Google Vizier: A Service for Black-Box Optimization

Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, D. Sculley

Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, Ameet Talwalkar

By Matthew, Jiho and Zineb

Google Vizier: A Service for Black-Box Optimization

Motivation

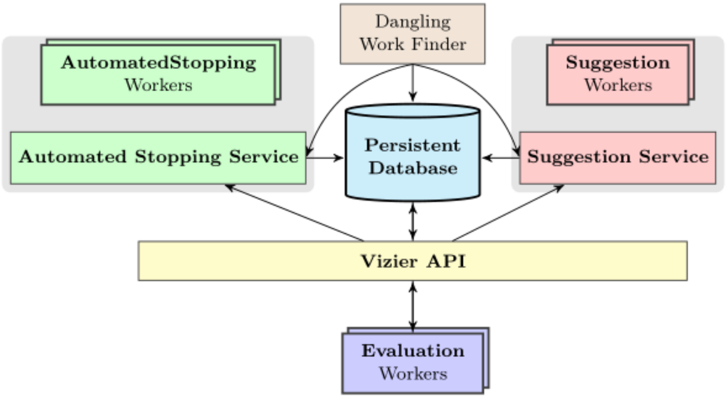
- The "black-box" is a function which can be evaluated with any input but of which no other information (gradient, Hessian) is known
- Evaluating the function may be expensive, so it is important to determine more efficient choices in inputs
- Used to calculate the best inputs to a system with measurable output
    - Hyperparameters of ML systems
    - Optimizing physical systems e.g. optimizing airfoils in simulation
- Minimal assumptions about the black-box make it applicable to many domains

Quick definitions

- Trial: list of parameter values, *x*, that will lead to an evaluation of *f(x)*
- Study: a run of evaluating a set of Trials
- Worker: evaluate a pending Trial

System Overview

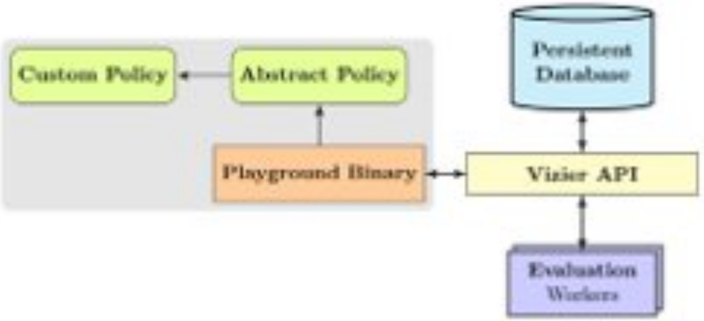
- Implemented as a service accessible with RPC API
- Modular system
    - Dangling work finder
    - Persistent database
    - Suggestion service
    - Automated stopping service
- Workers enable running multiple trials in parallel



Infrastructure

- Suggestion service is distributed across datacenters, further distributed to machines inside of each datacenter
    - Global locking with timed lease insures only one instance is generating suggestions for a given study at a time. DanglingWorkFinder process will reassign study to another Suggestion Service process
- DanglingWorkFinder will recognize Study that is failing too often and alert engineers

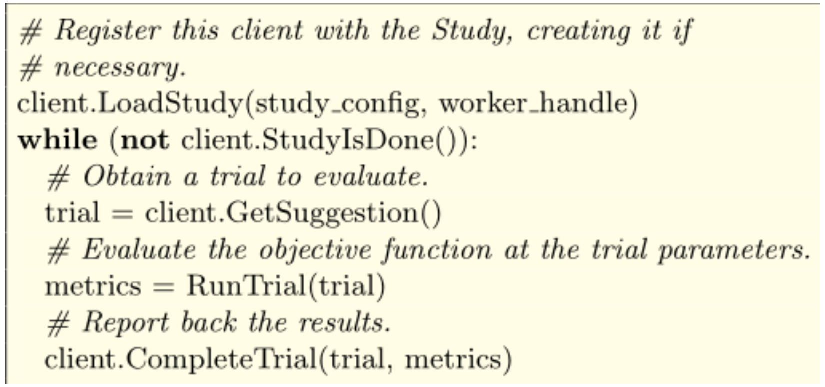
Infrastructure



- Algorithm Playground allows users to safely use their own code to add the Trials to Studies
- Benchmarking Suite allows for easy testing with many objective functions, including user-supplied
  - Distributed system, so many users can run benchmarks simultaneously
- Web dashboard monitors and effects Vizier studies
  - Same functionality as Vizier API
  - Allows users to track the progress of a study with interactive visuals
  - Enables creating, updating, and deleting a study as well as suggesting and stopping

User Workflow

- RPC APIs implemented in C++, Python, and Golang
- Users specify a *study configuration*
    - Set of parameters with feasibility sets (the ranges of parameters)
    - Feasibility sets may be continuous or discrete



```
# Register this client with the Study, creating it if
# necessary.
client.LoadStudy(study_config, worker_handle)
while (not client.StudyIsDone()):
  # Obtain a trial to evaluate.
  trial = client.GetSuggestion()
  # Evaluate the objective function at the trial parameters.
  metrics = RunTrial(trial)
  # Report back the results.
  client.CompleteTrial(trial, metrics)
```

Algorithms

- Studies with <1000 trials default to Batched Gaussian Process Bandits
    - Find local maxima with a gradient-free hill climbing algorithm with random starting points
    - Researchers found that Bayesian deep learning models depend too much on their hyperparameters as currently understood
- Larger studies use a variety of search algorithms

Stopping Algorithms

- Automated Early Stopping uses intermediate performance measures to identify and terminate trials that are unlikely to do well
  - Accessible in Playground
- Automated Stopping Algorithms look at the full state of all trials
  - The Performance Curve Stopping Rule estimates the result of a Trial given previously completed Trials and its own intermediate results
  - The Median Stopping Rule stops a Trial at step *s* if it's best objective value yet is less than the median objective value reached by Trials by step *s*

Transfer Learning

- Want to minimize work repeated from a previous, similar Study
- Create a stack of Gaussian Process regressors
  - Each is assosciated with a single study
  - Trained relative to the previous Gaussian Process regressor
- Each regressor is weighted by its number of completed Trials
- Used in conjunction with the default Batched Gaussian Process Bandits algorithm
- Poorly chosen prior Studies will minimally affect new Studies as more Studies are performed

Results

Performance evaluation:

- Pre-selected functions
- known optimal points

How is the success of the optimizer measured : **Optimality Gap**

- f : benchmark function
- $x*$ minimizes f, x' is the best solution

=> $|f(x') - f(x*)|$

Empirical Results

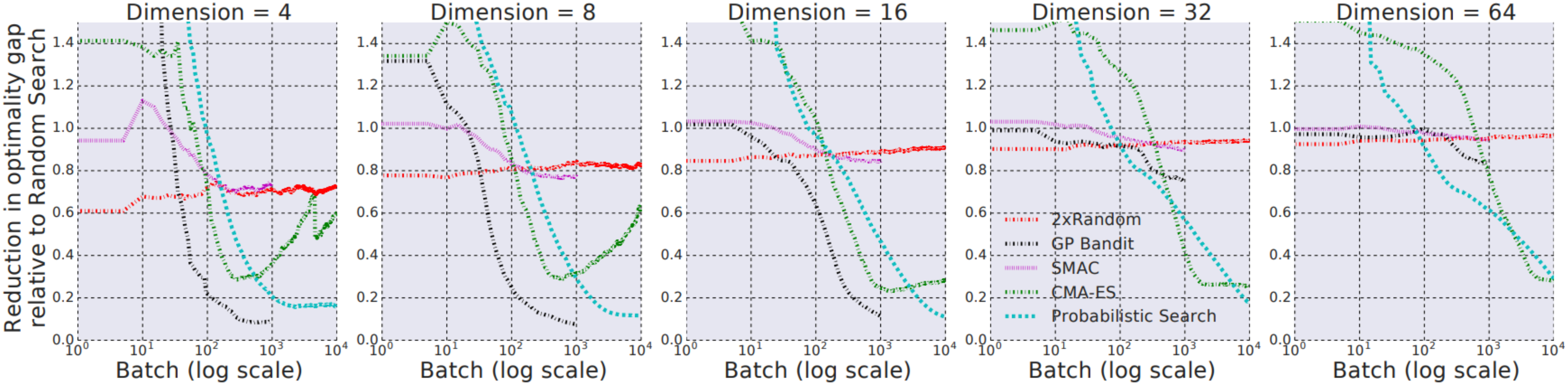


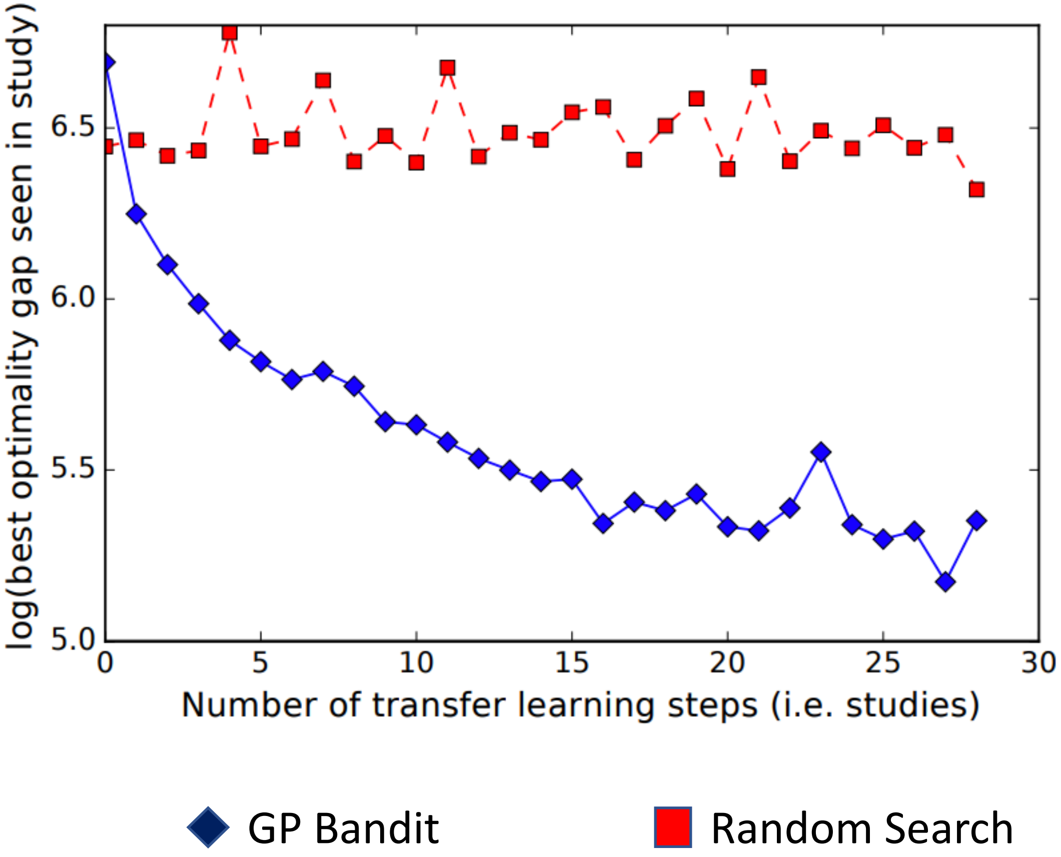
Figure 6: Ratio of the average optimality gap of each optimizer to that of Random Search at a given number of samples. The 2×Random Search is a Random Search allowed to sample two points at every step (as opposed to a single point for the other algorithms).

Transfer Learning

Benefits over time :

- **consistent** improvement in optimality gap

=>effective transfer of knowledge from the earlier trials



Automated stopping

- Performance Curve Stopping Rule:
  - Comparable to those achieved without the PCSR with 50% fewer CPU-hours while tuning hyperparameter for DNN

- Median Automated Stopping Rule
  - 2x to 3x speedup over Random Search while always finding the best performing trial

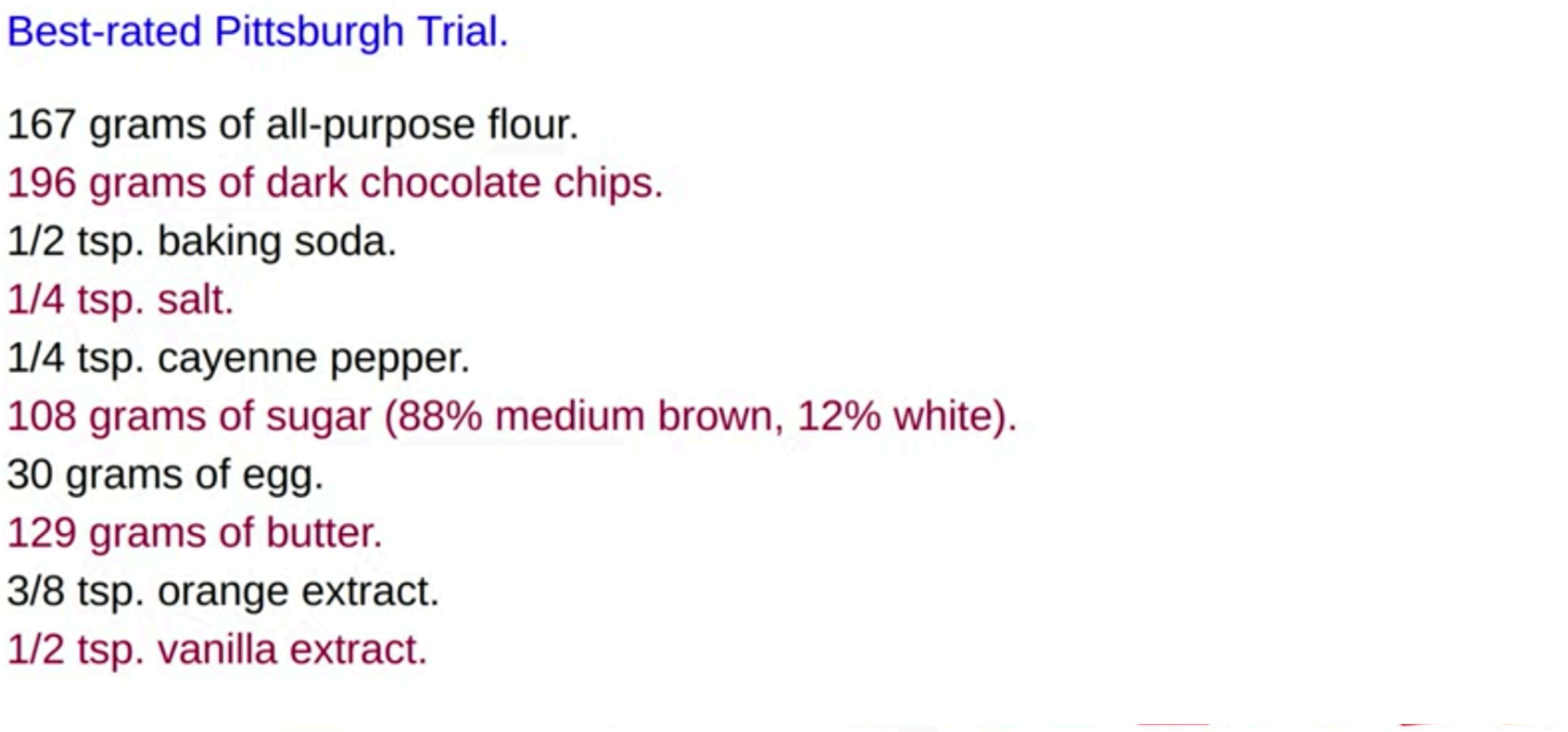
Use Cases

- Hyperparameter tuning and Hypertune
- Automated A/B testing
- Delicious Chocolate Chip Cookies

Machine Learning cookie

- Infeasible trials
  - Cannot be evaluated -> training could diverge -> garbage models
- Manual override of suggested trials
  - Updating/deleting trials
- Transfer Learning
  - Small-scale then large-scale

Bonus Slide : best cookie’s recipe



Best-rated Pittsburgh Trial.

167 grams of all-purpose flour.
196 grams of dark chocolate chips.
1/2 tsp. baking soda.
1/4 tsp. salt.
1/4 tsp. cayenne pepper.
108 grams of sugar (88% medium brown, 12% white).
30 grams of egg.
129 grams of butter.
3/8 tsp. orange extract.
1/2 tsp. vanilla extract.

Piazza discussion questions

- "The first sentence of the paper is the following: "Any sufficiently complex system acts as a black box when it becomes easier to experiment with than to understand." It seems to me that black box optimization in general would yield very little information about why certain parameter values are more optimal, rather it would just determine the optimal values. I feel like knowing *why* a certain parameter value is optimal is very important in most use cases, why is this less important in black box optimization?"
  - Useful where the system is not well understood or too complex to realistically spend the time to analyze properly
  - Applicable where the function trying to be optimized is not sutible for gradient descent

Piazza discussion questions

- "Could you explain figure 4? I wasn't able to fully understand it. "
- Each strand is a Trial
- Each parallel line represents a dimension of information on the Trial
- Parallel coordinates are a general analysis visualizer used in different fields

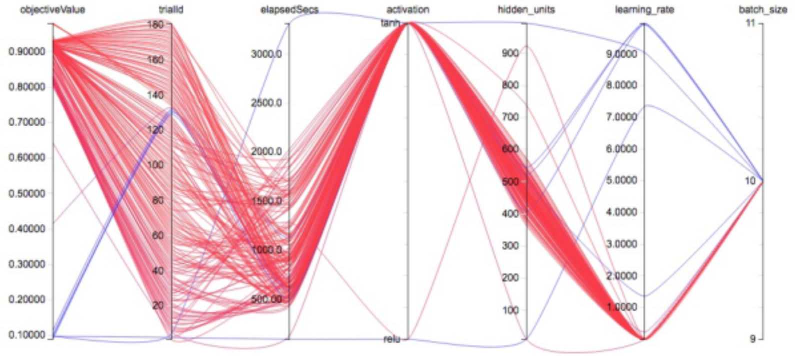


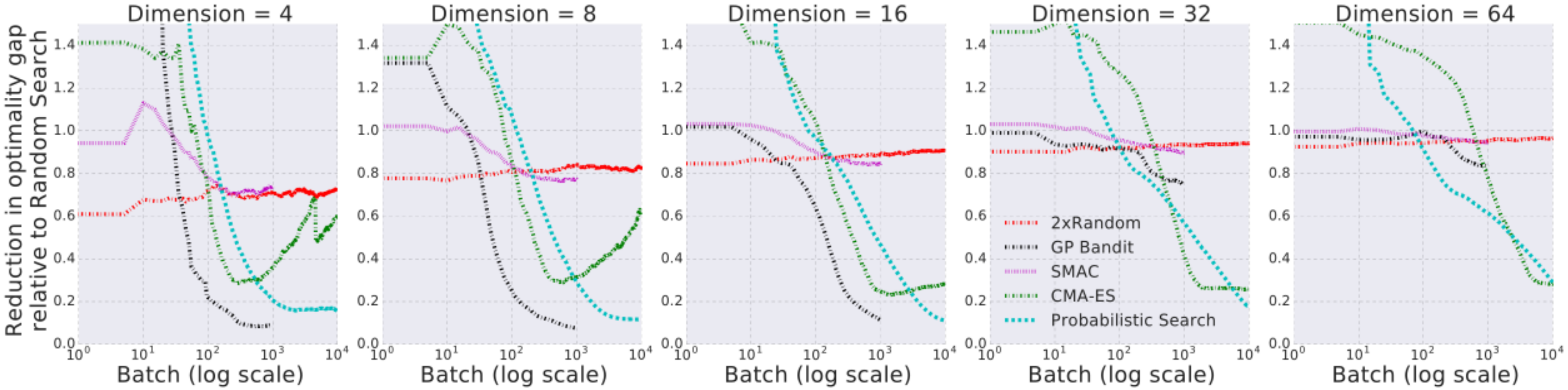
Figure 4: The Parallel Coordinates visualization [18] is used for examining results from different Vizier runs. It has the benefit of scaling to high dimensional spaces (~15 dimensions) and works with both numerical and categorical parameters. Additionally, it is interactive and allows various modes of slicing and dicing data.

Piazza discussion questions

- "Of the four optimization algorithms currently implemented that were discussed in 4.2, do you know if Vizier has any way of helping a user choose which algorithm to use? Or is that on the user to be familiar with these algorithms and choose one for their use case?"
- By default, the system will dynamically choose which algorithm is most likely to perform better for a given Trial or Study

Piazza discussion questions

- "Something that stood out to me was that in Figure 6, CMA-ES appears to get worse compared to random search at a certain point (Usually between 10^2 and 10^3) when the dimensionality of the problem is low. I was hoping to discuss why that might be.”
- The only point to notice is that they run some benchmarks on d dimension problems with different algorithms. The one that are best in 4 dimension are different from the ones that are best in 32 dimensions. This not related to properties of a particular algorithm but more of a general phenomenon.
- Vizier automatically detects the characteristics of your problem and then select the best algorithm for those problems.



Discussions

- The paper states that Vizier resulted in measurably better user experiences for over a billion people? What products and how?
- Comparison to other black-box optimizer?
- Benefits of using a Database?

Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization

Motivation

- What is Hyper Parameter Optimization?

Neural Cell : {1, 2, 3, 4, 5, 6}

Learning Rates : {1, 2, 3, 4, 5}

Regularization : {1, 2, 3, 4, 5, 6, 7}

# of configuration = 210 (6x5x7)

Challenge

Hyperparameter optimization in Machine Learning

- Growing number of Tuning Parameters difficult to set
- how these hyperparameters interact with each other to affect the resulting model

Hyperband Algorithm : Speedup the evaluation of Hyperparameter configurations

+ Identify a good set of hyperparameters

Motivation

Theory?

Speeding up random search as it offers a simple and theoretically principled launching point

How?

- formulate hyperparameter optimization as a pure-exploration adaptive resource allocation problem
- early-stopping strategy to allocate resources
- general-purpose technique that makes minimal assumptions

Benefits?

=> 5× to 30× faster than Bayesian optimization

Strategy

- Early stopping
- Train on a subset of Data
- Subset of Features

Search Algorithm : Random search

Strategy : Speedup the Random search through the hyperband strategy of resource allocation

Related work

- Hyperparameter Optimization:
  - Bayesian optimization techniques
  - Gaussian process
  - …

What is the novel idea?

the evaluation time is relatively inexpensive and the goal is to early-stop long-running training procedures by evaluating partially trained models on the full validation set.

Related work

- Bandit algorithm
  - Previous works either assume stochastic rewards or need to know something about the underlying function

What is the novel idea?

Hyperband is devised for the non-stochastic setting and automatically adapts to unknown cumulative distributed function without making any parametric assumptions.

Hyperband Algorithm

- Two main components
  - Successive Halving
  - Hyperband

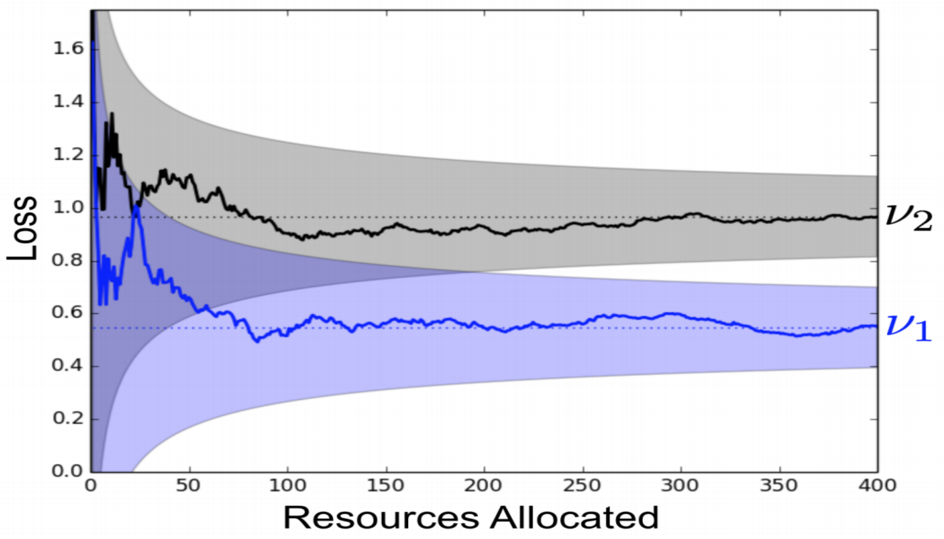
Successive Halving

- Finite amount of resources
  - Would like to compare many different configurations in order to get the best hyperparameter values.

- Successive Halving
  - Splits the resources uniformly
  - Every round until one configuration exists:
    - Discards poor performing half of configurations
    - Keeps the other half configurations.

Choosing n value?

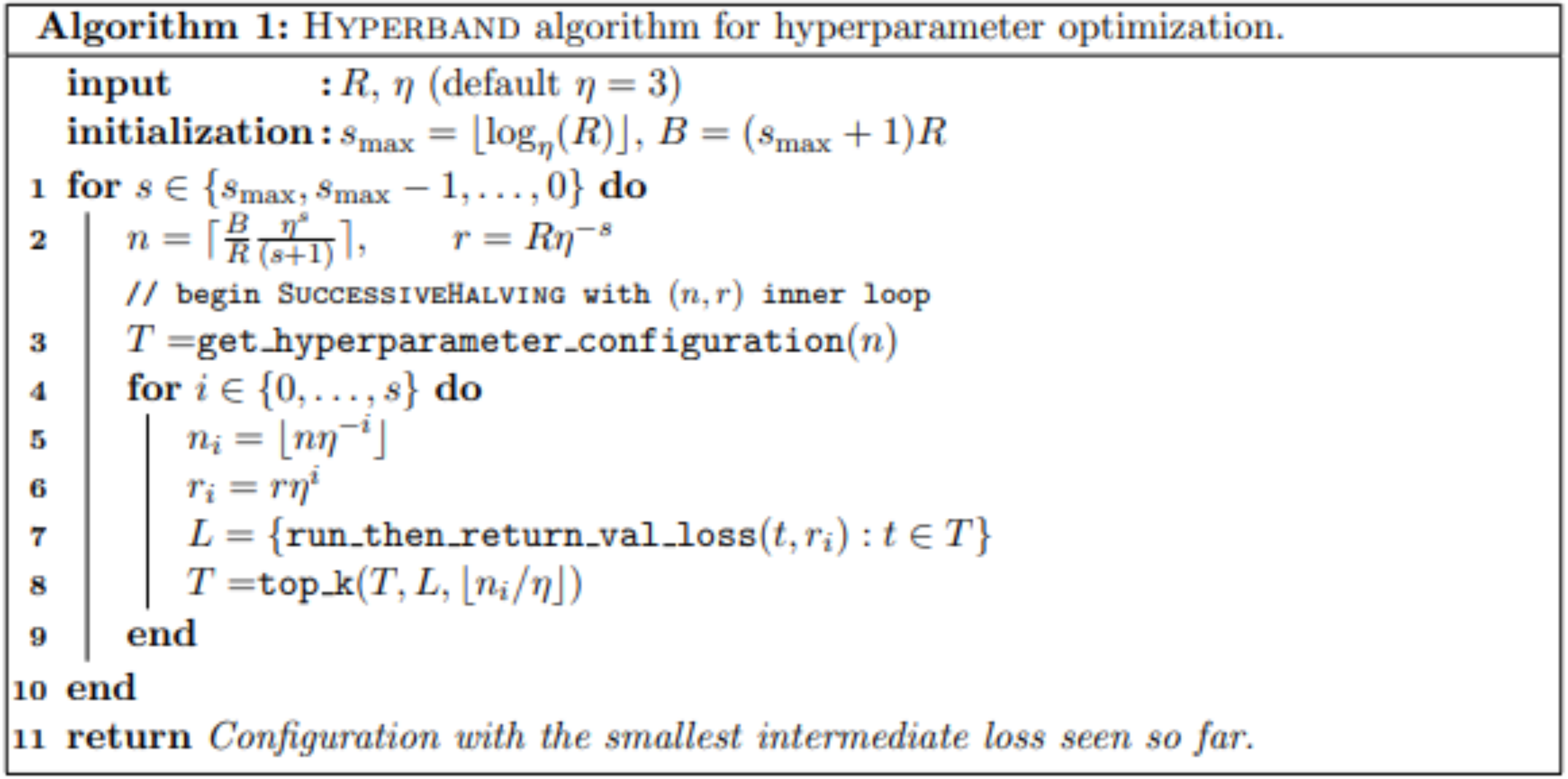
- Successive Halves require the user to choose n values.
  - Difficult to choose a n values.
- Each configuration uses ~B/n amount of resources.
- Small n: Longer computation but does not test a lot of configurations.
- Large n: Many configuration, small computation time.
  - May not be an accurate estimation



Hyperband

- Attempts to address the n vs B/n problem.
- Runs Successive Halving multiple times with different n values.
- Two inputs to hyperband :
  - No n value
  - R: Maximum amount resources that can be allocated to a single configuration
  - η: The proportion of configurations that are discarded in each round of SucessiveHalving
    - Higher η means greater discards.

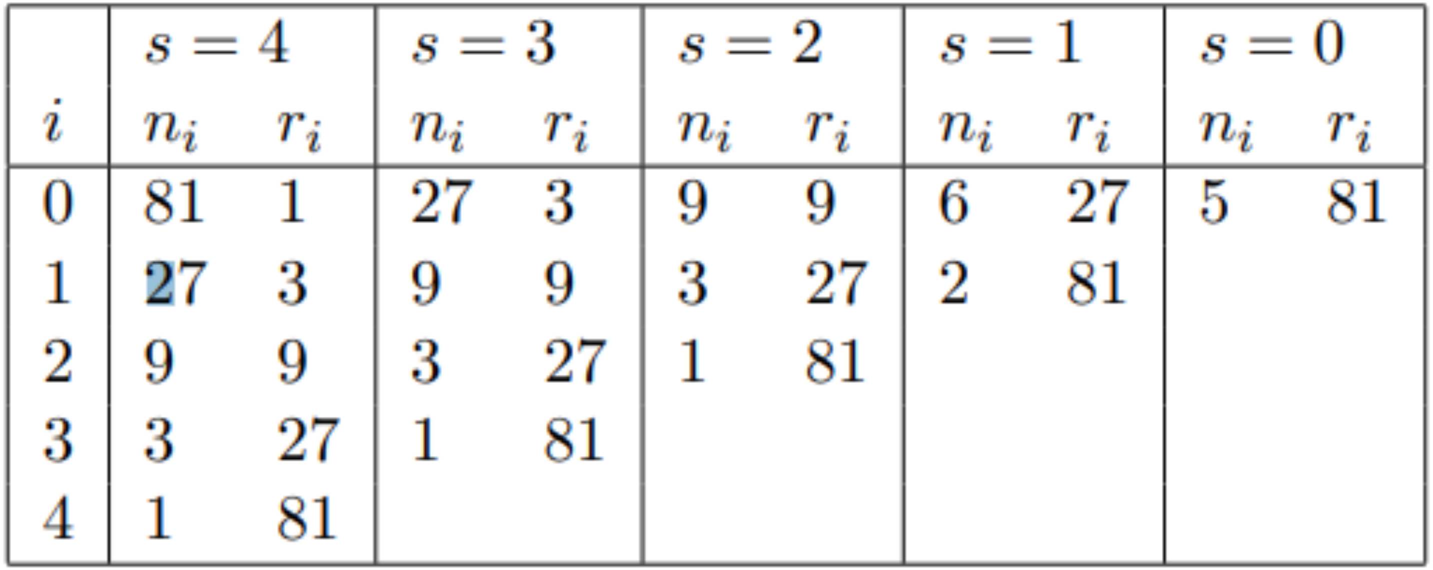
Algorithm



Algorithm 1: HYPERBAND algorithm for hyperparameter optimization.

```
input          : $R$, $\eta$ (default $\eta = 3$)
initialization : $s_{\max} = \lfloor \log_\eta(R) \rfloor$, $B = (s_{\max} + 1)R$
for $s \in \{s_{\max}, s_{\max} - 1, \ldots, 0\}$ do
    $n = \lceil \frac{B}{R} \frac{\eta^s}{(s+1)} \rceil$,        $r = R\eta^{-s}$
    // begin SUCCESSIVEHALVING with $(n, r)$ inner loop
    $T$ =get_hyperparameter_configuration($n$)
    for $i \in \{0, \ldots, s\}$ do
        $n_i = \lfloor n\eta^{-i} \rfloor$
        $r_i = r\eta^i$
        $L$ = {run_then_return_val_loss($t, r_i$) : $t \in T$}
        $T$ =top_k($T, L, \lfloor n_i/\eta \rfloor$)
    end
end
return Configuration with the smallest intermediate loss seen so far.
```

Example



	$s = 4$		$s = 3$		$s = 2$		$s = 1$		$s = 0$	
i	n_i	r_i	n_i	r_i	n_i	r_i	n_i	r_i	n_i	r_i
0	81	1	27	3	9	9	6	27	5	81
1	27	3	9	9	3	27	2	81		
2	9	9	3	27	1	81				
3	3	27	1	81						
4	1	81								

The values of ni and ri for the brackets of Hyperband corresponding to various values of s, when R = 81 and η = 3.

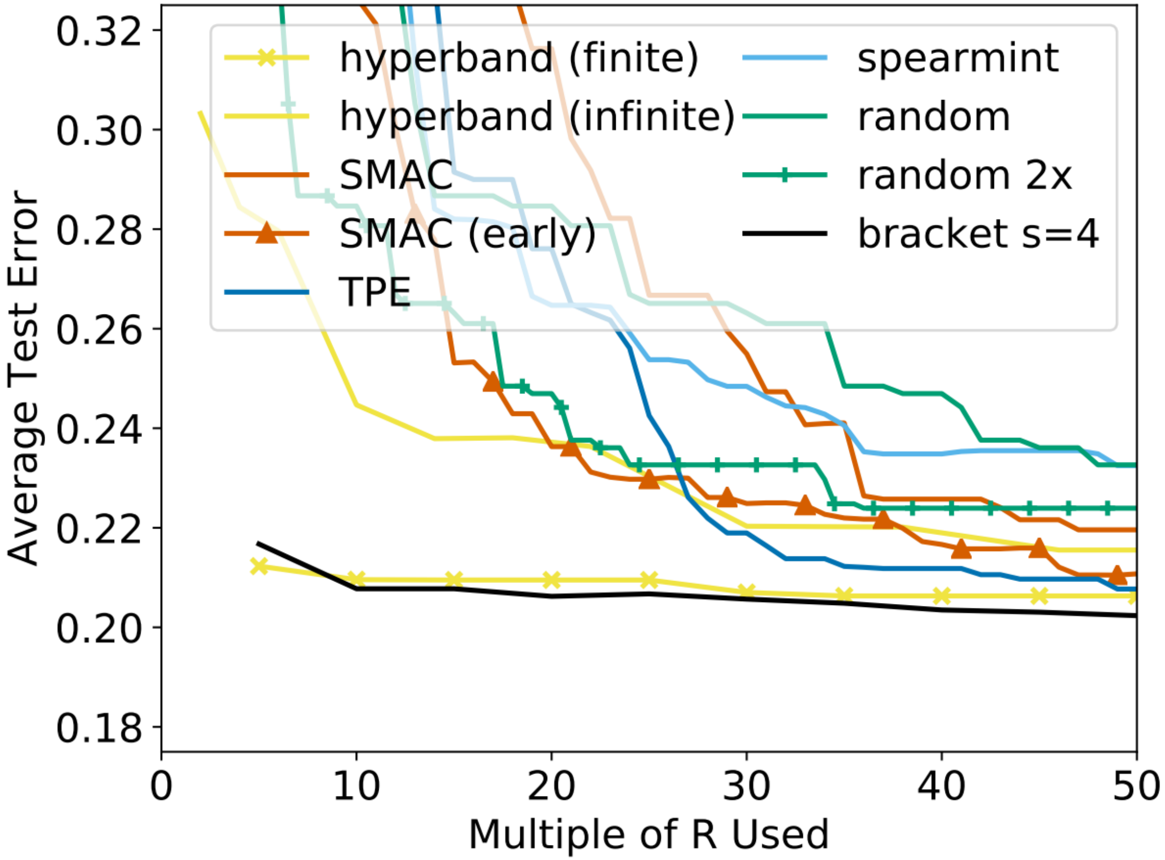
Inputs

- Setting R values
  - Smaller R will give a result faster
  - Larger R will give a better guarantee of successfully differentiating between the configuration
  - Infinite Horizon Hyperband if R value is unknown
    - This version doubles the budget overtime.
- Setting η values
  - Results are not sensitive to the choice of η
  - Theoretical bounds recommend η value of e
    - Practically 3 or 4.

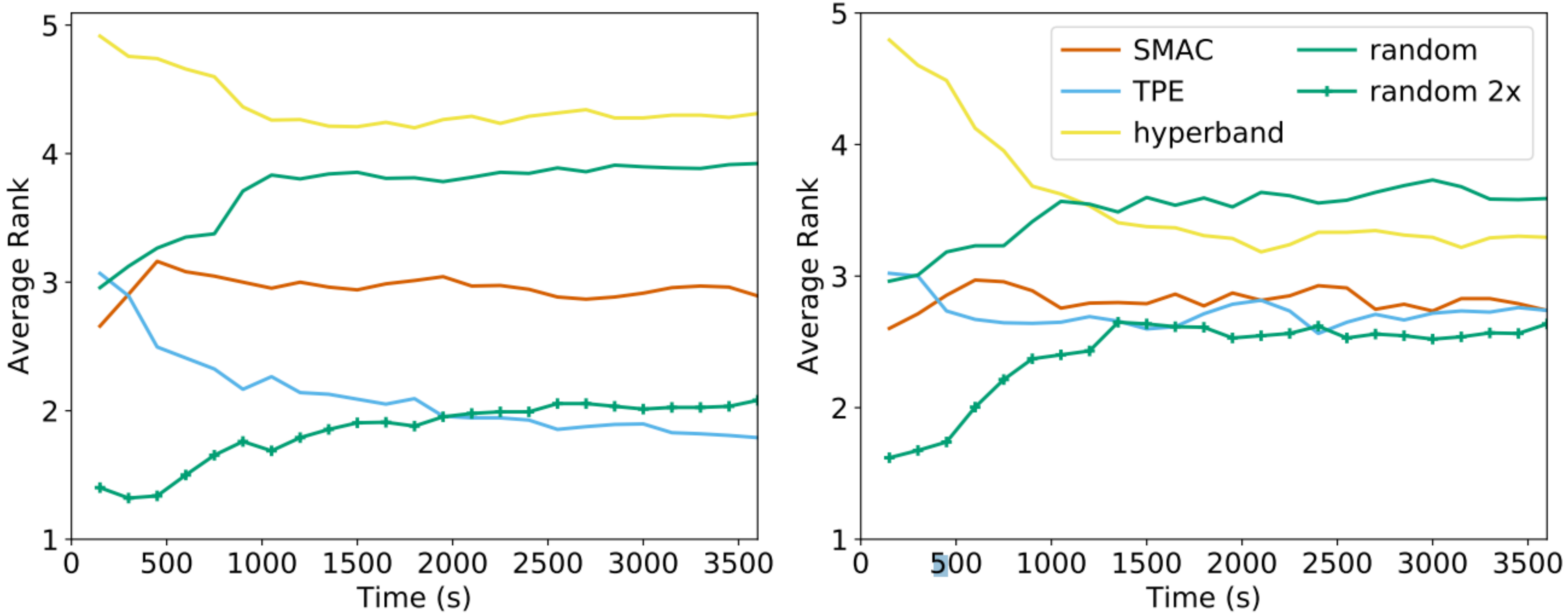
Results

- Will evaluate three different resource type:
  - Iterations
  - Data set Subsamples
  - Feature Samples
- Will compare against:
  - Bayesian Optimization algorithms
  - Random search
- Reasonable R values, but large enough for early stopping.
- η should depend on R and be selected to yield ≈ 5 runs with a minimum of 3 runs of SucessiveHalving.

Early stopping iterative algorithms



Dataset subsampling



(a) Validation Error on 117 Data Sets

(b) Test Error on 117 Data Sets

Comparisons to SucessiveHalving

- The most aggressive bracket of SucessiveHalving outperformed Hyperband
  - Was unknown that it would perform better.
- However, may yield different results with different problems
  - High pruning, but startup costs.
  - Important to have prior knowledge, even with Hyperband
    - Can limit the brackets that are explored, yielding in better results

Piazza Discussions

- 1. Do we really not have much better than brute force for hyper parameter tuning?

- 2.Why does bayesian optimization not outperform random search in higher dimensions?

- 3.There are only two inputs needed for HyperBand. What’s the best/a good way to think about them as an end user?

- 4.Page 11, 3.6. Why is 5 a reasonable number of brackets to explore for most problems?