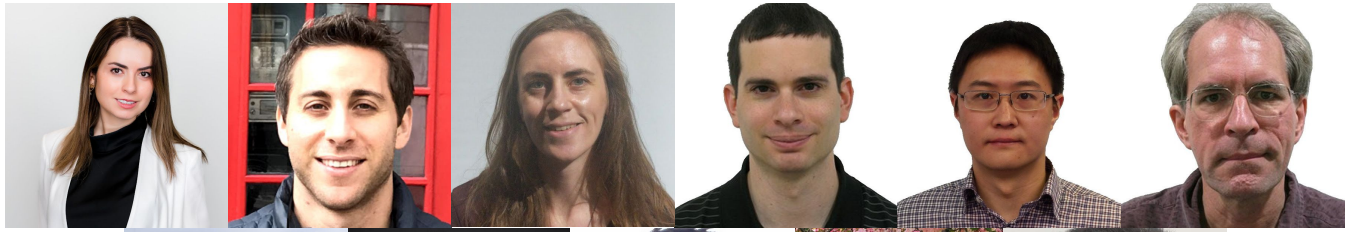
Open Source Vizier: Blackbox Optimization Service

Xingyou (Richard) Song
xingyousong@google.com

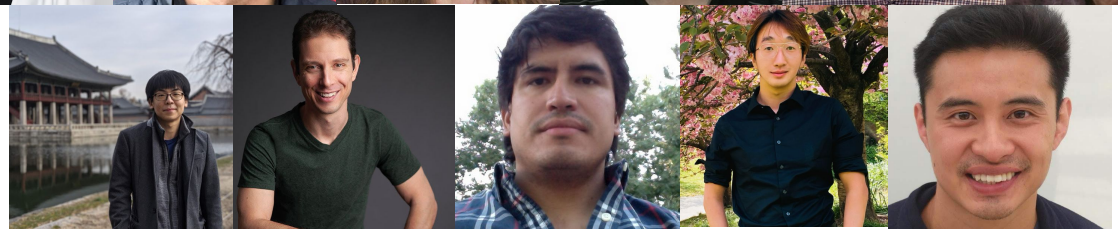
On behalf of the Vizier Team

Google Research

Vizier Team



Setareh Ariafar | Lior Belenki | Emily Fertig | Daniel Golovin | Tzu-Kuo Huang | Greg Kochanski

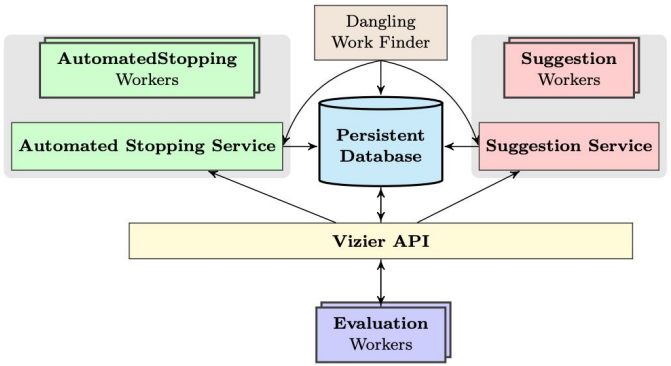


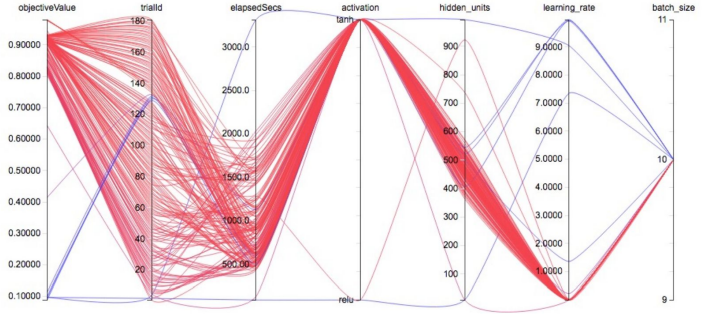
Chansoo Lee | Sagi Perel | Adrian Reyes | Xingyou Song | Richard Zhang

Google Research

Google Vizier (2017)

- Tunes many of Google's research + products
- Thousands of monthly users
- Tuned millions of objectives





Google Research

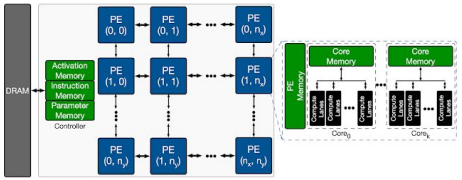
Notable Users / Downstream Wins

Production

- Search, Ads, Youtube

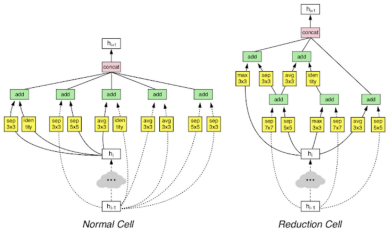
Tuning Research Results:

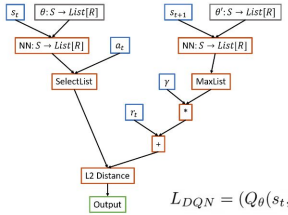
- Hardware Design, Robotics
- Protein Design



Backend for Evolution:

- Neural Architecture Search
- Symbolic Algorithm Search





$$L_{DQN} = (Q_\theta(s_t, a_t) - (r_t + \gamma * \max_a Q_{\theta'}(s_{t+1}, a)))^2$$

Google Research

Table of Contents

Systems

1. Why a Service?
2. Comparisons to Other Packages
3. OSS Vizier Infrastructure

API

4. User/Client API: Distributed Tuning
5. Developer API: Writing Algorithms

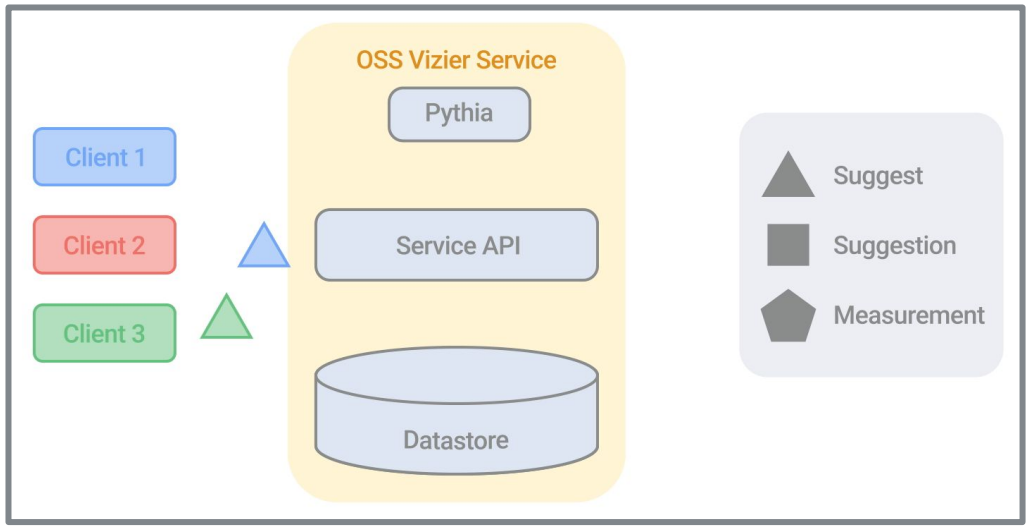
Research

6. Default Algorithm: GP-Bandit
7. Integrations
8. Future

Google Research

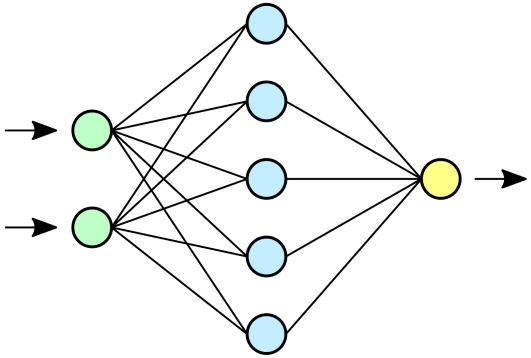
Questions?

Why a Service?

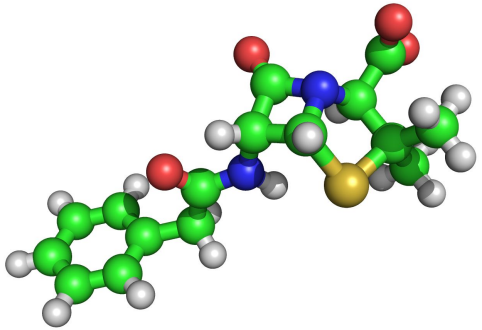


Google Research

The Wide Variety of Scenarios



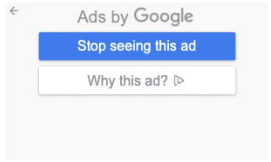
- Tuning large ML model hyperparameters
- Chemical/Biological processes
- Optimizing cookie recipes



Very different workflows!

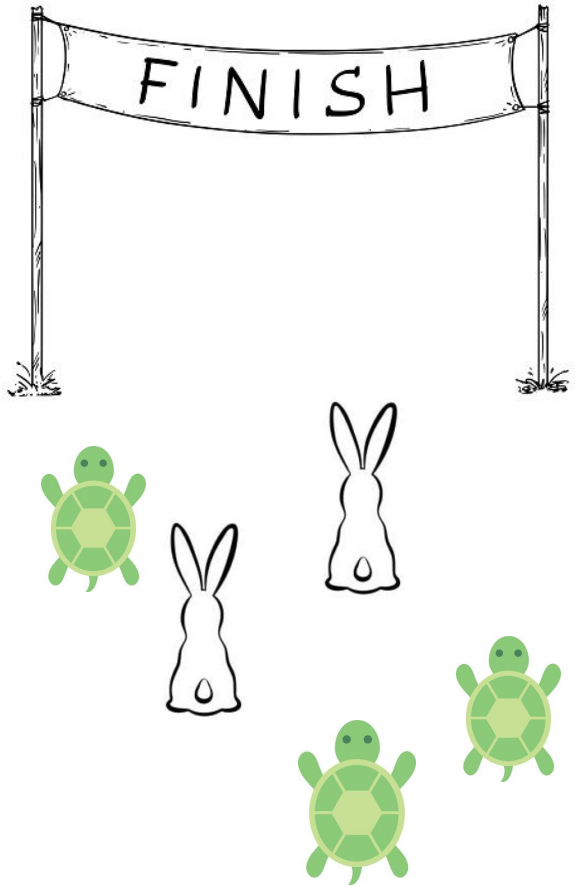
Google's new AI learns by baking tasty machine learning cookies

The system "designs excellent cookies", according to its creators



Workflow Possibilities

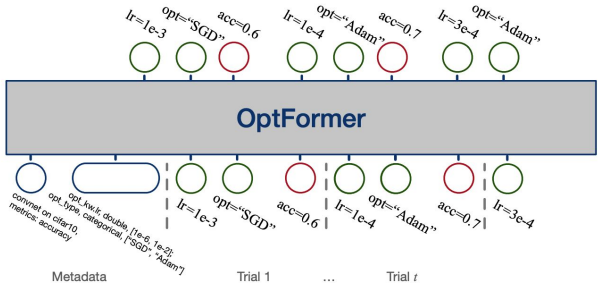
- Eval Latency: Seconds to Weeks
- Eval Budget: 10^1 to 10^7 Trials
- Asynchronous or Synchronous (Batched)
- Failed evaluations: Retried or abandoned
- Early Stopping



Google Research

Benefits of a Service: No Evaluation Assumptions!

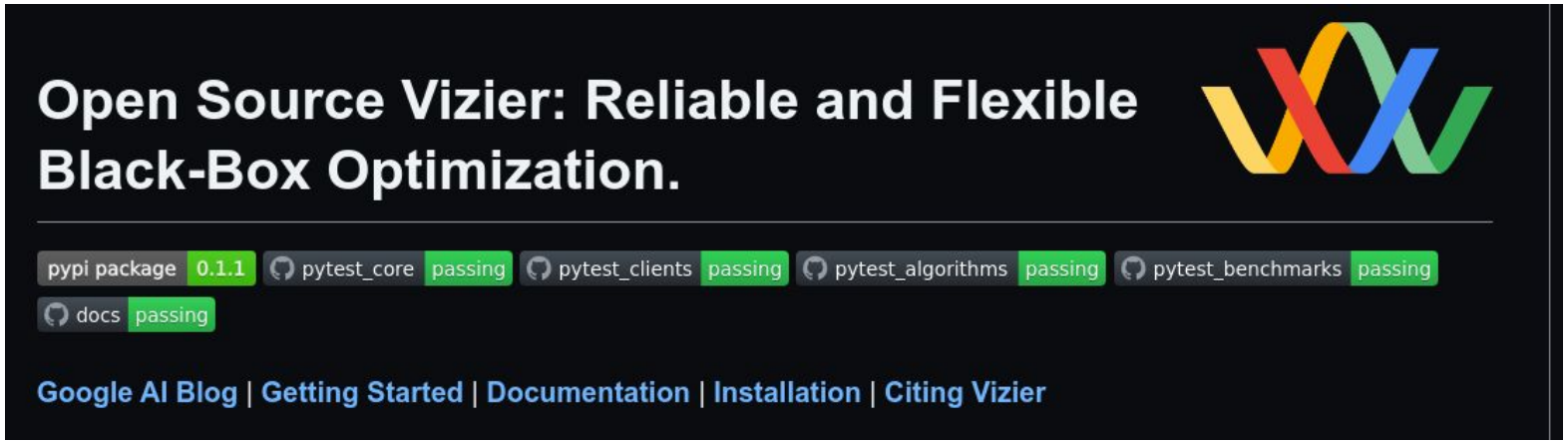
- Users have freedom of when to:
  - Request trials
  - Evaluate Trials
  - Report results
- Service can preserve data on prior usage
  - Led to [OptFormer paper](#)!



Google Research

OSS Vizier: 2022

- Standalone + customizable Python codebase
- User can host service



Google Research

Comparisons to Other Packages

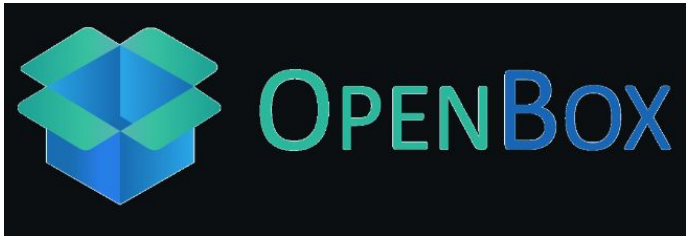
Google Research

Types of Packages

- **Services:** Host algorithms on a server.
  - More flexible + scalable
  - Additional engineering complexity
- **Frameworks:** Execute entire optimization (both algorithm + objective)
  - Convenient, full automation
  - Requires same programming language + knowledge of entire eval pipeline
- **Libraries:** Implement blackbox optimization algorithms
  - Offer most freedom
  - Lack scalability features / limited to single machine / same programming language

Services

- **OSS Vizier (ours)**
- Advisor (2017)
- OpenBox (2021)



Google Research

Frameworks

- Ax (2021)
- HpBandSter (2018)

Google Research

Libraries

- BoTorch (2020)
- HyperOpt (2013)
- Dragonfly (2020)

Comparisons

Name	Type	Client Languages	Parallel Trials	Features*
OSS Vizier	Service	Any	Yes	Multi-Objective, Early Stopping, Transfer Learning, Conditional Search
SMAC	Framework	Python	Yes	Multi-Objective, Multi-fidelity, Early Stopping, Conditional Search, Parameter Constraints
Advisor	Service	Any	Yes	Early Stopping
OpenBox	Service	Any	Yes	Multi-Objective, Early Stopping, Transfer Learning, Parameter Constraints
HpBandSter	Framework	Python	Yes	Early Stopping, Conditional Search, Parameter Constraints
Ax + BoTorch	Framework	Python	Yes	Multi-Objective, Multi-fidelity, Early Stopping, Transfer Learning, Parameter and Outcome Constraints
HyperOpt	Library	Python	No	Conditional Search
Emukit	Library	Python	No	Multi-Objective, Multi-fidelity, Outcome Constraints

OSS Vizier Infrastructure

Google Research

Distributed Communication

- Remote Procedure Calls (RPCs) formatted as Protocol Buffers (protobufs)
- Server + Client classes based on gRPC

Google Research

PyVizier: Abstracting away Protobufs

- Hides away RPC protobufs from user + algorithms
- Use same Python libraries across all Vizier variants
- More Pythonic data structures

```python
from vizier.service import study_pb2
from google.protobuf import struct_pb2

param_1 = study_pb2.Trial.Parameter(parameter_id='learning_rate', value=struct_pb2.
    Value(number_value=0.4))
param_2 = study_pb2.Trial.Parameter(parameter_id='model_type', value=struct_pb2.
    Value(string_value='vgg'))
metric_1 = study_pb2.Measurement.Metric(metric_id='accuracy',value=0.4)
metric_2 = study_pb2.Measurement.Metric(metric_id='num_params',value=20423)
final_measurement = study_pb2.Trial.Measurement(metrics=[metric_1,metric_2])
trial = study_pb2.Trial(parameters=[param_1,param_2], final_measurement=
    final_measurement)
```

Original Protobuf: Verbose + Complex

```python
from vizier.pyvizier import ParameterDict, ParameterValue, Measurement, Metric,
    Trial

params=ParameterDict()
params['learning_rate'] = ParameterValue(0.4)
params['model_type'] = ParameterValue('vgg')
final_measurement = Measurement()
final_measurement.metrics['accuracy'] = Metric(0.7)
final_measurement.metrics['num_params'] = Metric(20423)
trial = pv.Trial(parameters=params,final_measurement=final_measurement)
```

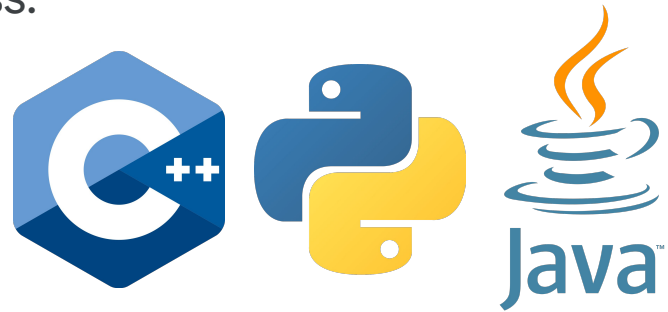
PyVizier: More Pythonic!

Google Research

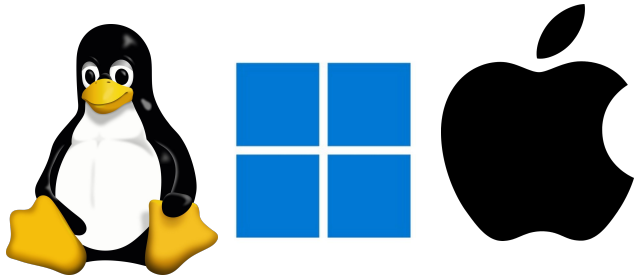
Language + Platform Independence

Protobufs (for RPC Backend) are ubiquitous across:

- Languages
  - C++, Python, Java, and many more

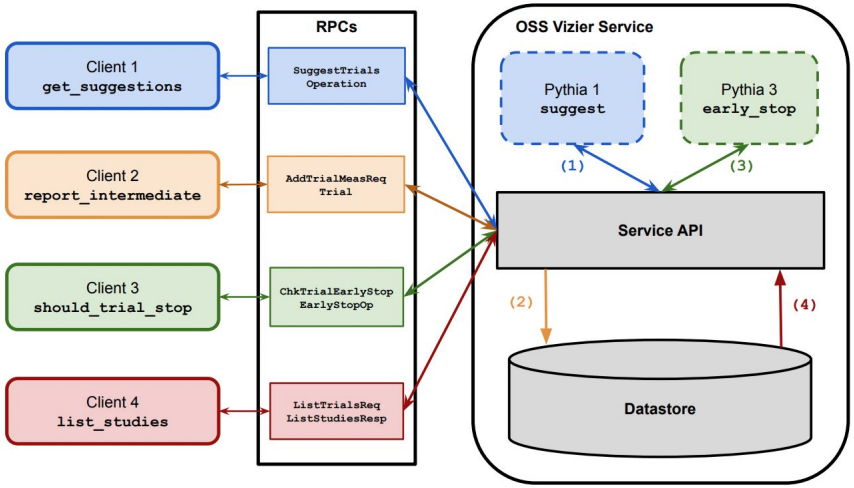


- Platforms
  - Linux, Windows, Mac



Core Server-Client Procedure

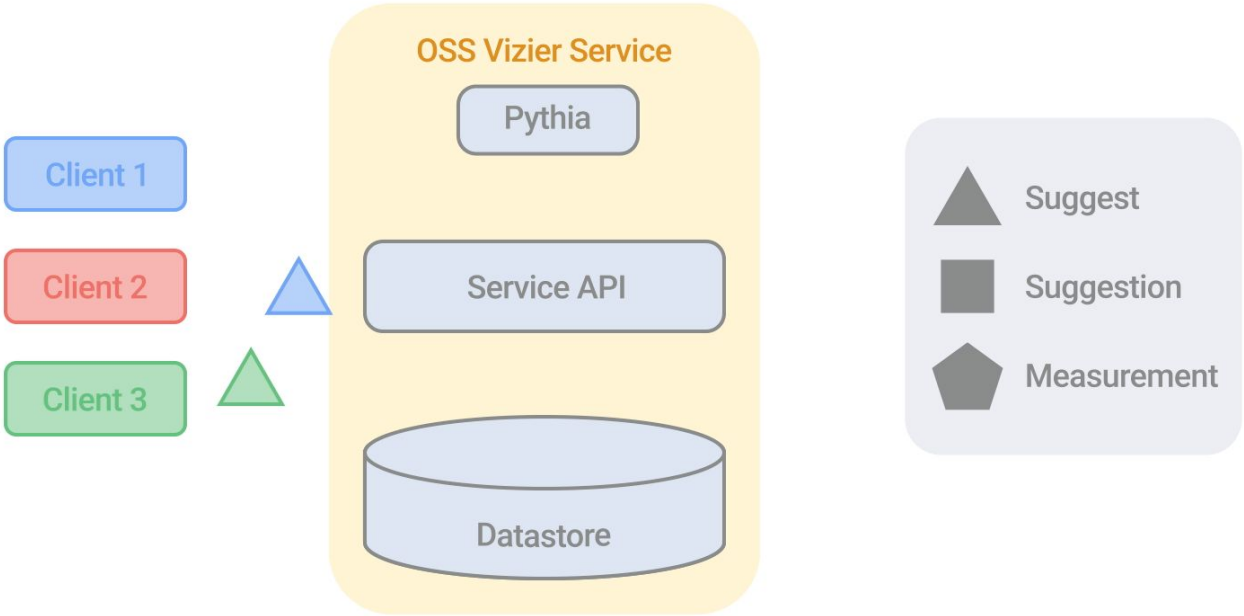
- Client sends SuggestTrials RPC to Server.
- Server starts Pythia policy
  - `Operation` protobuf to keep track of everything
- Client repeatedly pings server on status of `Operation`
- Client finally receives suggestion



All transactions + operations are stored in server datastore!

Google Research

Suggestion Animation (Full)



Google Research

Questions?

User/Client API: Distributed Tuning

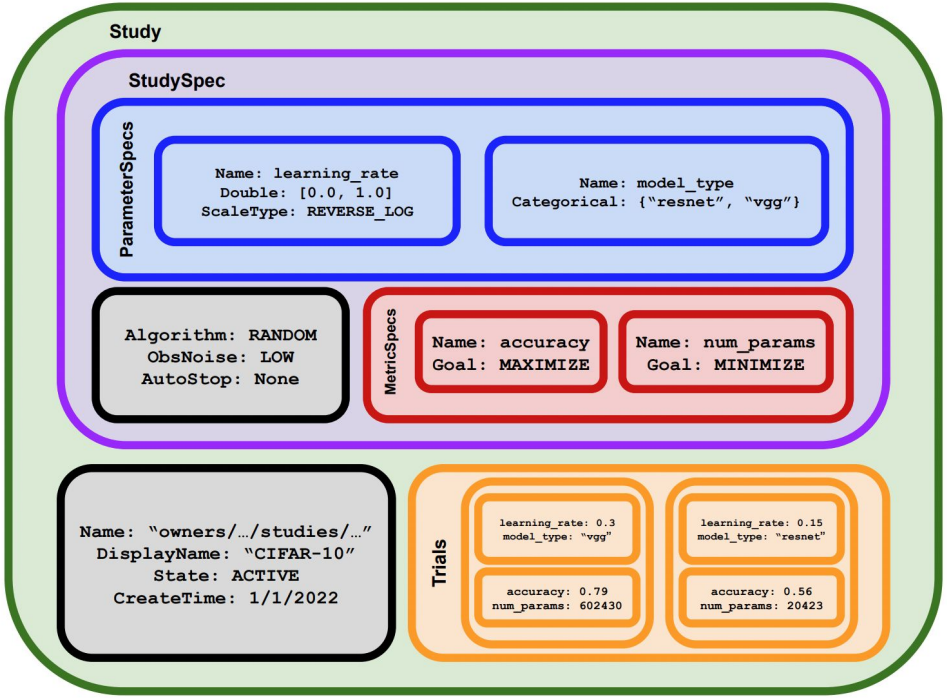
Google Research

Definitions

- **Study:**

  Entire Optimization Run
- **StudySpec: Configuration**
  - **Search Space**
  - **Algorithm**
  - **Noise**
  - **...**
- **ParameterSpec: Parameter Specification**
- **MetricSpec:**

  Metric Specification



Google Research

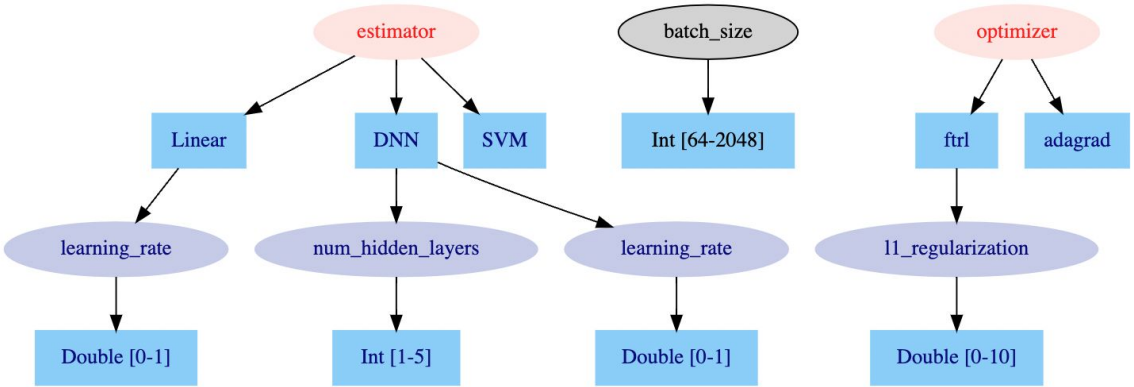
Search Space Construction: ParameterSpecs

Core:

- Double: Continuous range [a,b]
- Integer: Integer range [a,b]
- Discrete: Finite set of floats.
- Categorical: Finite set of strings.

Each ParameterSpec also contains:

- Scaling Type (uniform, log)
- Child/Conditional Parameters



Google Research

Setting up Client

```
# Algorithm, search space, and metrics.
study_config = vz.StudyConfig(algorithm='GAUSSIAN_PROCESS_BANDIT')
study_config.search_space.root.add_float_param('w', 0.0, 5.0)
study_config.search_space.root.add_int_param('x', -2, 2)
study_config.search_space.root.add_discrete_param('y', [0.3, 7.2])
study_config.search_space.root.add_categorical_param('z', ['a', 'g', 'k'])
```

```
study = clients.Study.from_study_config(study_config, owner='my_name', study_id='example')
```

Google Research

Setting up Server (Optional)

- Server will be implicitly + locally created if not specified.

```
server = vizier_server.DefaultVizierServer(host=FLAGS.host)
```

Google Research

Tuning Loop

Loop involves:

- Client obtains suggestions from server
- Evaluating suggestions
- Completing suggestions + updating server

```
for i in range(10):
  suggestions = study.suggest(count=1)
  for suggestion in suggestions:
    params = suggestion.parameters
    objective = evaluate(params['w'], params['x'], params['y'], params['z'])
    suggestion.complete(vz.Measurement({'metric_name': objective}))
```

Google Research

Developer API: Writing Algorithms

Google Research

Typical Algorithm Design: “Designer”

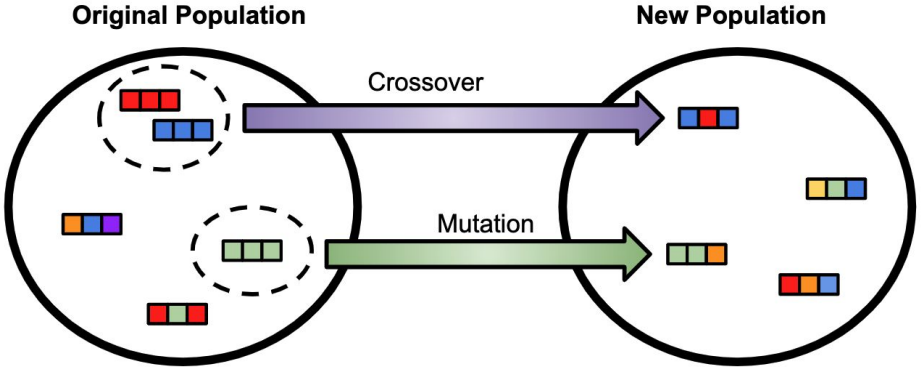
- Very typical API for writing an algorithm:

```python
class Designer(...):
  """Suggestion algorithm for sequential usage."""

  @abc.abstractmethod
  def update(self, completed: CompletedTrials, all_active: ActiveTrials) -> None:
    """Updates recently completed and ALL active trials into the designer's state."""

  @abc.abstractmethod
  def suggest(self, count: Optional[int] = None) -> Sequence[vz.TrialSuggestion]:
    """Make new suggestions."""
```

Google Research

Service Requirements

- Ensure fault-tolerance on algorithms:
  - Fresh algorithm can recover when needed
  - Use historical trials as “algorithm state”!
- Querying the history:
  - Algorithm can query whichever trials they need.
  - Very useful for algorithms which work in batches/populations
    - e.g. Genetic Algorithms



Hosted Algorithm: “Policy”

- **`PolicySupporter:`** Query previous trials to recover state.
- **`stateless_algorithm:`** Stateless algorithm or Designer

```
class TypicalPolicy(Policy):

  def __init__(self, policy_supporter: PolicySupporter):
    self._policy_supporter = policy_supporter

  def suggest(self, request: SuggestRequest) -> SuggestDecision:
    all_completed = policy_supporter.GetTrials(status_matches=COMPLETED)
    all_active = policy_supporter.GetTrials(status_matches=ACTIVE)
    suggestions = stateless_algorithm(all_completed, all_active)
    return SuggestDecision(suggestions)
```

Google Research

Algorithms Included

- **Classic:** Random, Grid, Shuffled-Grid, Quasi-Random
- **Evolution:** CMA-ES, NSGA2
- **Boolean:** BOCS, Harmonica
- **Bayesian:** GP-Bandit

Google Research

Default Algorithm: GP-Bandit

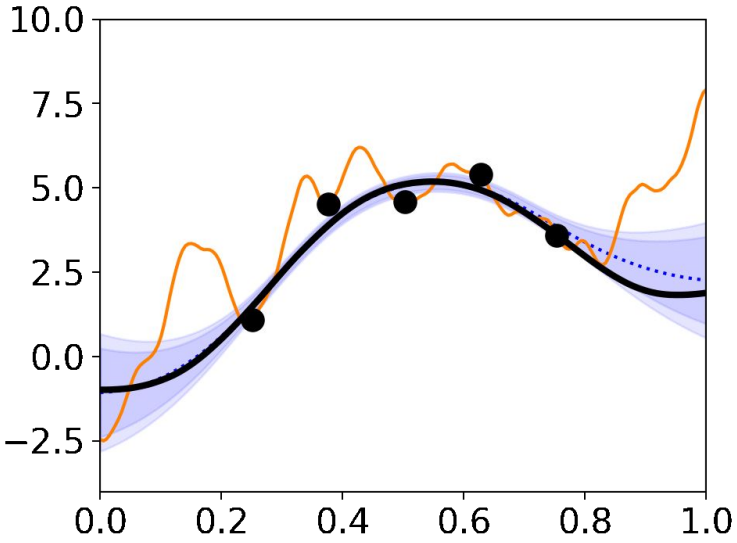
Google Research

Animation from [Wang et al., 2023 “Pre-trained Gaussian processes”]

Vizier GP-Bandit Main Components

Inspired by original 2015 C++ implementation **(before AutoDiff)**

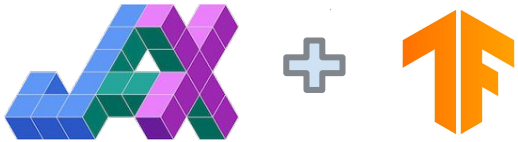
- Gaussian Process Kernel
  - Matern-5/2
- Upper-Confidence Bound Acquisition
  - Evolutionary Optimizer “Eagle Strategy”
- Objective Warping
  - Outlier removal + Gaussian-fitting + Log warping
- ARD Optimization
  - JAX-based LBFGS-B



Google Research

Advantages over other BayesOpt Algorithms

- AutoDiff + GPU support via JAX + Tensorflow Probability
  - Most other packages only use NumPy or Sklearn
- “Advanced” Tricks
  - Trust Region, Warping, ARD-optimization, Self-Tuning
- Industry-Grade Code Quality
  - PyType, Rigorous testing, Clean abstractions



OPTUNA

Dragonfly
Scalable Bayesian Optimisation

SIGOPT

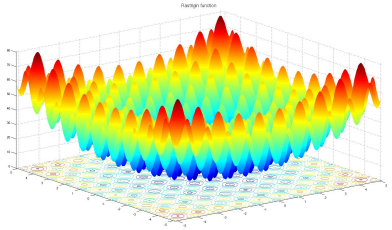
Google Research

Questions?

Integrations

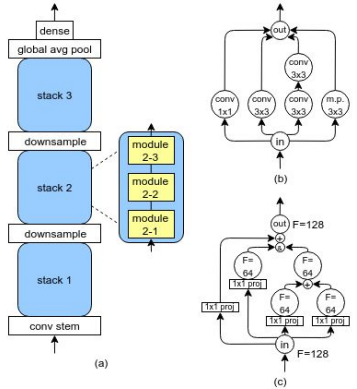
Google Research

Benchmarks



Included:

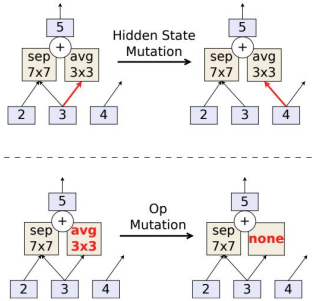
- BBOB, COMBO
- NASBENCH (101 + 201)
- HPOB
- Atari100K
- Utilities (Noise, Shifting, Sparsifying, etc.)

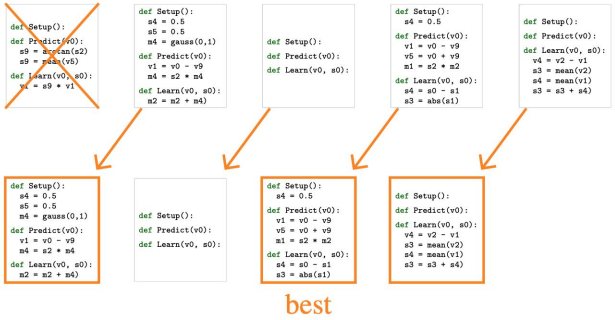


[Daiyi Peng et al., 2021 “PyGlove: Symbolic Programming for Automated Machine Learning”]

PyGlove: Evolutionary + Combinatorial Computation

- OSS Vizier only supports flat search spaces + conditionals
  - Lacks choice function: $\binom{n}{k} = \frac{n!}{k!(n-k)!}$
- Integrate Vizier backend w/ PyGlove!
  - Vizier handles distributed system
  - Ex: Evolution for NAS, Genetic Programming

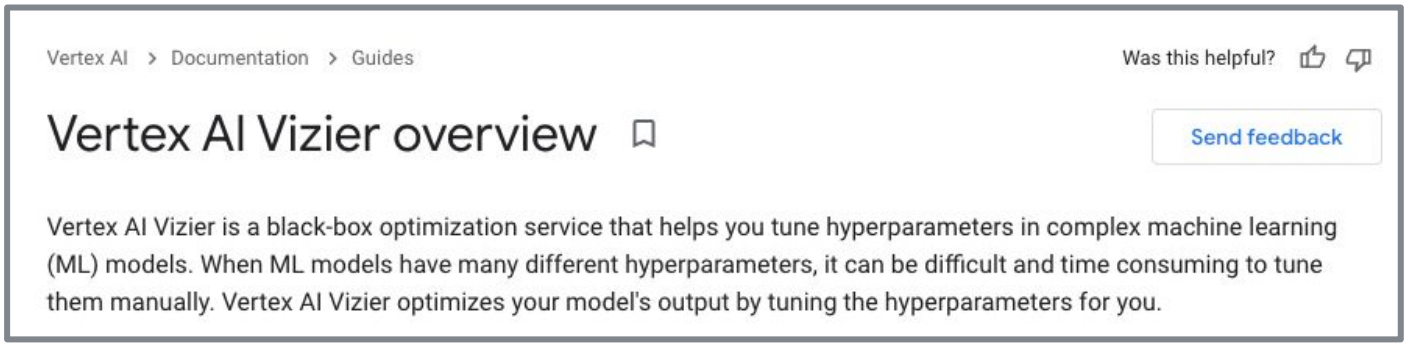




Google Research

Vertex/Cloud Vizier

- Prod service for external users / businesses
- Shared client API: Easily switch b/w OSS or Cloud



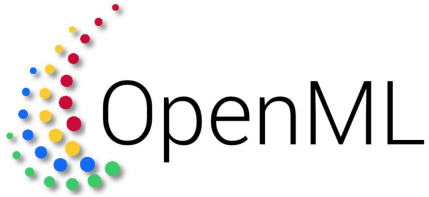
Google Research

Future

Google Research

Potential + On-going External Integrations

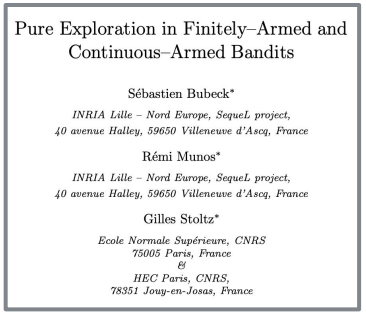
- Potential algorithm add-on to RayTune
- Cross-study integration w/ OpenML



Google Research

Algorithms

- Upcoming GP-UCB-PE algorithm
  - PE = “Pure Exploration”
- Baseline reimplementations
  - Ex: HEBO, TuRBO
- Upcoming whitepaper on Vizier’s GP algorithms
  - Exact descriptions to allow reproducibility
  - Comparisons to existing packages


Pure Exploration in Finitely–Armed and Continuous–Armed Bandits

Sébastien Bubeck*

INRIA Lille – Nord Europe, SequeL project, 40 avenue Halley, 59650 Villeneuve d'Ascq, France

Rémi Munos*

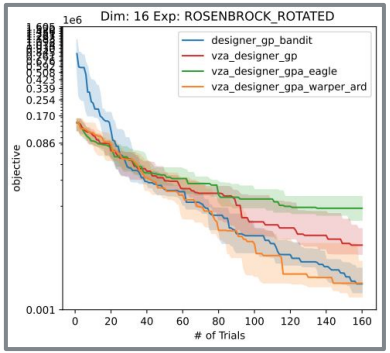
INRIA Lille – Nord Europe, SequeL project, 40 avenue Halley, 59650 Villeneuve d'Ascq, France

Gilles Stoltz*

Ecole Normale Supérieure, CNRS 75005 Paris, France & HEC Paris, CNRS, 78351 Jouy-en-Josas, France

HEBO 河伯

Heteroscedastic Evolutionary Bayesian Optimisation



Google Research

Links

Code: https://github.com/google/vizier

Documentation: https://oss-vizier.readthedocs.io/en/latest/index.html

AI Blog:

https://ai.googleblog.com/2023/02/open-source-vizier-towards-reliable-and.html

Paper: https://arxiv.org/abs/2207.13676

OpenReview: https://openreview.net/forum?id=SfIRITSUxc

Google Research

Thanks!

Google Research