Secure AI Framework

Our Approach to Protecting AI Training Data

Authors
Jason Novak, David Deutscher, Jeremy Wiesner, Ben Kamber, Niha Vempati, Yurii Sushko, Cindee Madison, Cindy Muya, Reiner Critides

Abstract

Google has over 25 years experience protecting data from inappropriate access and unauthorized use. In the era of AI, Google has extended these best practices in data protection to ensure that the right data is used the right way to train models. This paper presents a number of these best practices, describes how Google applies them in its systems, and describes how Google Cloud customers can use Google Cloud capabilities to implement these practices themselves.

Protecting data requires both technical controls to enable safe data use at scale, and governance processes to ensure that companies have visibility and control over how their data is used. This fundamentally requires: understanding data and ensuring it has sufficient metadata in the form of attributes, controlling the data and implementing policies to allow (or disallow) certain usage based on those attributes, transforming data to enable its usage in policy compliant ways, and human oversight and governance.

Protecting data in AI inherits these requirements and introduces new requirements to account for unique AI-specific risks including memorization/recitation and the costs of training foundational models. Meeting these new risks requires new capabilities including enhanced understanding of data and model lineage as well as an increased ability to control data usage through checks on data for policy compliance at the time a training job is configured before it is run.

This white paper offers an in-depth look at data protection best practices and Google’s data protection capabilities, and is one of a series of publications about Google's Secure AI Framework (SAIF). Building upon its secure development practices, Google has developed and deployed a number of capabilities to understand, control, and transform data in its infrastructure so that data is both protected and used appropriately. This involves robust annotation systems to represent metadata and enable granular understanding of data at both an item and dataset level, policy engines that evaluate machine readable policies on that data using the metadata attributes, and sensors to understand how data is flowing across Google’s systems and raise alerts when policy violations occur. Moreover, Google has developed de-identification and anonymization systems to transform data to make it policy compliant and safer to use for AI training.

Table of contents

Best Practices for Protecting AI Training Data

At an abstract level, AI models can be considered to consist of two parts: their architecture and the statistical correlations made by the model during training on data. For the model to be policy compliant, the training data used to build the model must itself be appropriate for use and policy compliant. As this paper discusses in the following sections, ensuring policy compliance implies that: 1) the data is well understood with standardized metadata in the form of attributes ("Understanding training data"); 2) the policy enforcement can be readily applied to exclude non-compliant data ("Control: Policy engines"); 3) data can be modified to reduce training risks ("Control: Transformers"); 4) the organization has observability into these processes to be certain what data is used to train AI ("Governance"). This paper discusses each of these points in detail and then describes Google's data protection capabilities in these areas.

Requirements

> AI data protection is similar to other data protection efforts.

AI data protection is similar to other data protection efforts. It requires the consideration of a number of factors including the nature of the organization, its location, how it has collected its data, the type of data the organization is using, its customers, its agreements with customers and partners, and its regulatory and legal requirements.

These corporate governance and regulatory requirements will impact multiple parts of a model's lifecycle including: the metadata that is necessary to apply to training data and report on; policies that are enforced on training data; and reporting regarding the training of a model.

AI data protection typically exists in the context of companies having already addressed data protection for other existing obligations and having developed technical capabilities to do so. An organization should leverage these existing technical capabilities by determining what opportunities exist to extend them and what limitations are present in them. This assessment itself serves as technical requirements for an organization's approach to AI training data protection, and where and how they implement understanding and control as described below.

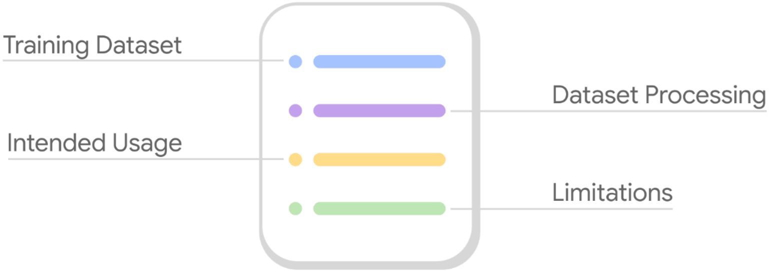
Given the speed at which AI is evolving, an organization may have to start with a best guess approach for building infrastructure and tooling that anticipates that specifics may change or come late in the process.

Understanding training data

To fully understand training data, including fine tuning and evaluation data, requires technical comprehension of how data moves through a company's infrastructure, a clear understanding of how that data is processed, and insight into the type of processing that is appropriate for that data. Moreover, these three types of knowledge need to be reduced into a singular machine-readable and human-understandable set of metadata values that can be applied to data, with rules that operate on that metadata to determine whether a given process can access that data.

The processing of data in an organization's infrastructure can broadly be described as a structured graph—a series of interconnected storage nodes with directional edges that represent processing. This graph represents the various stages of processing the data went through before it was used to train a model. However the graph alone is not sufficient to understand the data. The details of processing—what type of job was run on the data, what filtering was applied, and so on—are necessary to gain semantic understanding of the processed data and its properties; and some of these details have to be provided by humans.

Once an organization has the graph that shows how the training data was processed, as well as metadata that shows the results of that processing, the organization can address specific questions such as "what is the inventory of all models" and "what data was used to train which models," as well as identify the most advantageous places in the graph to implement controls to ensure compliant use of the data. Answering these questions with mechanical answers (which data was processed, for example) is necessary, but a complete answer requires the semantics of *why* data was processed and *the outcomes of* that processing.



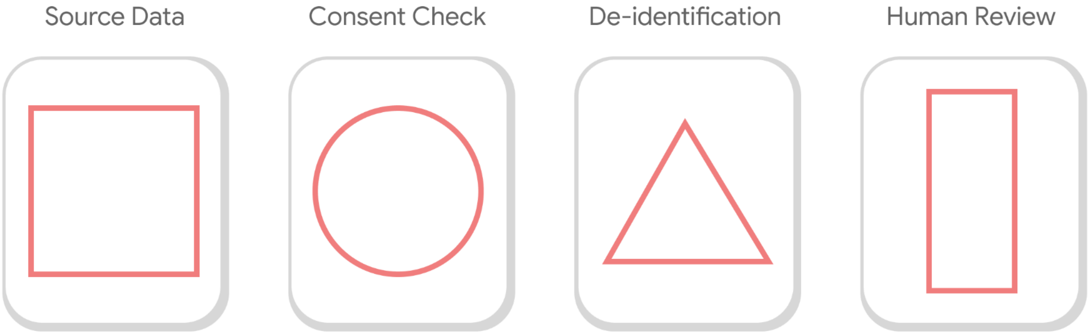
Model cards

Organizations' understanding of data is increasingly expected to be expressed in the form of model and data cards, which provide a structured report of what data was processed to train a core model. At a high level, a data card describes what types of data a model was trained on and what preprocessing occured; a model card describes the architecture and capabilities of the model. As a result, model and data cards are effective point in time views on the processing graph. Because the graph can change, model and data cards need to be frozen at a point in time to reflect the view of the graph at that point in time. This conception of cards combined with their structure makes them effective tools to use for both governance of models as well as for reporting to customers and regulators. In addition, the fact that model and data cards are views on the processing graph also implies that model and data cards naturally travel with the model insofar as an organization has complete lineage of a model, its copies, and fine tuning of those copies. If the lineage graph is broken—for example, if a model is copied in such a way that disassociates it from prior processing—then the model and data cards need to be manually copied and transmitted with the model as metadata of the model itself. This need to support both point in time views of the lineage graph as well as frozen copies of the model and data cards means that model storage needs to support linking a model to its predecessors as well as manual associations of models with model and data cards.

Lineage

Given the iterative nature of AI development—where a model may be derivative of predecessor models and data may be reused across training runs—understanding the relationship of data and models takes on additional importance for an organization. We call these relationships—which data is used to train which models, which models are derived from each other, and the interrelated processing—*lineage*. Having a complete view of lineage allows automated and manual governance of data and model usage, reporting that can be used to demonstrate what data was used to train which models, improved model training efficiency, and the ability to calculate aggregate training costs.

Lineage relies on building the structured graph discussed above to understand how data has been processed. There are two primary ways that lineage is developed. It can be collected automatically by sensor systems that observe reads and writes to storage and processing jobs, or manually by a human who declares the process they used to modify the data. These systems bring complementary perspectives: while automatic collection can provide a complete picture of the graph (that is sometimes overconnected as in the case of multi-tenant systems used across a wide variety of purposes), it does not have semantic understanding of *why* certain actions were taken to process data or *what* the goal of that data processing was. This semantic understanding often needs to be provided by humans as they understand *why* a job was run. This human understanding can be provided through a hybrid approach where observed reads/writes are aggregated and enriched by human declared lineage. This combination of lineage approaches also helps resolve overconnected graphs (that tend to be created through automated metadata propagation alone) by adding additional manual declarations of lineage as needed.



Transformations with lineage

Once this lineage is built, reports can be built from it that can address specific questions about the properties of a dataset, or what datasets were used to train a model. As discussed below, there are different levels of metadata granularity that an organization can collect and there are tradeoffs to collecting metadata at different levels. It may be faster for an organization to collect coarse-grained metadata; however, doing so inherently limits that organization's ability to provide detailed reporting. If the organization is later required to provide fine-grained reporting, they have to undertake an expensive retroactive metadata collection process. Given that collecting lineage is a time consuming process that is difficult to do post-facto, collecting metadata at a fine-grained level at the time of model training provides an organization with options in reporting, and helps future-proofs metadata collection.

Metadata

Metadata—or data that describes properties of data—can be used to apply automated policy enforcement, while human interpretation is needed to make data management easy during development and for governance. An organization's approach to metadata will depend on its data use and governance requirements. This is an extension of historical data governance practices to the needs of AI.

Data structure and metadata

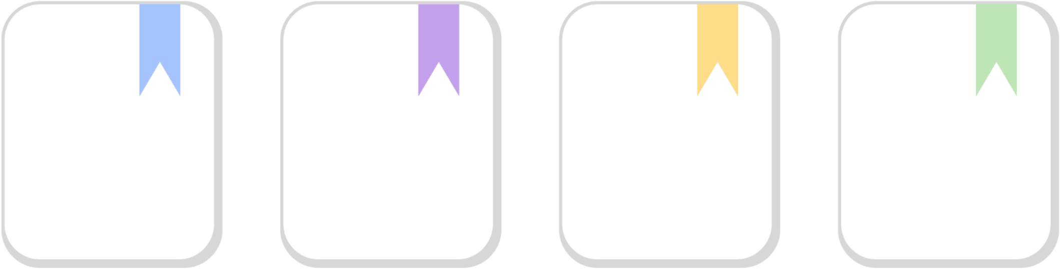
Data can be stored in a variety of formats with different types of storage: files on a hard drive, rows in a database, or even unstructured data like emails. Each of these storage solutions have implications about metadata, including the technical feasibility of collecting it, and the level of granularity it provides. The way that an organization stores its data is related to how it labels its data with attributes. Other considerations include how the data is used, and what policies the organization wants to enforce on its data. Moreover, applying metadata attributes to one level (an entire database, for example) does not rule out applying metadata attributes to other levels (such as a single row in the database). In fact, applying metadata attributes to multiple levels may be necessary to achieve the organization's goals. The cost of this flexibility might be additional checks or reconciliation logic to ensure that the metadata stays consistent across these levels.

Another part of an organization's metadata strategy is how it chooses to collect and store data. If a dataset is homogeneous—from a single source, stored in a single format, with uniform properties—metadata about the dataset can be recorded at the dataset level. For example, if a dataset contains data from a single product, the dataset can be labeled with the product name (or an identifier that can stay stable if the product's name changes). If the dataset is heterogeneous in sourcing and storage—that is, it comes from different products and each product records different information—metadata has to be stored about each record in the dataset alongside the record itself. Even in this case, there may be metadata that is desirable to capture at the dataset level—for example, the responsible party for governance inquiries or other operational needs like billing.

The lifecycle of metadata is similar to the lifecycle of the data itself. When data is acquired by an organization, the appropriate origination metadata should be recorded with the data, such as date acquired, source, authority, relevant consents, and licenses. As data is processed, its metadata should be updated to reflect that processing, including metadata from human-based reviews of the data. And, as data is used to train models, those models' metadata may take upon relevant attributes from the source data.

Different levels of granularity allow for different types of enforcement. Dataset level metadata allows an organization to check at the file level whether a data is valid for its intended use. Row level metadata allows for an organization to achieve more use of its data as it enables heterogeneous datasets to be used in policy compliant ways. Finally, applying metadata at the field level allows for organizations to apply field level policies to data—for example, selective redaction of records or masking of fields when displayed in various tools.

There are advantages and disadvantages to each level of granularity, and an organization has to consider these in its metadata strategy. File level metadata is more efficient—it is generally faster to label at the file level, just a single check can determine whether the data is valid for the intended use, and it requires less storage space as only one copy has to be stored for all records in the file. However, the coarseness that gives file level metadata its efficiency may also make the data less useful if the data is heterogeneous—for example, a single record in a file containing location would require the entire file to be treated as location data even if no other record contained location. As a result, row level data allows for an organization to achieve more use of its data at the cost of increased labeling time, storage, processing time, and number of policy evaluations computed. Each row has to be labeled, and each row's metadata has to be stored and evaluated for policy compliance, but then individually compliant rows can be used for training.



Attributes on datasets

Different levels of granularity of metadata and policy enforcement are appropriate at different points in model development. Broadly, in model pretraining there are fewer researchers operating on fewer—but larger—datasets than in model finetuning where there are more researchers working on more datasets of different properties. Therefore, during pretraining, dataset level attributes and policy enforcement is possible and effective without introducing significant overhead. In contrast, during finetuning, when a researcher may be mixing examples from diverse datasets and or hand curating data in experiments, it is important to support row level enforcement so that policy can be consistently applied. In addition, this means that row level metadata needs to be propagated throughout the finetuning process and across tools.

Taxonomy

Effectively leveraging metadata to protect AI data requires a common taxonomy of metadata definitions and values to ensure that data can be understood consistently over time, between systems, and across organizational boundaries. If an organization does not have a common taxonomy, data will end up having the wrong metadata applied to it, which could lead to it having the wrong protections—or no protections at all—applied to it.

It's essential for an organization to create a consistent and common taxonomy for their metadata. For example, a location can be many things: it can be a user's current whereabouts, or it can be the destination the user is searching for, or it can be a point of interest a user is browsing. A location also exists at different granularities: precise (like a street address) or coarse (like a postal code). Location information can also originate from a variety of sources like a user's device, an IP address, or a listing provided by a store. Each of these dimensions require common terms that an organization can use and reuse over time to determine what a location datapoint is and is not, which in turn can be used to determine what policies should be applied.

Metadata's usefulness also makes the metadata itself sensitive and necessitates a multilayered approach to metadata security. Fine grained information about what precise files were used to train a model is appropriate for model developers, but may not be appropriate for an entire organization to know; in many cases, users and finetuners of the model would need only summarized metadata describing the training set. For example, model developers need to know which videos were used to train a model, while a model user or fine tuner may need to know only that a model was trained on videos.

Moreover the security of the metadata should not be the same as the security of the data. Metadata's usefulness (including for compliance purposes) also makes the metadata itself sensitive (from a strategic business or privacy perspective). Broadly, by default, it is desirable to make metadata world-readable to enable broad reuse of data while keeping access to the underlying data based on business needs and tightly controlled. This approach empowers model developers to more readily discover datasets that can be useful for training—for example, a model developer can more readily find a location-based data set if location data is labeled and those labels discoverable, even if the data is restricted.

Each organization should develop its own taxonomy based on its data, products, and regulatory requirements. Determining the proper granularity and span of an organization's data taxonomy is crucial. When a taxonomy is too granular, it can result in an explosion of values. This can cause confusion for individuals collecting and labeling the data and make it difficult for individuals trying to understand data by wading through multiple attributes. On the other hand, too coarse a taxonomy can prevent the reuse of attributes across policies.

Consistent use of attributes is also important. For example, in the case of location, if a nine digit US Postal Code were being stored, there are equally valid reasons for it to have metadata describing it as "POSTAL CODE", "ZIP", "ZIP+4", or "COARSE". The *appropriate* metadata attributes for an organization depend on the organization itself, but there always needs to be a balance between describing data, and effectively using (and reusing) attributes.

Development of a common taxonomy is challenging as it involves both technical and human demands. It should be assumed that taxonomies will change over time as individuals develop experience using them and identify gaps and new attributes needed. It is not atypical for the meaning of an attribute to evolve over time as it is applied to new datasets and for new regulations. Building the taxonomy so that it can be versioned allows for an organization to adjust its granularity as needed and prevents the perfect from being the enemy of the good.

Manual versus automated application

Metadata attributes can be applied manually (by humans) or through automation (by computers) and both have their place in data protection. There are also drawbacks for both, and therefore a hybrid approach is needed.

Humans have context regarding data that a machine may not and, as a result, they can be more descriptive in their metadata—while a computer can tell you *which* file was used, a human can explain *why* that file was used to train a model. However, human application of metadata is manual, fixed at a point in time, does not scale, and prone to error—for example, applying an incorrect attribute (applying "coarse" on street address level location data when they should apply "granular"), or applying the correct attributes incorrectly (mistyping an attribute as "ZIP4" instead of "ZIP+4").

Key to effective human labelling is it being done *in the same context and at the same time* as the data collection code being written. It is much easier for a human to add context at the time that context is in mind rather than doing so post-facto. Computers can analyze data and provide metadata for it at scale but are limited to specific heuristics—for example, that data matches a given pattern or which processes have touched a given file. In practice, these limitations mean that manual metadata should be seen as very high signal low volume, and used to augment automatically collected metadata that, without context, may be low signal high volume. Notably, these approaches can be hybridized—for example, ML inferences can be made automatically about what metadata should be applied to data, systems can surface those suggestions to humans in context, and humans can then approve or reject that metadata.

In addition to observing which files are used by which processes, automated metadata can also refer to how the data is processed. For example, if a process anonymizes data, that process should automatically apply attributes to the data indicating that the data is anonymous, how it was anonymized including software version and configuration, and the date at which the anonymization occurred. Applying this metadata automatically as part of the process is less error prone than a human manually applying this information as a post-processing step.

Automated metadata also helps address the gap between conceptual descriptions of data and their physical representation. It is typical for a researcher to have multiple copies of a dataset with different filtering and preparation applied. For example, if a researcher has an economics dataset, they may have multiple copies of it filtered in different ways: by year, by century, by country, by region. Saying "the economics dataset" does not distinguish between these different copies, each of which is a different subset of the data with its own physical storage on disk. Automatically applied metadata can make it clear that there are different copies of the data with different filters applied over time and can help practitioners be precise as to which copy of the dataset they refer to when they refer to "the economics dataset".

Finally, metadata should be understood as dynamic rather than static. For example, if metadata is meant to reflect a user's current preferences, it may be preferable to store an identifier for the user and look up the current state of the preference when processing the data, rather than storing an attribute of that preference. For posterity and retrospective analysis, it may be desirable to store metadata as a versioned attribute that can track historical values or to store historical snapshots of the graph.

Propagation

Metadata attributes should propagate with data to ensure consistent enforcement across time and systems. Labeled data in a storage system can be used to enforce that system's policy on reads and writes. When data *leaves* the source storage system, the attributes present on that data should travel with the data through processing systems and ultimately be written into destination storage systems. This ensures that provenance or other properties of data is not lost as it undergoes processing. Moreover, processing systems should add or remove attributes as appropriate given the processing they undertake. For example data may be marked “user data” at its source, but if it is read and processed by an anonymization system, that system should remove the user data attribute before writing the now anonymized data to the destination storage system.

Control

There are multiple types of controls and control implementations that can be applied to AI data to ensure that it is being used in a policy compliant way. For AI data protection, four primary controls are: policy engines, transformers, governance processes, and lineage. Policy engines, transformers, and governance processes are largely extensions of existing practices and concepts, while lineage takes on new importance and depth in the AI space due to the cost of training models.

Policy engines

A policy engine is a control that determines whether data or a dataset is allowed to be used for training based on the understanding the system has about the data and policy presented to the engine. To effectively enforce policy on AI training data, it is important to both enforce policy at the dataset level at training job configuration time—that is, determine the compliance of the entire dataset before processing begins—and to enforce policy at an object level at processing time—that is, determine on an object-by-object basis whether that object can be used for training based on the semantic properties of its metadata.

Training a foundational model is extremely costly, and because of the inherently stochastic nature of training, finetuning, and retraining means that the results are not always reproducible. Given such, understanding the properties of the data prior to using it for training is preferable to filtering it in real time. Real-time filtering can result in inefficient use of resources as a data set may be filtered out to be useful for training purposes. Understanding the amount of data that will be filtered out before training begins and allowing for an organization to evaluate whether to run training at all on the remainder allows for the efficient allocation of resources.

An organization should consider if its policy engines should “fail open” (allow data use) or “fail closed” (block data use) if metadata is missing from a dataset or object. This decision may be different for different types of metadata. It may be appropriate for an organization to assume that if certain metadata fields are missing that a default value can be assumed and the data can fail open and be processed. However, there are some metadata fields (for example, metadata for regulatory compliance) where having no metadata should cause the policy to fail closed, blocking the training job.

Transformers

Transformer controls modify—or transform—data so that it may be used to train models in policy compliant ways. These differ from policy engines as they do not return verdicts on the use of data but rather modify the data to make it policy compliant. These modifications include de-identification and anonymization.

De-identification is a data processing technique that removes personal identifiers from data.

Anonymization is the process of rendering personal data unidentifiable, so that no one associated with the data can be identified either directly or indirectly through auxiliary information. At a technical level, anonymizing data can be achieved through techniques such as differential privacy and k-anonymity. Care must be taken when anonymizing to avoid potential pitfalls such as re-identification.

When applied to training data before or during model training, anonymization and de-identification prevent models from memorizing (and eventually reciting) identifiable information linkable to individuals who contributed the data.

Governance

Governance processes generally refer to human-driven controls. Similar to the discussion of manual vs. automated application of metadata, process controls provide more nuanced, adaptable oversight in fast-evolving, complex domains. AI products continue to evolve, and with the use of new techniques, new data protection implications appear. Process controls enable us to recognize emerging risks and patterns. This, in turn, creates new opportunities to apply technical controls that provide a higher consistency and volume of data protection.

Governance processes have three major components: guidance, reviews, and observability. Guidance refers to written and approved documentation directing the use, development, and deployment of ML data, models, and applications, and includes:

- Formal company-wide requirements in Policies and Standards, reviewed and commonly approved by privacy, security, and legal subject matter experts.

- Guidelines that specify recommended ways to adhere to requirements, as well as best practices for data handling.

Reviews are assessments and approvals made at various junctures in the software development lifecycle (SDLC). These range from advisory (non-blocking) reviews and consultations, to formal, gating (blocking) reviews that are required to proceed to the next step in the SDLC or to launch. Reviews may be conducted by governance professionals (such as privacy, security, legal, and trust and safety), or by product or company leadership.

In the context of governance, observability refers to the organization's ability to see and validate that its actions and outcomes match its intent, and that all of the controls developed and applied are functioning as expected and protecting data in practice. Observability combines some of the capabilities for understanding data (metadata, lineage, and so on) with data about the controls, decisions, and actions taken. That information is aggregated and shared with key stakeholders who are accountable for data protection governance. Governance processes generally mature over time as an organization gains experience and familiarity with AI development and identifies opportunities to automate or refine its tooling.

Initially, organizations drive governance through policies and guidance that anticipate specific risks and requirements based upon subject matter experts' prior work in the field and risk forecasting. This governance broadly is manual as it requires consideration of new architectures, patterns, and risks. This work provides immediate coverage and oversight of high risk areas as well as early, manual telemetry on AI risks and practices, which can be used to develop automated controls as the organization's understanding of risk matures.

As an organization develops a better understanding and observability of risks and mitigations in the AI space, those learnings can be provided to infrastructure teams to inform the creation of technical controls. Reviews can be streamlined, allowing some manual checks to be automated using source-of-truth understanding and lineage that is manifested in artifacts such as data cards and model cards. When guidelines are supported by automated data policy enforcement tools or systems, and the adoption of that enforcement is part of most keystone systems, those guidelines can be promoted to formal written policy/requirements, further discouraging the use of bespoke systems that don't have protected-by-default capabilities.

Lineage

In addition to providing *understanding* of data, lineage can also serve as a *control* on data. Before training begins, the lineage of datasets can be used to determine whether datasets have undergone all necessary policy checks and filtering for use. Moreover, the lineage of a model can be used to determine whether it is fully approved and policy compliant for use in a given application; this is similar conceptually to a Software Bill of Materials. Lineage is needed in addition to other controls for preparing data for training as it is a *post-facto* control: it tells you what has happened. In the AI space, given the high costs of training, other controls are needed to ensure that *only* compliant data is used for training. Lineage can then be used as a verification step.

Google's capabilities

> Different data protection capabilities may be deployed in each step depending on existing infrastructure, regulatory needs, and other requirements.

Google's approach to protecting AI data is to extend our existing data protection capabilities—the steps we take to understand, control, transform, and govern data—and apply them as appropriate to steps in the AI lifecycle. These steps include data sourcing, training and tuning, model serving, and input/output handling. Different data protection capabilities may be deployed in each step depending on existing infrastructure, regulatory needs, and other requirements. Google thinks about these requirements and data protection holistically and, as such, applies appropriate data protections at the appropriate places in the end-to-end path training data takes in training a model.

Understanding training data

Google has developed robust systems for understanding data and expanded them for AI. These include a unified, interoperable taxonomy that allows for engineers to express both what data is and how it may be processed. Google has also developed systems that allow for the labeling of data at the field, object, and dataset level with appropriate attributes.

Metadata

Google's systems for understanding data were extended for AI by adding definitions for AI concepts and policies, including distinguishing between evaluation and training data. These terms were defined in both row and dataset level vocabularies. For example, if the entire dataset is evaluation data, the dataset level can be marked as such, but, if a dataset is heterogeneous and consists of both evaluation and training data, each row can be indicated appropriately.

Understanding of data is achieved through a combination of manual labeling (specifying attributes) and automated labeling (approving ML suggested attributes). For example: when a Google engineer defines a protobuf, they can add attributes as extensions to the protobuf's fields for purposes of providing a semantic description of what the data is. Similarly, when a Spanner database schema is declared, attributes can be applied as options in the schema to declare properties such as data type, retention, and source. In addition, attributes are automatically suggested. For example, when an engineer submits code, an ML-powered presubmit check can automatically suggest relevant labels, which the engineer can review and apply before submitting the code.

Sensors

There are a variety of ways that Google engineers can interact with data and transform it for use in AI models. Rather than have developers manually attest to every interaction they have with data, Google has developed sensors to automatically observe the transformations that data undergoes as it is read or written by different processes.

Event sensors

Processing and storage systems integrate with Google's automated event sensor library to send information regarding read and write events to centralized logging storage. A recording of a read or write events can be thought of as a tuple that includes: the processing job identifier, the identifier of the data storage being accessed, the action (read, write, delete), and select privacy and security metadata. This is high fidelity, high volume data that requires mapping between identifiers and semantic meanings, since a job identifier or storage identifier is not inherently meaningful (for instance, a job id could just be a numeric indicator). This mapping adds additional attributes to data even as it is read, written, and transformed throughout our systems. For example, if a specific job is an anonymization job and the output data is anonymous, the output data has metadata automatically added to indicate that it is anonymous.

Semantic sensors

In addition to the fine-grained processing data collected by event sensors, Google also collects semantic metadata and enriches its understanding of data with it. AI data processing applications incorporate a sensor logging library to provide semantic understanding to processing activities. For example, an anonymization process may call the manual sensor library to record to Google's central logging systems that "anonymization was run on dataset 123".

To fully map declared semantic data usage and processing in AI systems, this data must be combined with information from automated sensors.

Control

Policy engines

Google has developed policy engines that integrate with its storage, processing, and serving systems. When data is used (read, written, sent to a different system, or transformed), the policy engine evaluates a tuple composed of the action, metadata attributes, and context. This tuple is checked against all relevant policies (often hundreds or more constraints) to determine whether to permit the operation. Similar to the capabilities for understanding, Google has policy engines that operate on the dataset level or row level for purposes of enabling fine-grained policy control as needed.

These policy engines provide a comprehensive policy enforcement solution that provides both config-time checks and runtime filtering to ensure compliance in data processing pipelines. This combined approach ensures that data usage adheres to predefined policies throughout the entire data lifecycle, from pipeline configuration to runtime execution.

To protect AI data, Google extended these policy engines to operate on systems that store and process training data to enforce AI specific policies.

Config-time checks

Config time checks operate on processing pipelines to determine whether that pipeline is compliant by evaluating the datasets, their metadata attributes, and the processing to occur. The mechanism does so through an examination of the complete graph of training and the attributes present on datasets in the graph. In turn, this requires training systems to declare all of its inputs and outputs prior to processing.

Runtime filtering

Runtime filtering can be used to prevent individual data objects from being used to train models. It does this by evaluating the metadata of an object (the type of data, the purposes for which it was collected, the transformations applied to it and so on) against a policy (for example, whether consent is required for the purpose of training a model) and filters out data whose metadata indicates its use is not policy compliant in this context, permitting only compliant data to be trained upon.

Lineage enforcement

Lineage enforcement provides review flows for ML input artifacts and enforcement of policies before a given dataset can be used for training. This enforcement is based on reviewing the lineage of data and determining if the dataset has been either approved or transformed as necessary for the intended use.

Transform

Google has developed capabilities to reduce or remove the risk of using certain data in AI training, including de-identification and anonymization technologies. .

Data de-identification

Cloud DLP is a tool for filtering out personal identifiers and sensitive information from text data including transcribed audio, alt text in images, and OCR'ed images. Using regular expressions, semantic context, and dictionary-based lookups, Cloud DLP can detect more than 190 different types of information and allows clients to redact, replace, or mask its findings. Running Cloud DLP on text-based training data substantially reduces the risk of models memorizing sensitive, personally identifiable information.

Notably, not all personally identifiable information is risky to memorize. "George Washington" is personally identifiable but rarely risky to have memorized because he is a historic, public figure. As a result, some filters may not be appropriate for the goals of a model. Cloud DLP usage may also need to be supplemented with human review to remove personally identifiable information from a dataset. The risk of personally identifiable information being memorized depends on a number of factors including: the source of the data that contains this information, the goals of the model, and the intended deployment of the model.

Anonymization

Google has developed multiple methods and technologies to support anonymization. In the context of AI, anonymization can be applied in two general ways.

The first approach is to mitigate any memorization risks during model training. An example of how this can be achieved is through use of Tensor Flow Privacy and JAX privacy, Google's machine learning libraries that enable the training of anonymized models using differentially-private training protocols.

The second approach is to use anonymous data itself for model training. For example, anonymous training data can be created with Google's differential privacy libraries or via BigQuery's differential-privacy integration. Another promising and rapidly evolving area is differentially private synthetic data generation, which may provide greater usability for developers because such data is designed to resemble real data by shape and properties, while providing strong anonymization guarantees.

Governance

Google's data protection capabilities are supplemented by governance experts who leverage Google's capabilities to provide guidance and to conduct reviews at scale for product teams.

Guidance

Google has dedicated subject matter experts who define formal policies, and who participate in standards bodies to continually align with and promote data protection best practices in the industry. Google also creates training, documentation, and advocacy efforts to ensure that its workforce understands, internalizes, and adheres to these policies and standards.

Google also proactively participates in research and development of innovative data protection solutions, including solutions that respond to emerging risks and trends in the development and use of AI. These activities become the source for establishing new data protection guidance and best practices at the company.

Reviews

Google requires every user-facing product to undergo a review process prior to launch, conducted by subject matter experts in privacy, security, safety, and compliance risks, who also deeply understand each of Google's products and their implications for data protection. Google also maintains teams of experts to provide additional consultations and exercise depth of judgement over complex domains such as AI.

Google has developed robust, flexible platforms and tools to support tamper-proof, optimized, and observable reviews, approvals, and related workflows. These platforms integrate with Google's other data protection capabilities to enable informed reviews with enforceable outcomes.

These platforms include tooling to track approvals for launches through an interface that is highly customizable, letting product teams design and manage their launch processes efficiently and consistently. This tooling generates governance artifacts that are auto-populated with relevant metadata obtained from integrated systems, and provides triage tools to help connect product developers with the right set of reviewers and experts at critical moments.

Conclusion

The approaches outlined in this white paper are intended to guide organizations on best practices for protecting their data in AI by extending existing data protection practices to AI systems.

As we've discussed in this paper, protecting AI training data starts with having a complete understanding of what that data is, where it comes from, and how it has been modified. That understanding should be recorded as metadata with a consistent and clearly understood taxonomy. This metadata can be added to the training automatically through compute processes, or manually through human curation. When data is understood and described clearly by its metadata, its use in training can be controlled through automated policy enforcement modeled on your data needs and governance requirements.