Understanding AVID

Author: Nathan Butters, Sven Cattell, Subho Majumdar
Date: 11-21-2022
Version: 0.1

This document is intended to be a guide to the taxonomy and database schemas of AI Vulnerability Database (AVID). As the first open-source, extensible knowledge base of AI failures, AVID aims to

- encompass coordinates of responsible ML such as security, ethics, and performance,
- build out a taxonomy of potential harms across these coordinates,
- house full-fidelity information (metadata, harm metrics, measurements, benchmarks, and mitigation techniques if any) on evaluation use cases of a harm (sub)category
- evaluate models and datasets are either open-source, or accessible through APIs.

AVID has two components: a **taxonomy** that provides a landing place of instances of AI system/model/dataset failures, and a **database** that actually stores information on such instances in a structured manner.

Taxonomy

The AVID taxonomy is intended to serve as a common foundation for data science/ML, product, and policy teams to manage potential risks at different stages of a ML workflow. In spirit, this taxonomy is analogous to MITRE ATT&CK for cybersecurity vulnerabilities, and MITRE ATLAS for adversarial attacks on ML systems.

At a high level, the AVID taxonomy consists of two *views*, intended to facilitate the work of two different user personas.

- **Effect view**: for the *auditor* persona that aims to assess risks for a ML system of components of it.
- **Lifecycle view**: for the *developer* persona that aims to build an end-to-end ML system while being cognizant of potential risks.

Note that based on case-specific needs, people involved with building a ML system may need to operate as either of the above personas.

Effect (SEP) view

The domains, categories, and subcategories in this view provide a ‘risk surface’ for the ML artifact being evaluated, may it be a dataset, model, or the whole system. This view contains three top-level domains:

- Security
- Ethics

- Performance

As described below, each domain is divided into a number of categories and subcategories, each of which is assigned a unique identifier.

Security

This domain is intended to codify the landscape of threats to a ML system.

ID	Name	Description
S0100	Software Vulnerability	Vulnerability in system around model—a traditional vulnerability
S0200	Supply Chain Compromise	Compromising development components of a ML model, e.g. data, model, hardware, and software stack.
S0201	Model Compromise	Infected model file
S0202	Software compromise	Upstream Dependency Compromise
S0300	Over-permissive API	Unintended information leakage through API
S0301	Information Leak	Cloud Model API leaks more information than it needs to
S0302	Excessive Queries	Cloud Model API isn't sufficiently rate limited
S0400	Model Bypass	Intentionally try to make a model perform poorly
S0401	Bad Features	The model uses features that are easily gamed by the attacker.
S0402	Insufficient Training Data	The bypass is not represented in the training data
S0403	Adversarial Example	Potential Cause: Over permissive API
S0500	Exfiltration	Directly or indirectly exfiltrate ML artifacts.
S0501	Model inversion	Reconstruct training data through strategic queries.
S0502	Model theft	Extract model functionality through strategic queries.
S0600	Data poisoning	Usage of poisoned data in the ML pipeline.
S0601	Ingest Poisoning	Attackers inject poisoned data into the ingest pipeline

—
NOTE
Notice that certain categories map directly to techniques codified in MITRE ATLAS. In future, we intend to cover the full landscape of attacks under the Security domain.
—

Ethics

This domain is intended to codify ethics-related, often unintentional failure modes, e.g. algorithmic bias, misinformation.

ID	Name	Description
E0100	Bias/Discrimination	Concerns of algorithms propagating societal bias.
E0101	Group fairness	Fairness towards specific groups of people.
E0102	Individual fairness	Fairness in treating similar individuals.
E0200	Explainability	Ability to explain decisions made by AI.
E0201	Global explanations	Explain overall functionality
E0202	Local explanations	Explain specific decisions
E0300	User actions	Perpetuating/causing/being affected by negative user actions
E0301	Toxicity	Users hostile towards other users
E0302	Polarization/ Exclusion	User behavior skewed in a significant direction
E0400	Misinformation	Perpetuating/causing the spread of falsehoods
E0401	Deliberative Misinformation	Generated by individuals., e.g. vaccine disinformation
E0402	Generative Misinformation	Generated algorithmically, e.g. Deep Fakes

Performance

This domain is intended to codify deficiencies such as privacy leakage or lack or robustness.

ID	Name	Description
P0100	Data issues	Problems arising due to faults in the data pipeline
P0101	Data drift	Input feature distribution has drifted
P0102	Concept drift	Output feature/label distribution has drifted
P0103	Data entanglement	Cases of spurious correlation and proxy features
P0104	Data quality issues	Missing or low-quality features in data
P0105	Feedback loops	Unaccounted for effects of an AI affecting future data collection
P0200	Robustness	Ability for the AI to perform as intended in diverse circumstances

P0201	Resilience/stability	Ability for outputs to not be affected by small change in inputs
P0202	OOD generalization	Test performance doesn't deteriorate on unseen data in training
P0203	Scaling	Training and inference can scale to high data volumes
P0300	Privacy	Protect leakage of user information as required by rules and regulations
P0301	Anonymization	Protects through anonymizing user identity
P0302	Randomization	Protects by injecting noise in data, eg. differential privacy
P0303	Encryption	Protects through encrypting data accessed
P0400	Safety	Minimizing maximum downstream harms
P0401	Psychological Safety	Safety from unwanted digital content, e.g. NSFW
P0402	Physical safety	Safety from physical actions driven by a ML system
P0403	Socioeconomic safety	
P0404	Environmental safety	

Lifecycle view

The stages in this view represent high-level sequential steps of a typical ML workflow. Following the widely-used Cross-industry standard process for data mining (CRISP-DM) framework, we designate six stages in this view.

ID	**Stage**
L01	Business Understanding
L02	Data Understanding
L03	Data Preparation
L04	Model Development
L05	Evaluation
L06	Deployment

Figure 1 reconciles the two different views of the AVID taxonomy. We conceptually represent the potential space of risks in three dimensions, consisting of the risk domain—S, E, or P—a

specific vuln pertains to; the (sub)category within a chosen domain; and the development lifecycle stage of a vuln. The SEP and lifecycle views are simply two different sections of this three-dimensional space.

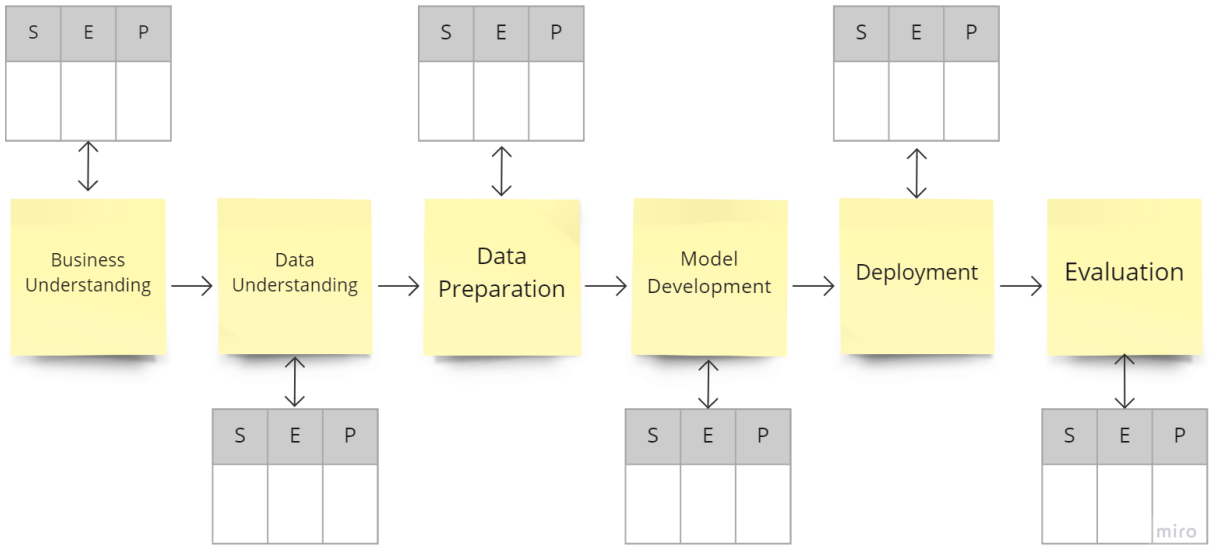

Figure 1. SEP and Lifecycle views of the AVID taxonomy represent different sections of the space of potential risks in an AI development workflow.

Database

The database component of AVID stores instantiations of AI risks—categorized using the above taxonomy—using two base data classes: **Vulnerability** and **Report**. A **vulnerability** (vuln) is a high-level evidence of an AI failure mode, in line with the NIST CVEs.These are linked to the taxonomy through multiple tags, denoting the AI risk domains (Security, Ethics, Performance) this vulnerability pertains to, (sub)categories under that domain, as well as AI lifecycle stages. A **report** is one example of a particular vulnerability occurring, and is potentially more granular and reproducible based on the references provided in that report.

As an example, the vulnerability AVID-2022-V001 is about gender bias in the large language model `bert-base-uncased`. This bias is measured through multiple reports, AVID-2022-R0001 and AVID-2022-R0002, which measure gender bias in two separate contexts, using different metrics and datasets, and record salient information and references on those measurements.

The above formulation is similar to how incidents and incident reports are structured in the AI Incident Database. See Figure 2 for a schematic representation of this structure.

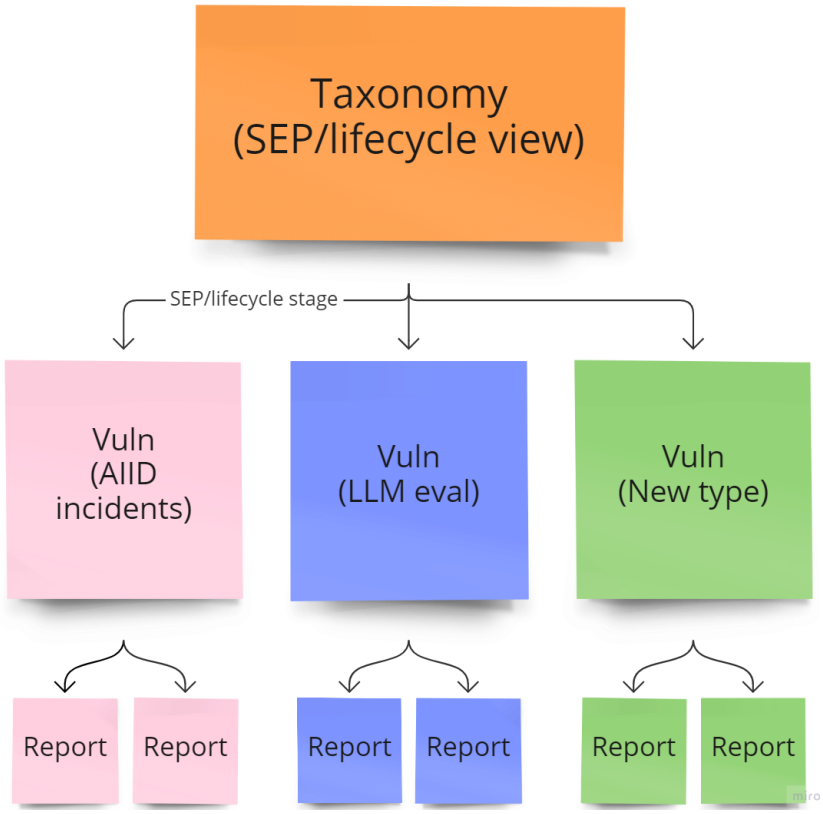

Figure 2. Schematic of the structure of the AVID taxonomy, vulns, and reports.

To account for diverse levels of details that different groups of AI risk examples can entail, we designate a class for each vulnerability and report. Each such vuln/report class extends the respective base class to a slightly different structure that enables storage of information at different granularities as required. For example, we currently support two vuln/report classes: evaluations of large language models (LLM Evaluation) and incidents from the AI Incident Database (AIID Incident). Both have the same set of vuln fields but slightly different sets of values to be filled in under references and tags.

Below we describe in detail the schemas of a vuln and a report, specifying types and explanations to possible values of each field and subfield. We use the following keys to represent this structure.

Keys

Submission Field - Something filled in as part of the form
Internal Field - Filled in by the AVID team
Inferred Field - Fields inferred from either Internal or Submission Fields or generated automatically

- *Possible Value*
  possible values are listed within this document they will be underneath the field and listed as bullet points. Any that require further explanation will be provided inline.

Report

report_id

Generated as AVID for the database ID,followed by the year
and sequential report number. (e.g. AVID-2022-R0001).

metadata

class

- *LLM Evaluation*
  An evaluation of a language model
- *AIID Incident*
  A report associated with an incident in the AI Incident Database
- *Other*
  Any report not belonging to the other two classes. This will lead to the creation of additional classes over time.

type

- *Incident*
  a single report on a model from a non-quantitative source (e.g. New York Times article, Twitter Post, LinkedIn, medium blog)
- *Advisory*
  an aggregation of several incidents that clearly identify a systemic vulnerability which requires additional consideration by society and industry.
- *Measurement*
  a quantitative analysis of the nature of a vulnerability, either through a direct investigation of the model or through an impact study of its effects.
- *Detection*
  a measurement that has been determined to be a significant deviation from ‘normal’, based on either a static threshold or a statistical significance test.

taxonomy_version

Version of the taxonomy the report was created in.
Auto-updated to the most recent.

submission

submitter_name

Name or handle of the person submitting the report.
If you choose to submit anonymously, fill in this field with *Anonymous*.

submitter_org

(Optional) Name of the group or organization the submitter would like to affiliate with reporting the incident.

submission_event

(Optional) The category of the event where the work was done for the report.

- *Research*
- *Blog*
- *Hackathon*
- *Bug Bounty*

`date`

Date of the submission.

`description`

`name`

Descriptive name of the reported vulnerability.

`description`

Description of the vulnerability, in reasonable detail.

It is recommended to include reach, and perceived severity.

`vuln_metrics`

List containing information on each detection being reported.

`name`

Name of metric.

`features`

Dictionary of features involved in this detection.

e.g. measured and sensitive features for bias detection.

`detection`

Dictionary containing information on the specific detection.

`class`

- *Upper threshold*: metric value higher than a static threshold counts as a detection
- *Lower threshold*: metric value higher than a static threshold counts as a detection
- *Significance test*: determined significant by a statistical test

`name`

Name of detection technique, e.g. disparate impact, z test for means.

`references`

`type`

The nature of the reference as it relates to the report.

- *Source*
- *Model*
- *Dataset*
- *Paper*

- *Misc*

`name`

Name of the specific reference.

`source`

- *GitHub*
- *Hugging Face*
- *Other* - an open text field

`url`

The publicly available URL where the reference can be accessed.

`tags`

`avid`

`vuln_id`

IDs of vulns associated with this report.
Note: some reports may highlight an intersection of several vulnerabilities.

`risk_domain`

SEP domain(s) relevant to the report.

- *Security*
- *Ethics*
- *Performance*

`sep_view`

List containing all relevant entries within the SEP hierarchy.

`id`

ID for the specific entry within the SEP hierarchy.

`name`

Human-readable name for the entry within the SEP hierarchy.

`lifecycle_view`

List containing all relevant ML lifecycle stages—per CRISP-DM.

`id`

Taxonomy ID for lifecycle view.

`name`

Human-readable name for the lifecycle stage.

`hf` (when `metadata.class` = *LLM Evaluation*)

`type`

Area of the Hugging Face platform.

- *space*
- *model*
- *dataset*

`lang`

The language of the artifact.
Choose from one or more of the ~20 spoken languages.

`name`

Name of the specific artifact,
e.g. `bert-base-uncased` as a model or `glue` as a dataset

Vulnerability

vuln_id

Generated as AVID for the database ID,followed by the year and sequential vuln number. (e.g. AVID-2022-V001).

metadata

class

- *LLM Evaluation*
  An evaluation of a language model
- *AIID Incident*
  A report associated with an incident in the AI Incident Database
- *Other*
  Any report not belonging to the other two classes. This will lead to the creation of additional classes over time.

taxonomy_version

Version of the taxonomy the report was created in.
Auto-updated to the most recent.

description

name

Descriptive name of the reported vulnerability.

description

Description of the vulnerability, in reasonable detail.
It is recommended to include reach, and perceived severity.

reports

List of reports associated with this vuln. Each list element contains the following.

report_id

Inferred from reports tagged with this vuln.

class

Inferred from reports tagged with this vulnerability. (See class in "metadata" of reports for the available options.)

name

Inferred from the reports tagged with this vulnerability.

references

References for this vuln. Intended to be of lower granularity than references linked in reports under a vuln.

`type`

The nature of the reference as it relates to the report.

- *Source*
- *Model*
- *Dataset*
- *Paper*
- *Misc*

`name`

Name of the specific reference.

`source`

- *GitHub*
- *Hugging Face*
- *Other* - an open text field

`url`

Publicly available URL where the reference can be accessed.

`tags`

`avid`

`vuln_id`

IDs of vulns associated with this report.

Note: some reports may highlight an intersection of several vulnerabilities.

`risk_domain`

SEP domain(s) relevant to the report.

- *Security*
- *Ethics*
- *Performance*

`sep_view`

List containing all relevant entries within the SEP hierarchy.

`id`

ID for the specific entry within the SEP hierarchy.

`name`

Human-readable name for the entry within the SEP hierarchy.

`lifecycle_view`

List containing all relevant ML lifecycle stages—per CRISP-DM.

`id`

Taxonomy ID for lifecycle view.

`name`

Human-readable name for the lifecycle stage.

`hf` (when `metadata.class` = *LLM Evaluation*)

`type`

Area of the Hugging Face platform.

- *space*
- *model*
- *dataset*

`lang`
: The language of the artifact.
Choose from one or more of the ~20 spoken languages.

`name`
: Name of the specific artifact,
e.g. `bert-base-uncased` as a model or `glue` as a dataset

`aiid` (when `metadata.class` = *AIID Incident*)

`incident_id`
: Unique ID for the incident, as given by AIID.

`report_count`
: Number of AIID incident reports associated with this incident.

`incident_date`
: Date of incident creation,

`editors`
: Name of AIID editors who created this incident.

`named_entities`
: Named entities found in this incident.

`config`
: System/model/dataset configurations relevant to any potential query.

`deployer` (when `metadata.class` = *AIID Incident*)
: Entity that deployed the system a vuln is reporting on.

`application`
: Real world application areas a vuln relates to, e.g. self-driving cars.

`task`
: Technical task a vuln relates to, e.g. generative models.

`architecture` (when `metadata.class` = *LLM Evaluation*)
: Model architecture, e.g. Resnet.