Google

Continuous Evaluation

Using CI Techniques For Experimentation At Scale

Nilesh Jagnik

Agenda

01 Background

02 Search Evaluations

03 CI Triggers

04 Challenges

05 Q&A

Background

Evaluation generates insights about the business impact of product changes.

Goal of Evaluation: Business Metrics

Quality

Study the quality of results generated by Google Search and other products.

Impact

Know how many users will be affected by changes proposed to Google products.

Information Satisfaction

Identify headroom for product growth. Study win/loss patterns for hill climbing in model development.

Metrics Usage

TUNING

01

Iterate on changes before requesting launch approvals.

LAUNCHES

02

Predict user impact of product changes to justify launch of new features.

GUARDRAILS

03

Detect unexpected behavior of product changes.

Platform Users

Feature Developers

Engineers using evaluation metrics for feature launches.

Metric Developers

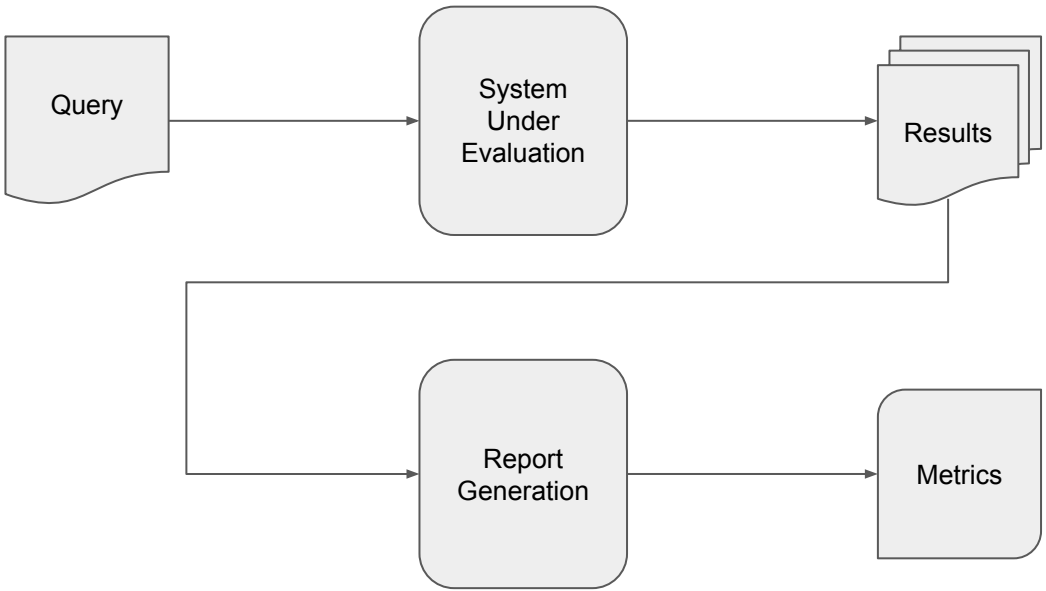
Data Scientists and Engineers developing new metrics based on user/market research.

Model Developers

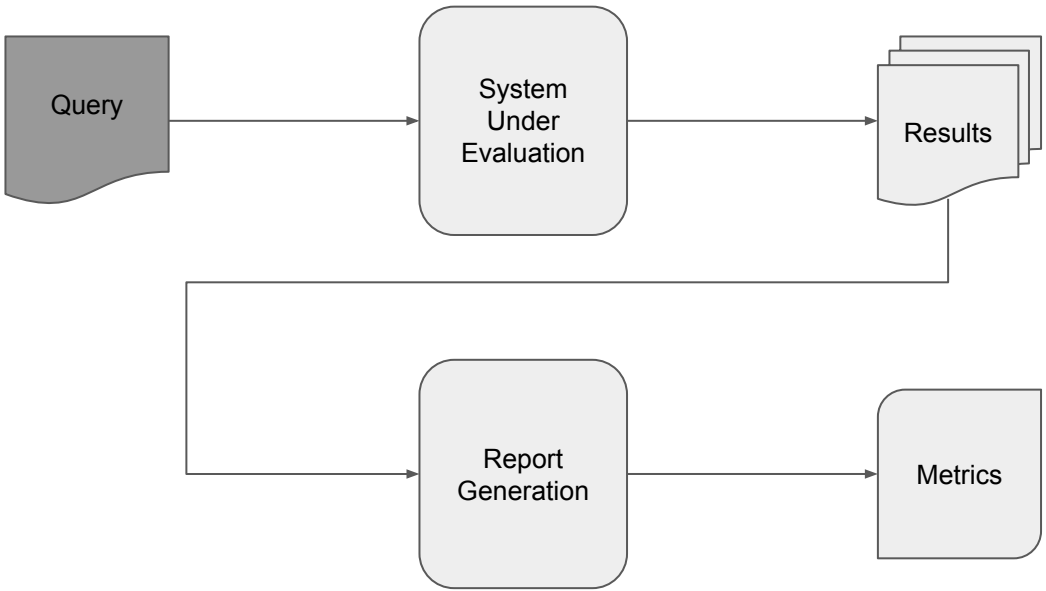
Data Scientists and Engineers developing GenAI models.

Search Evaluations

How Evaluations Work



How Evaluations Work: Queries



Queries

Source

Generated from filtered and anonymized production logs or synthetically.

Curation

Standardized query sets for various locales, languages, etc.

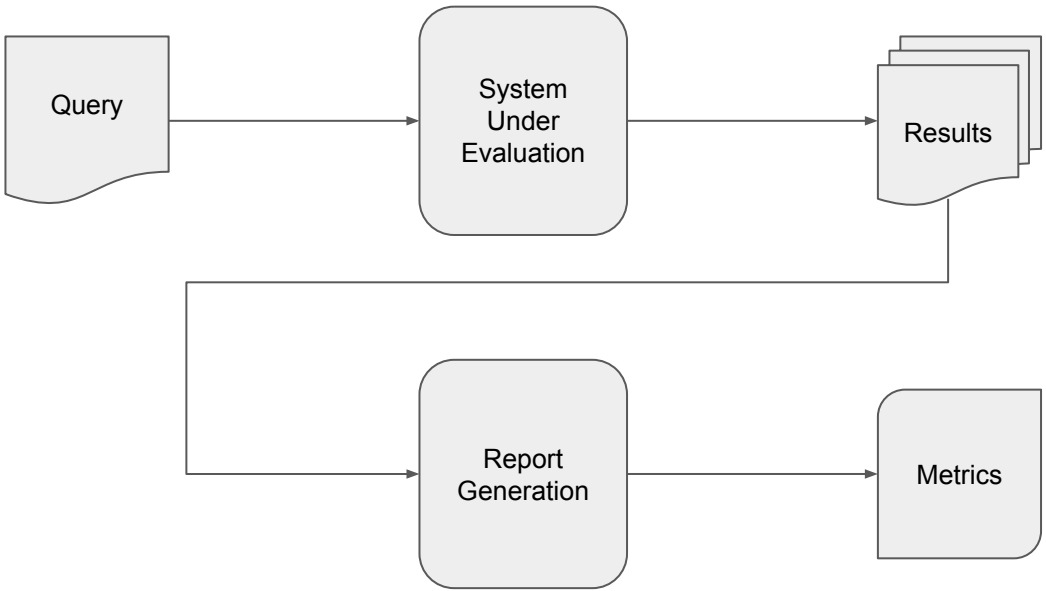
Sampling

Queries can be randomly sampled from curated pools.

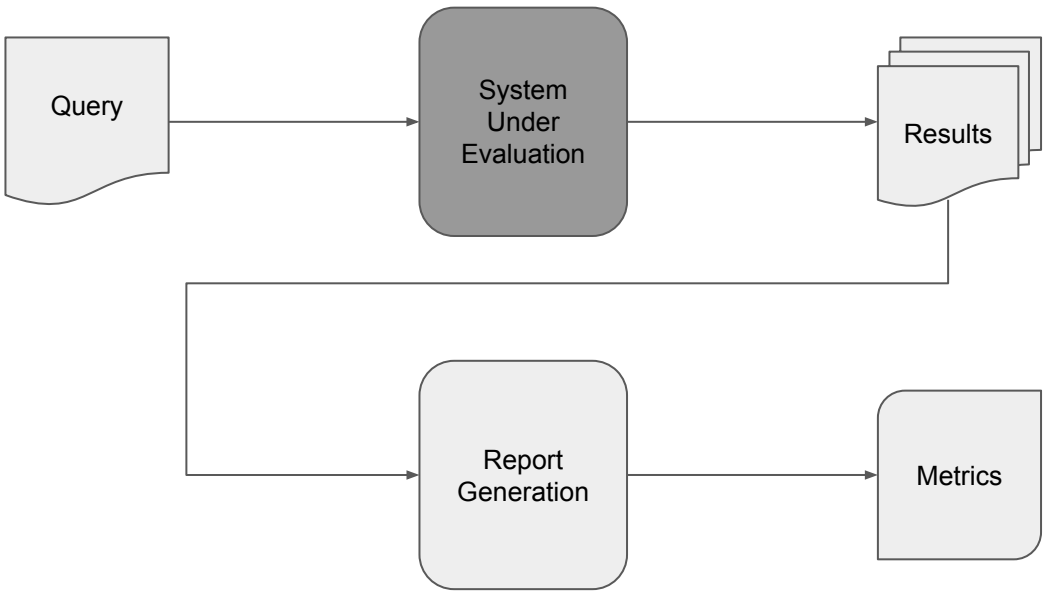
Size

Evaluations can run anywhere from a few thousand to hundreds of thousands of queries.

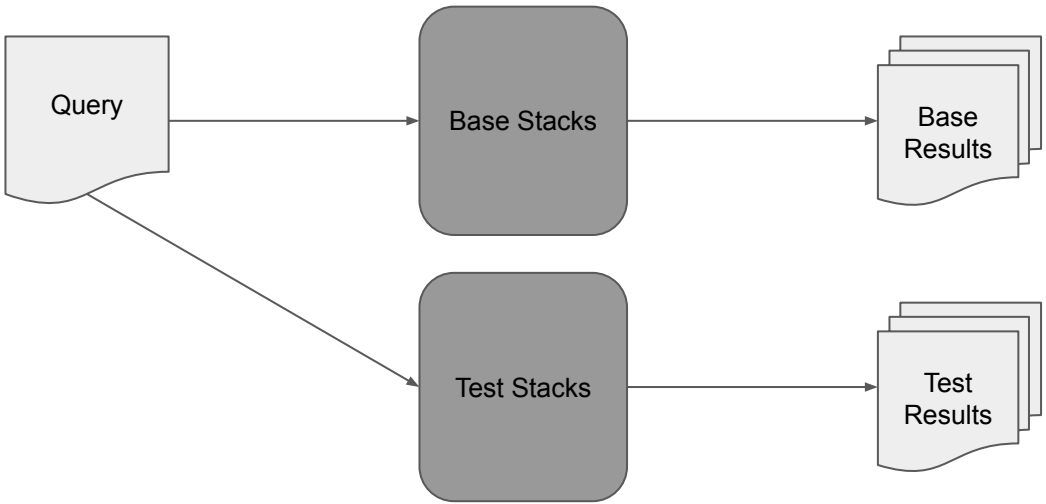
How Evaluations Work



How Evaluations Work: System Under Evaluation



System Under Evaluation: SxS Evaluations



Only servers with code changes are freshly created. Traffic flows through production servers (serving live users) for unchanged servers.

System Under Evaluation: Build Parameters

binaries

- Which servers to build. Other servers are re-used from production.

base_cl

- Optional parameter specifying how to build base stacks.

base_cl_strategy

- prefer_dev_base_CL: Infer base_cl using patch CLs.
- green_CL_only: Find a green CL near HEAD (based on CI tools).

base_patch_cls

- Changes specific to base stacks.

experiment_patch_cls

- Changes specific to test sides.

System Under Evaluation: Customizations

SxS Testing

Evaluations can be single or dual sided.

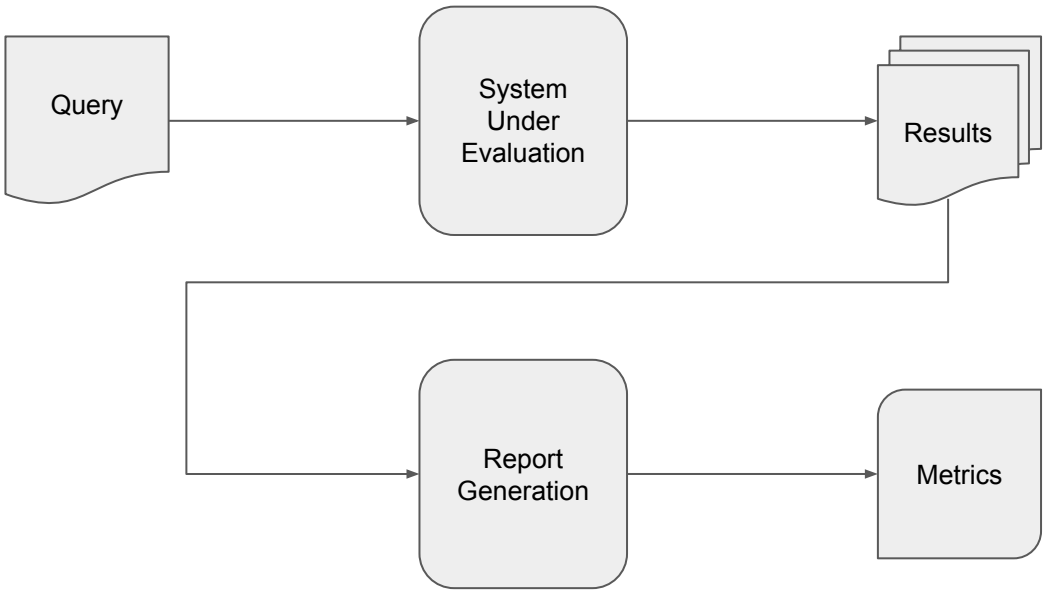
Url Customizations

Customize requests sent to each side.

Sampling

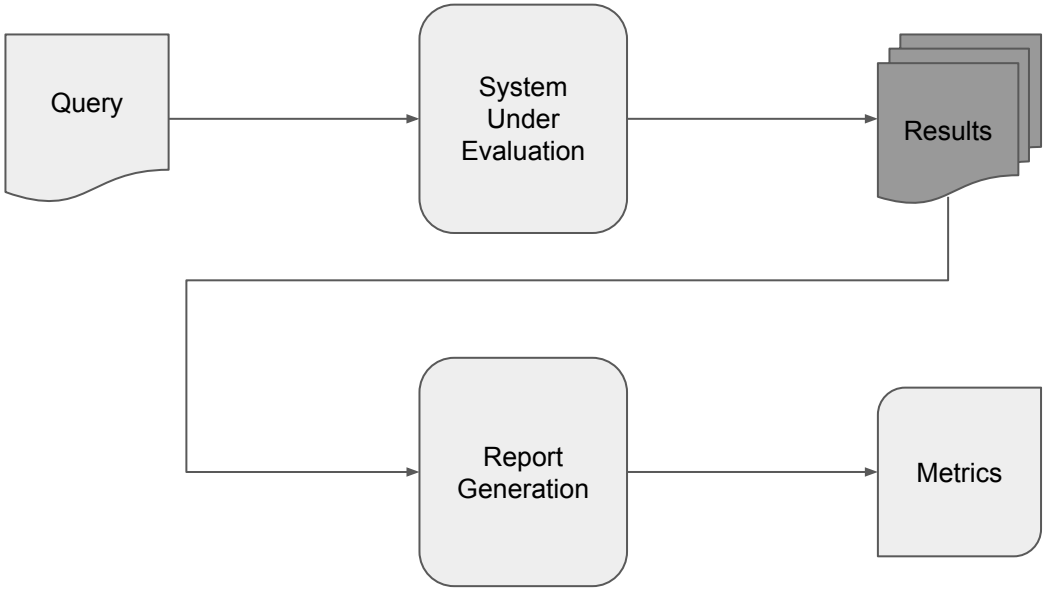
Keep sending queries to a dual sided SUE until X number of diffs are observed in results.

How Evaluations Work

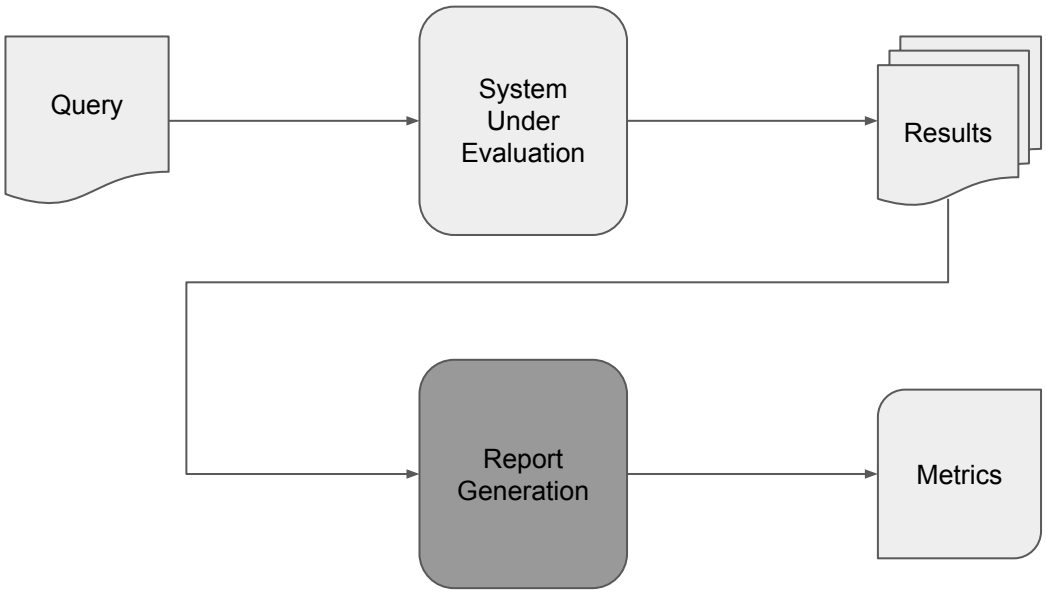


How Evaluations Work: Results

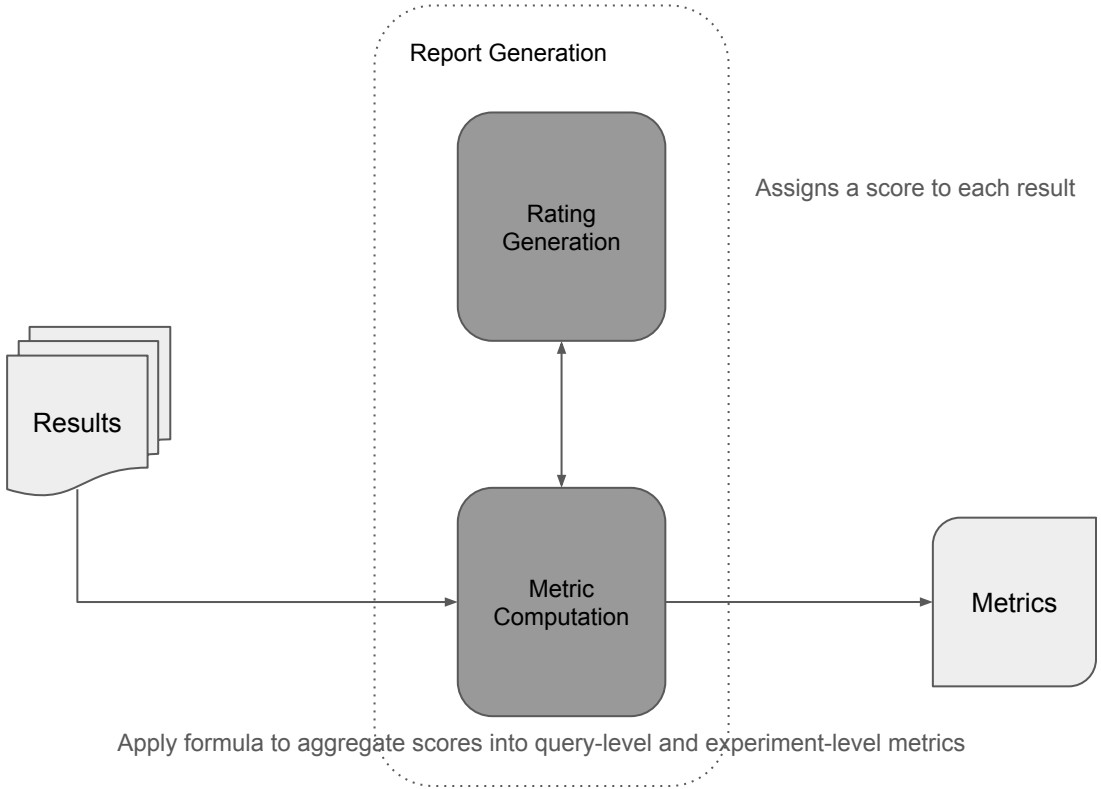
50k queryset dual sided evaluation generates 1M results!



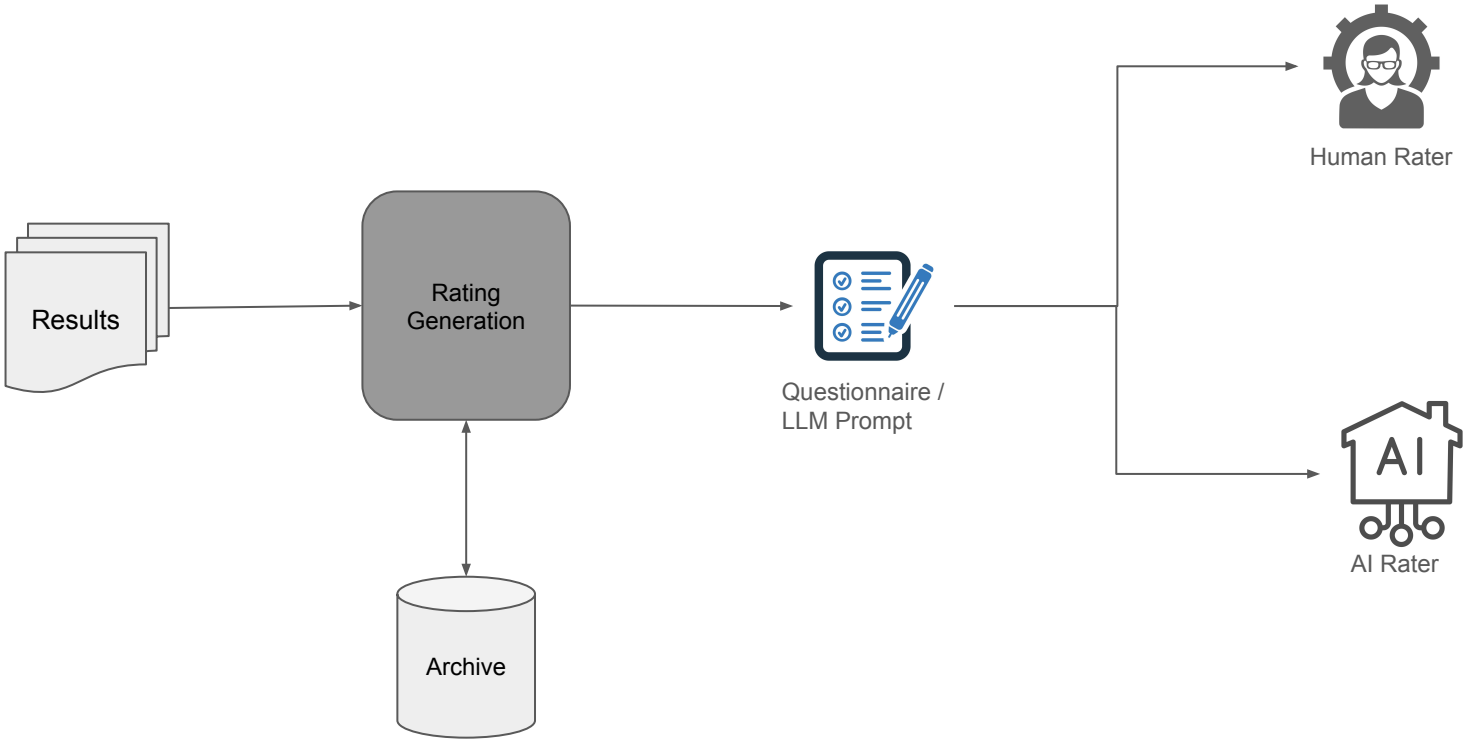
How Evaluations Work: Report Generation



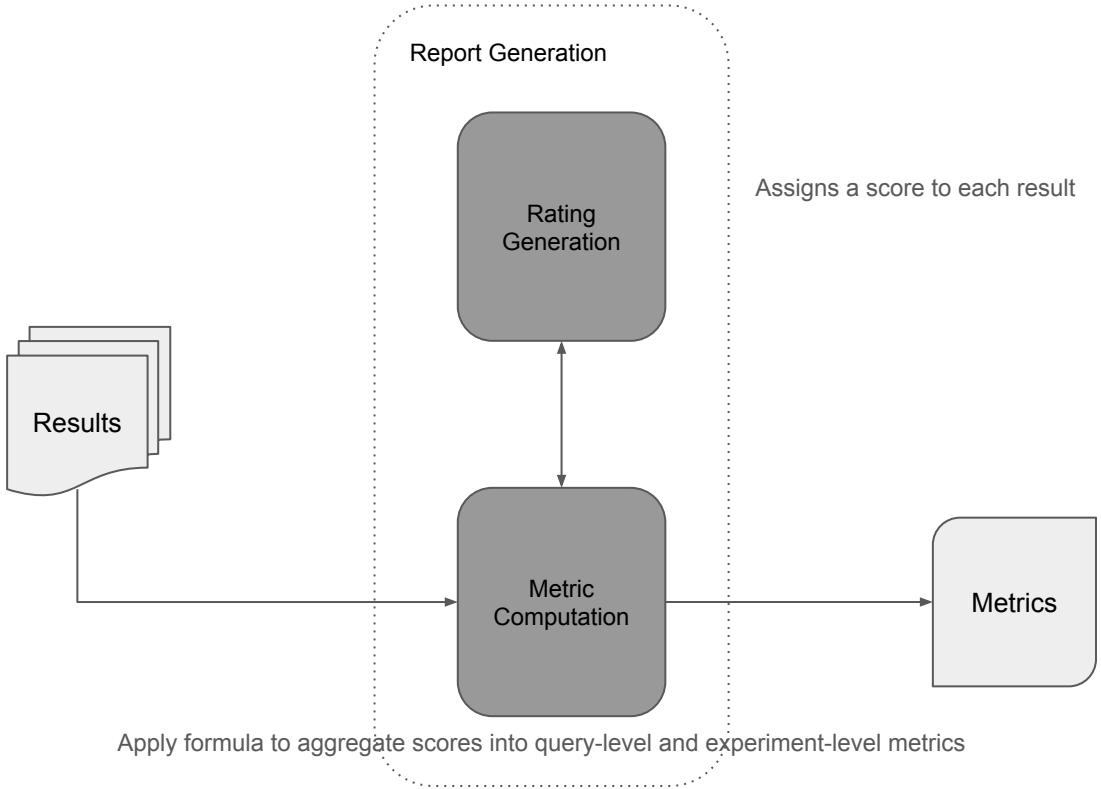
Report Generation



Rating Generation



Metric Computation



Metrics

- Presented with the help of a dashboard. Users can see dual-sided results page, ratings, metrics, etc.
- Aggregated views for query level and experiment level analysis.
- Users can filter and slice data using UI for specialized analysis.
- Results and metrics available in column oriented files for offline data analysis.

CI Triggers

Continuous Evaluation

- Evaluations that are periodically triggered.
- Tracks quality of system over time.
- Generates graphs that helps users visualize and analyze trends.
- Users can create customized graphs and dashboards.

Presubmit

- Tricorder: Generic presubmit framework that can trigger RPC requests to services when changes in certain directories are detected.
- Can trigger pre-configured evaluations as presubmit checks.
- Most common use case is for zero diff evaluations (for changes which are not expected to introduce diffs between test and base side).
- Optimization: Stacks for presubmits are auto updated when code changes are detected.

Challenges

Traffic Shaping and Prioritization

- Evaluations are queued up until enough resources (to run new jobs) are available.
- Tiered priority system where the first evaluation a user runs is assigned high priority.
- System triggered evaluations are assigned lower priority.
- When prod servers (serving live users) are out of quota, a back pressure signal is propagated back to the front end which stops scheduling more evaluations.

Scalability

- Initially created as a binary running locally.
- Run on cloud to standardize environment and reduce errors. Fresh executor jobs are created for each evaluation.
- Recent overhaul to have long running common pool of tasks with a leader/follower model.
- Qualified binaries using green CLs are used to create base side stacks, improving reliability and efficiency.

Other Challenges

- Standardization of Methodology
  - Too many advanced functionalities that don't always work well together.
  - Provides well lit paths for users and promotes best practices.

- Error Attribution: Complex system makes it harder to detect root cause when problems occur (and show it on UI).

- For continuous evaluations, human ratings can sometimes produce false alerts. AI ratings produce more trustworthy signals.

Q&A