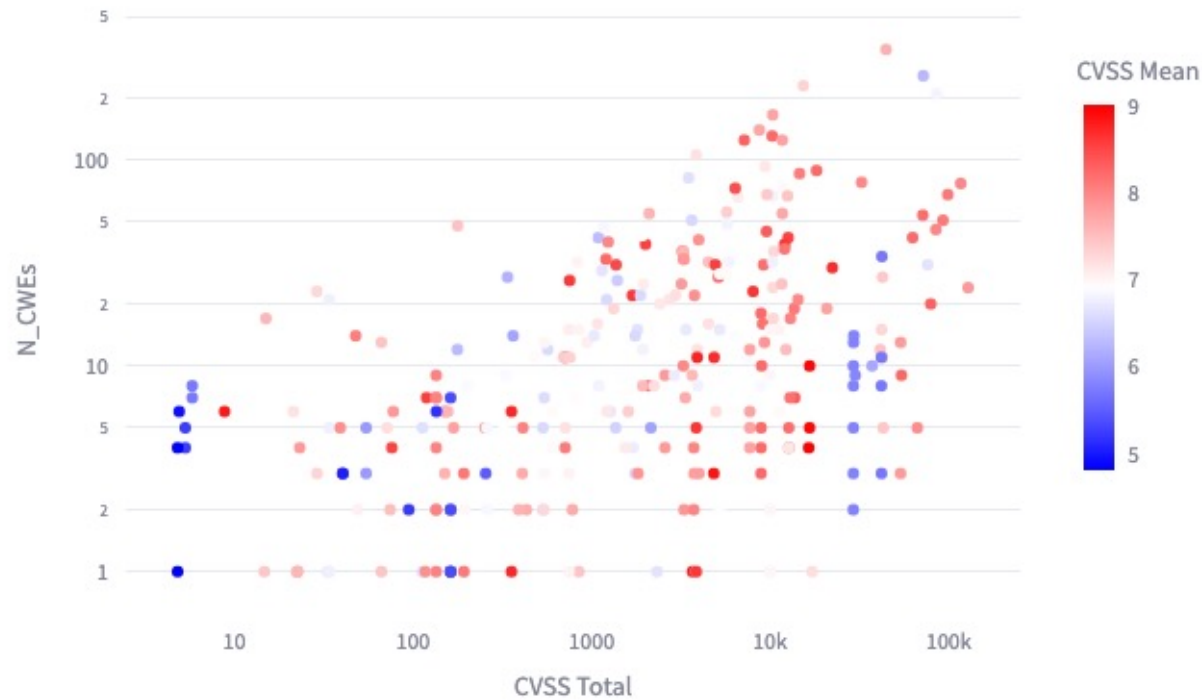


CVE Severity Score

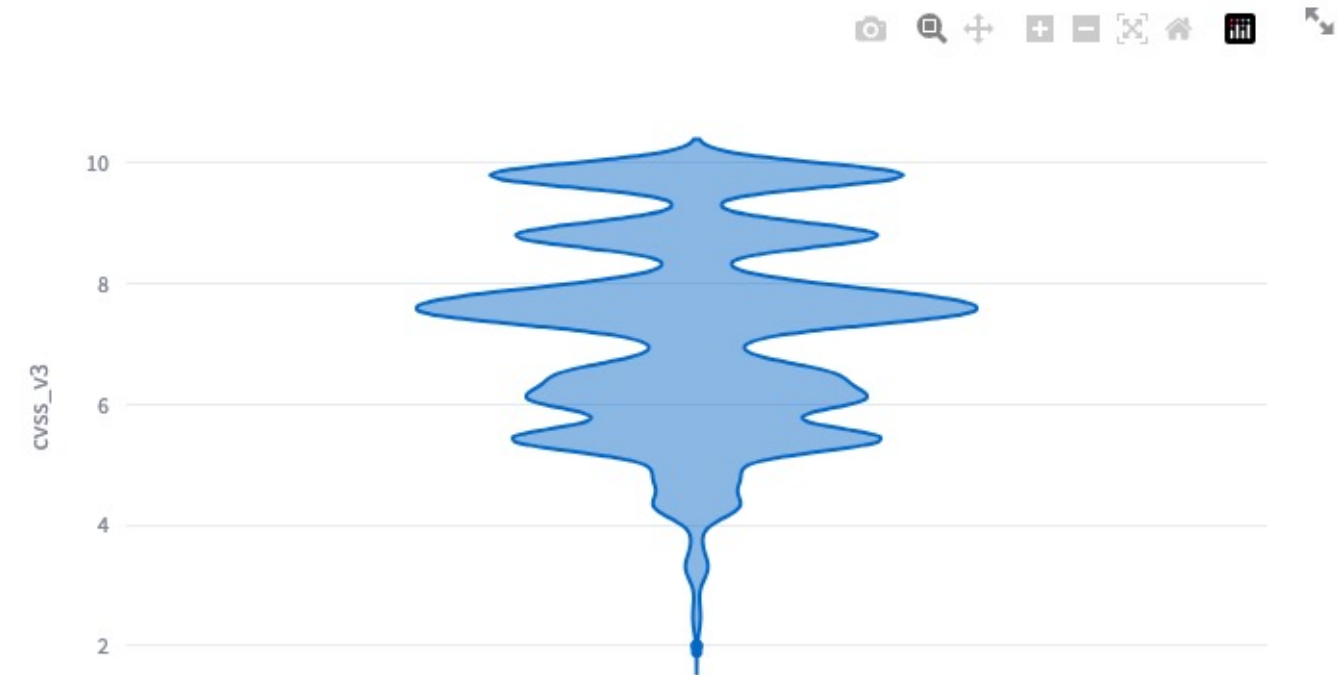


CAPEC Explorer

CWE Count and Severity CAPEC Total Severity vs Average Severity



Overall Buffer Overflow via Environment Variables MIME Conversion Buffer Overflow via Parameter Expansion



ID	CWEs	CVSS Total	CVSS Mean	N_CWEs	Name	Abstraction	Status
1	CWE-8 CWE-110 CWE-166 CWE-276 CWE-277 CWE-278 CWE-280	11,708.6	7.6327	55	Accessing Functionality Not Properly Constrained by ACLs	Standard	Draft
2	CWE-447 CWE-645	0	None	2	Inducing Account Lockout	Standard	Draft
3	CWE-8 CWE-24 CWE-26 CWE-28 CWE-29 CWE-30 CWE-32 CWE-33	10,270.4	8.1317	131	Using Leading 'Ghost' Character Sequences to Bypass Input Filters	Detailed	Draft
4	CWE-173 CWE-291 CWE-925	0	None	3	Using Alternative IP Address Encodings	Detailed	Draft
5	CWE-111 CWE-178 CWE-221 CWE-241 CWE-248 CWE-253 CWE-273	5,897.6	6.914	36	Blue Boxing	Detailed	Draft
6	CWE-78 CWE-141 CWE-142 CWE-143 CWE-145 CWE-146 CWE-153	8,028.7	8.5777	23	Argument Injection	Standard	Draft

CVE Home

SOM Cluster

CAPEC Cluster

CWE mapping

T5 Original

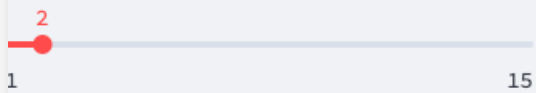
T5 Predicted

T5 Original

MITRE Related Weaknesses

CAPEC selection type

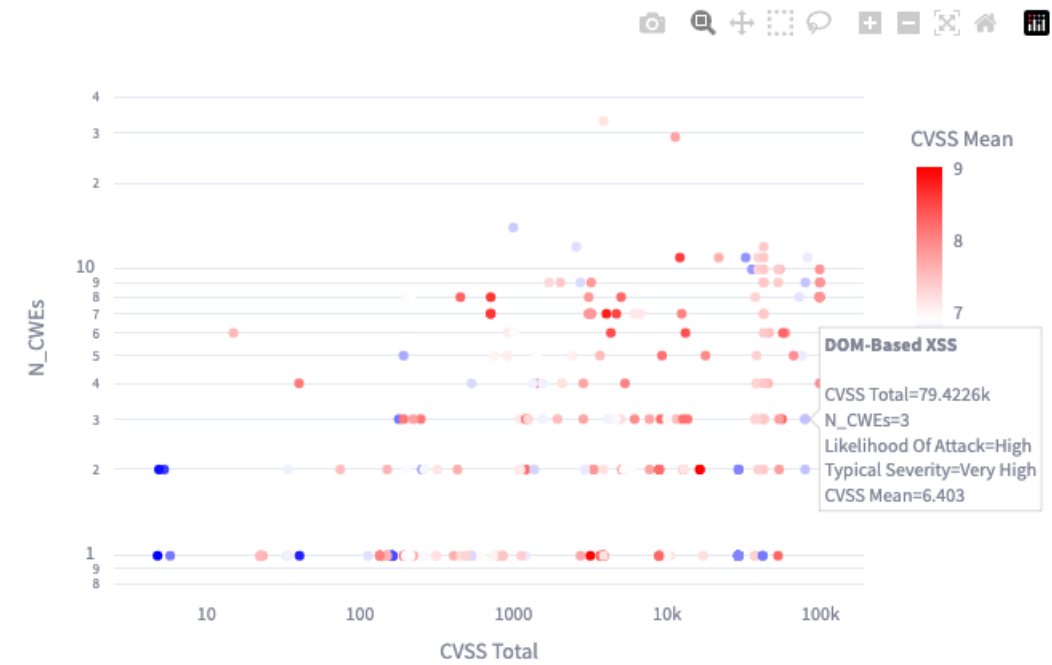
Top N by count



CAPEC Explorer

CWE Count and Severity CAPEC Total Severity vs Average Severity

Overall Exploiting Incorrectly Configured Access Control Security Levels Accessing Functionality Not Properly



Accessing Functionality Not Properly Constrained by ACLs

ID	CWEs	CVSS Total	CVSS Mean	N_CWEs	Name	Abstraction	Status
1	CWE-276 CWE-285 CWE-434 CWE-693 CWE-732 CWE-1191 CWE-1195	11,325	7.6572	29	Accessing Functionality Not Properly Constrained by ACLs	Standard	Draft
2	CWE-645	0	None	1	Inducing Account Lockout	Standard	Draft
3	CWE-20 CWE-41 CWE-74 CWE-172 CWE-173 CWE-179 CWE-180	42,672.4	7.3472	12	Using Leading 'Ghost' Character Sequences to Bypass Input Filters	Detailed	Draft

Summary & Conclusions

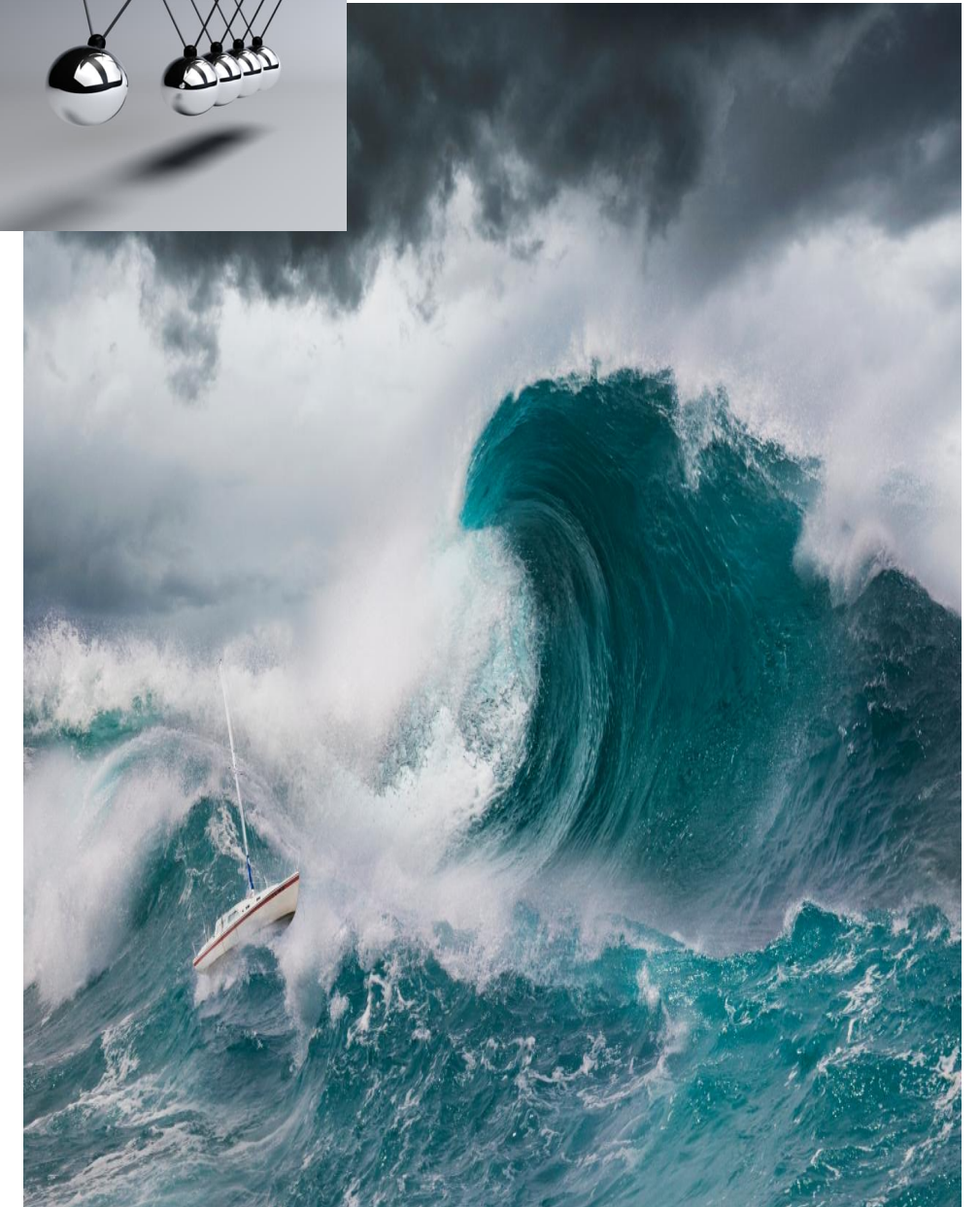


Summary of Contributions

- **First work** to provide complete mapping of CVEs to CWEs to CAPECs
- Showed how **Siamese link predictions** and large language models can be used for high quality mappings
- **Scaled** V2W-BERT on several generations of Nvidia systems and a GraphCore system
- Classified both frequent and **rarely occurring** CVEs better than all existing approaches
- Classified CVEs while maintaining the **hierarchical** relationships

Future Work

- Work with **subject matter experts** to perform validation and verification of mappings and clusters
- Enhance transfer learning techniques to classify CVEs/CWEs/CAPECs with few/zero training examples
- Build ability to predict if/when **new CWE** definitions are necessary
- Enhance mechanism to incorporate **novel** definitions over time
- Enhance the demonstration website

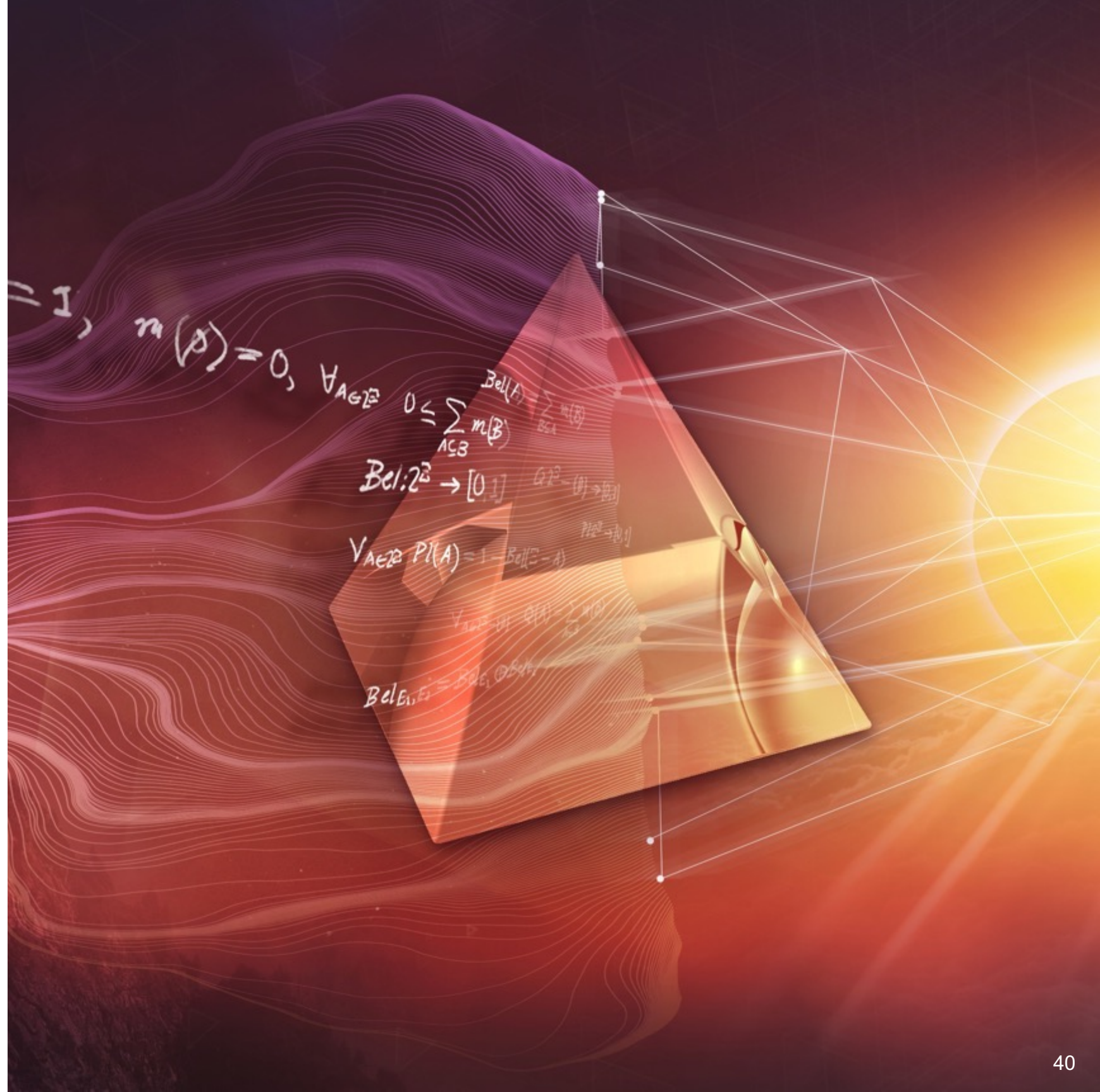


The coming tsunami from AI disruptions

Sponsors:

- PNNL Lab Directed Research & Development
 - The Mathematics for Artificial Reasoning in Science (MARS) initiative
 - The Data Model Convergence (DMC) initiative
- Department of Energy's Office of Advanced Scientific Computing Research
- Department of Defense

Thank you



References

- S Das, E. Serra, M. Halappanavar, A. Pothen, and E. Al-Shaer. "V2W-BERT: A Framework for Effective Hierarchical Multiclass Classification of Software Vulnerabilities." In proceedings of the *8th IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. Porto, Portugal. October 2021. **[Best Application Paper Award]**
- S. Das, A. Dutta, S. Purohit, E. Serra, M. Halappanavar and A. Pothen, "Towards Automatic Mapping of Vulnerabilities to Attack Patterns using Large Language Models," *2022 IEEE International Symposium on Technologies for Homeland Security (HST)*, Boston, MA, USA, 2022, pp. 1-7, doi: 10.1109/HST56032.2022.10025459. **[Best Paper Award in Cyber Security Track]**
- K. Panchal, S. S. Das, L. De La Torre, J. Miller, R. Rallo and M. Halappanavar, "Efficient Clustering of Software Vulnerabilities using Self Organizing Map (SOM)," *2022 IEEE International Symposium on Technologies for Homeland Security (HST)*, Boston, MA, USA, 2022, pp. 1-7, doi: 10.1109/HST56032.2022.10025443.
- Das S., M. Halappanavar, A. Tumeo, E. Serra, A. Pothen, and E. Al-Shaer. "VWC-BERT: Scaling Vulnerability–Weakness–Exploit Mapping 59 60 3 on Modern AI Accelerators." In *IEEE International Conference on Big Data (IEEE BigData 2022)* December 17-20, 2022. Osaka, Japan.

Self Organizing Maps (Kohonen Maps)

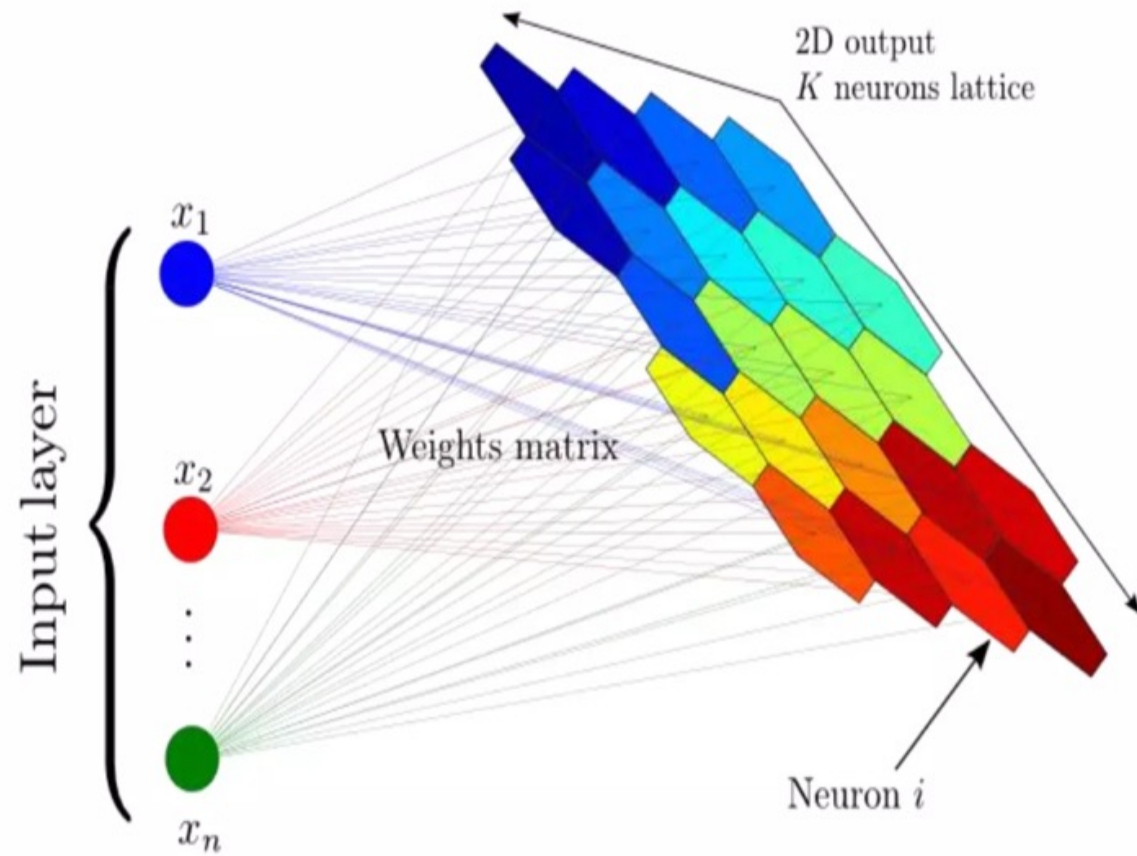
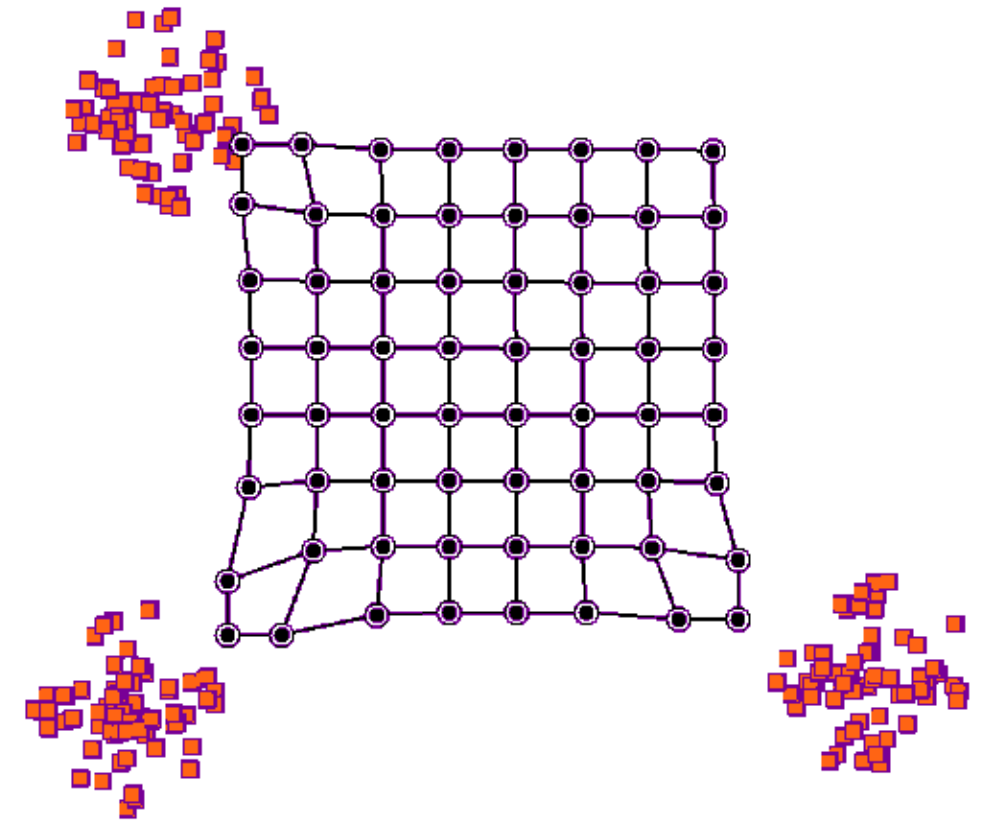


Image Adapted From: arxiv.org/pdf/1312.5753.pdf

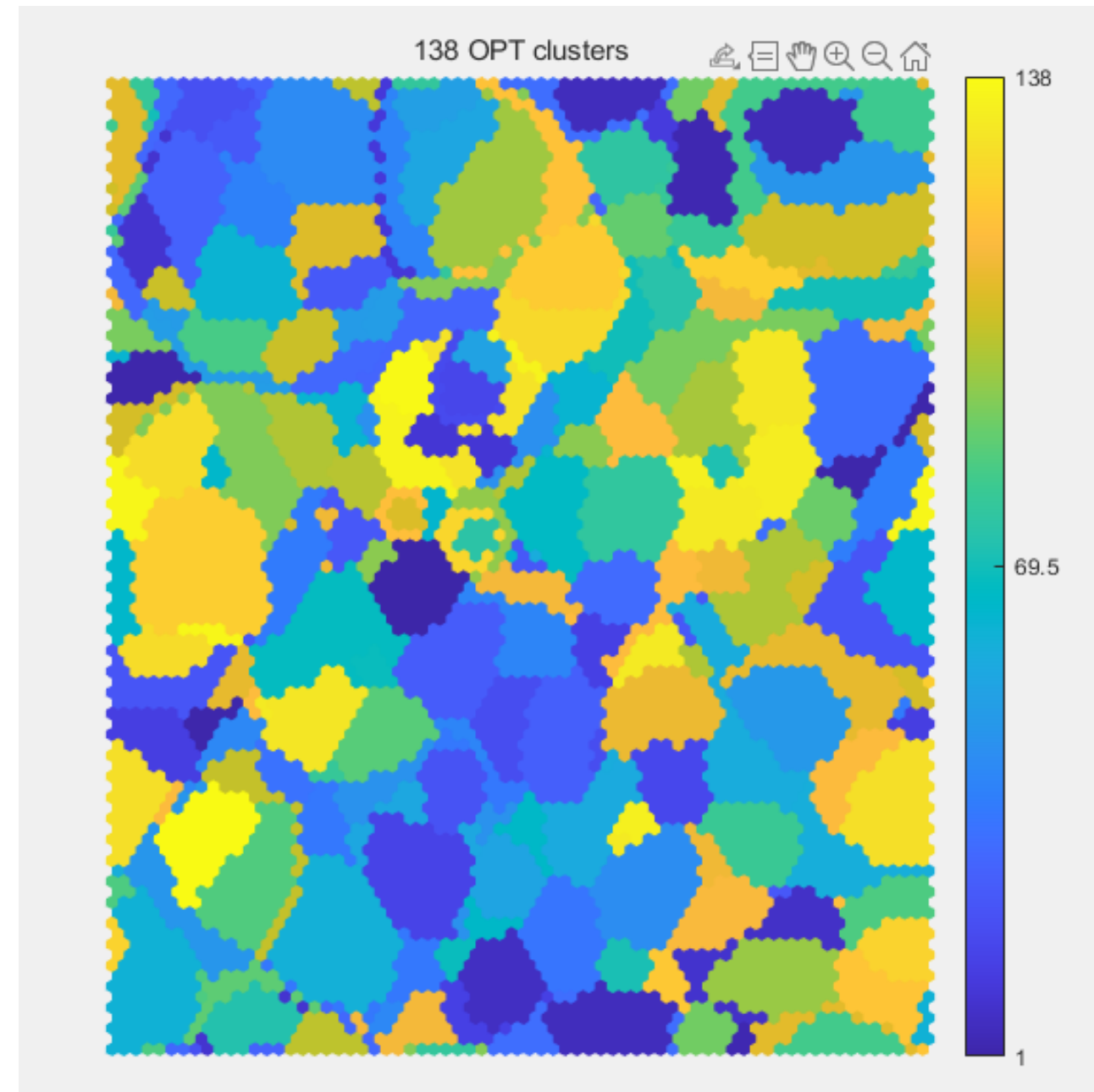
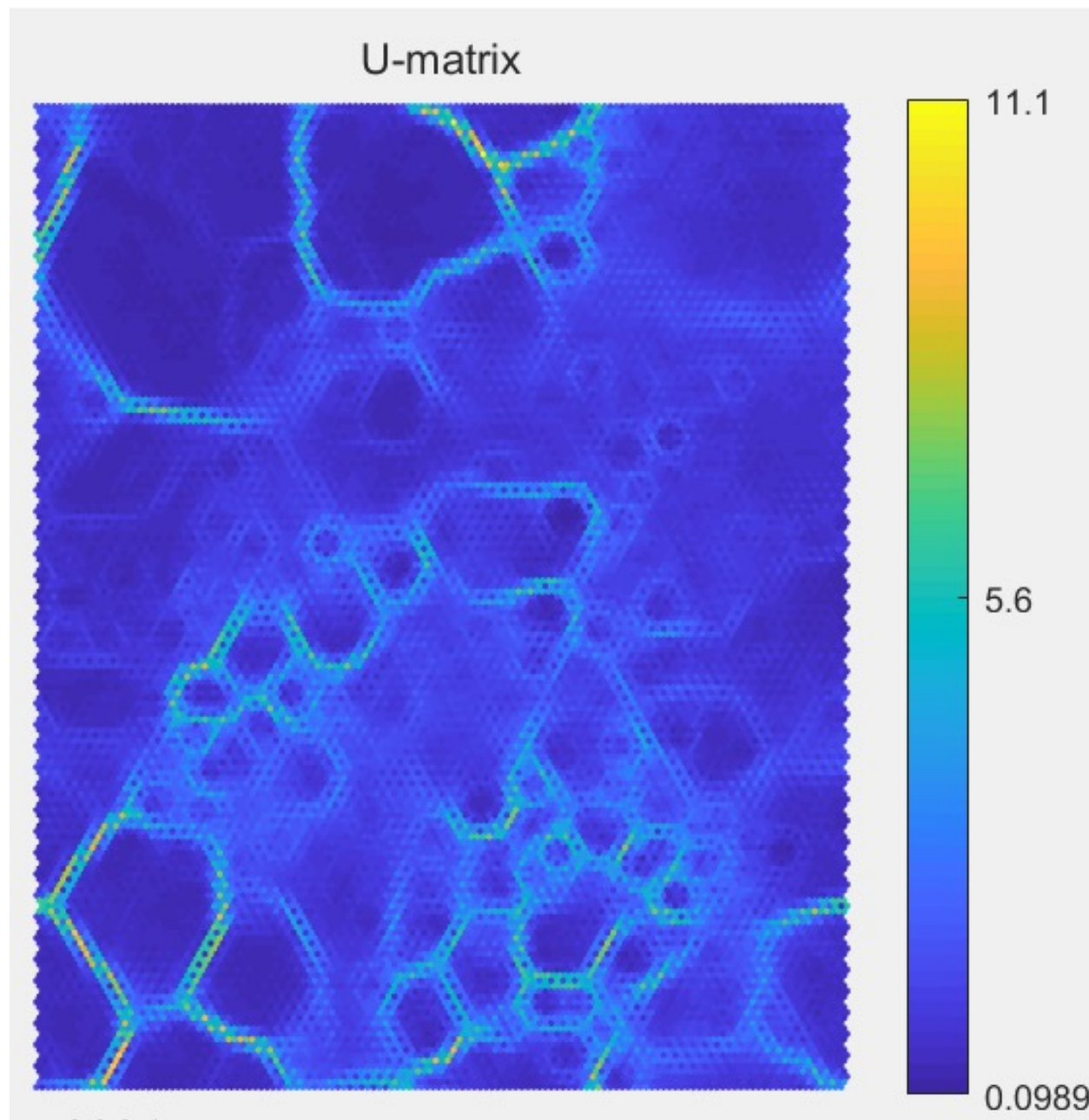


Source: https://en.wikipedia.org/wiki/Self-organizing_map

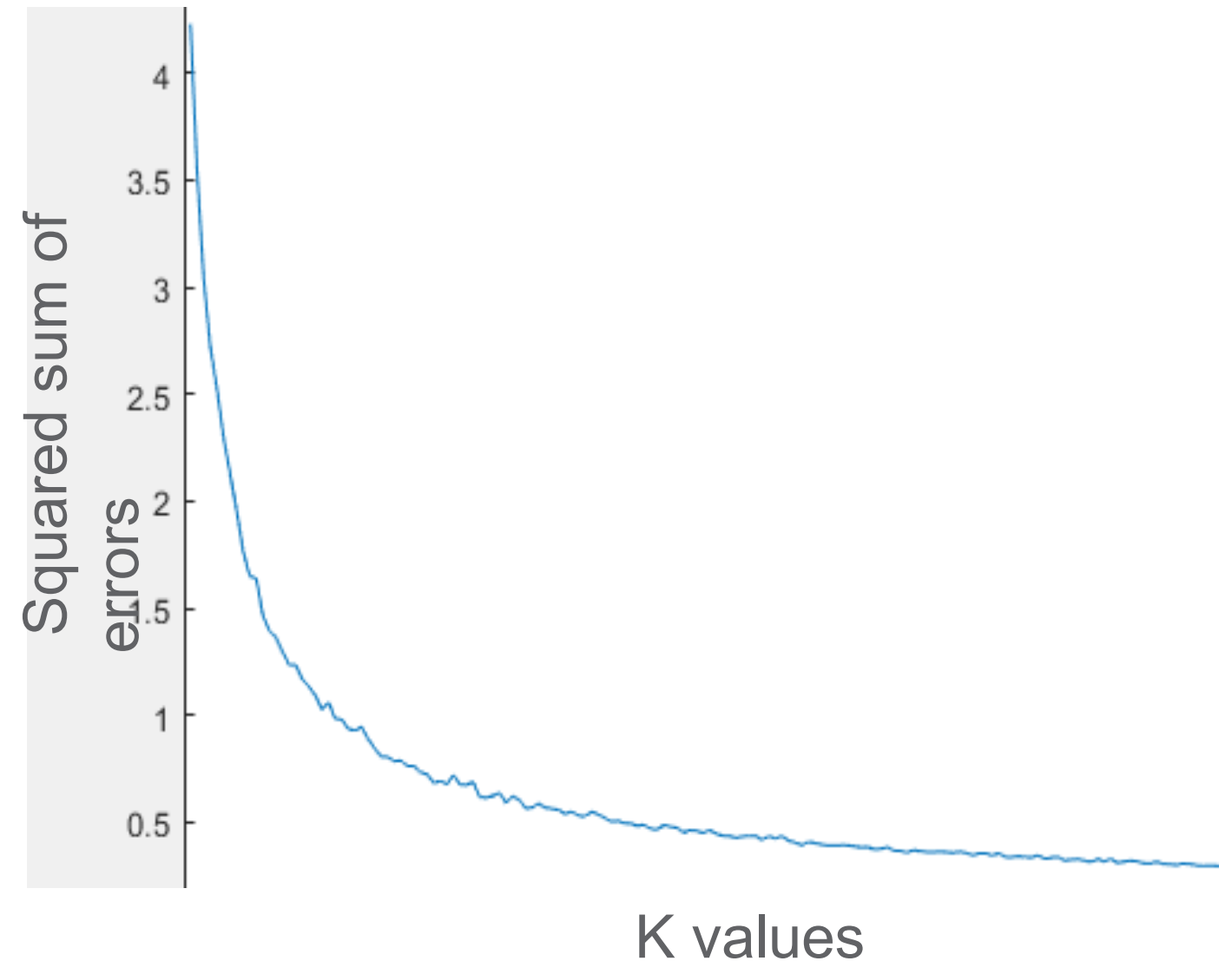
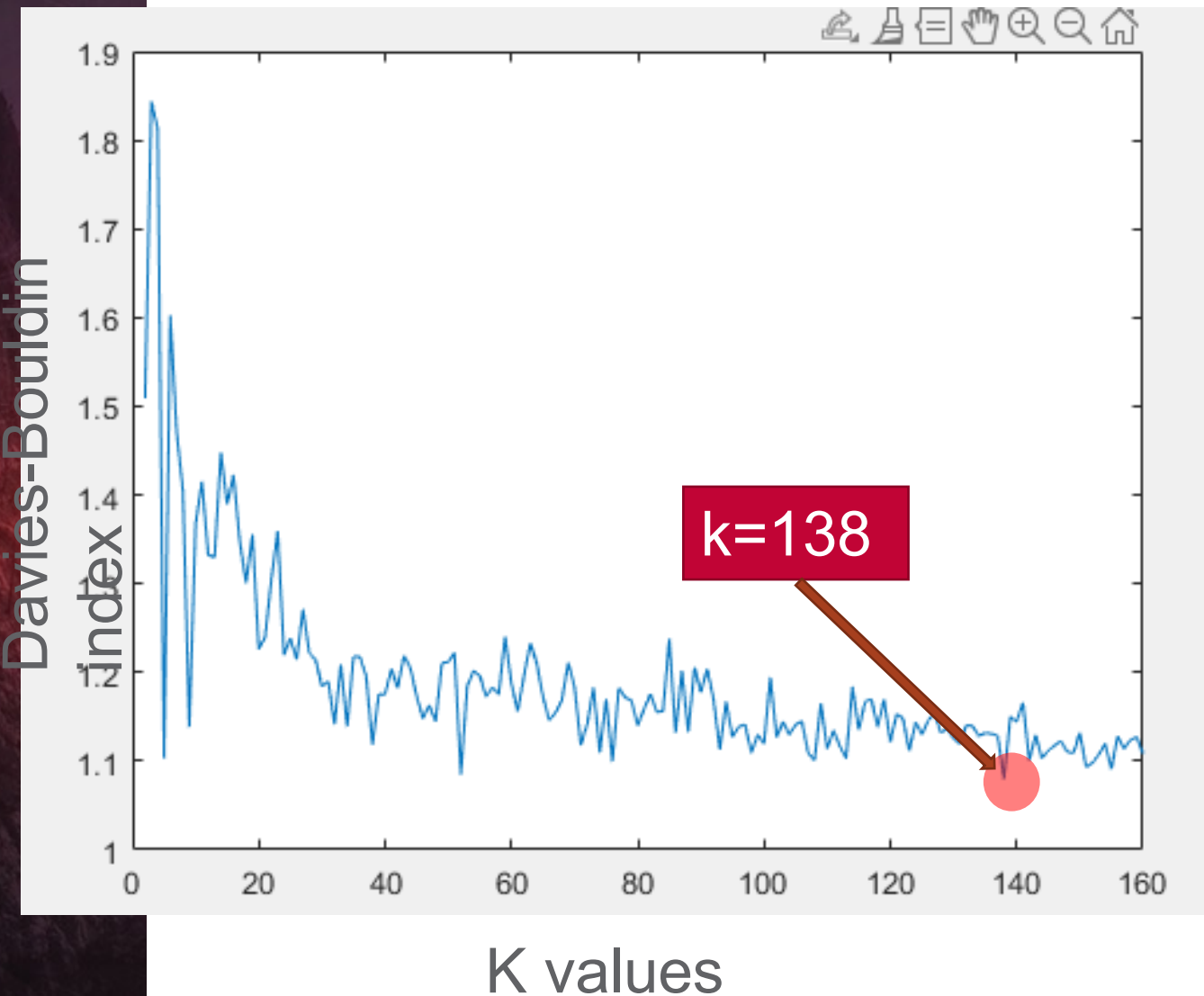
SOM Training: Output from V2W-BERT (~100k CVEs)

- `sD = som_read_data('V2W-LINK-distilbert-base-uncased-dp_rep.txt');`
- `sD` is a struct with `99950×768` elements.
- `sM=som_make(sD,'shape','toroid','mapsize','big','training','long','tracking',0);`
- To Create, initialize and train Self-Organizing Map
- `sM` struct with fields:
 - type: 'som_map'
 - codebook: `[6417×768 double]`
 - topol: `[1×1 struct]`
 - labels: `{6417×1 cell}`
 - neigh: 'gaussian'
 - mask: `[768×1 double]`
 - trainhist: `[1×3 struct]`
 - name: 'SOM 01-Dec-2021'
 - comp_names: `{768×1 cell}`
 - comp_norm: `{768×1 cell}`

K-means clustering with K=138



K-means clustering with Davies-Bouldin optimization for selecting best k



Clusters: CWE label Representation Bar chart representation (first 10 clusters)

