Mango: A Python Library for Parallel Hyperparameter Tuning

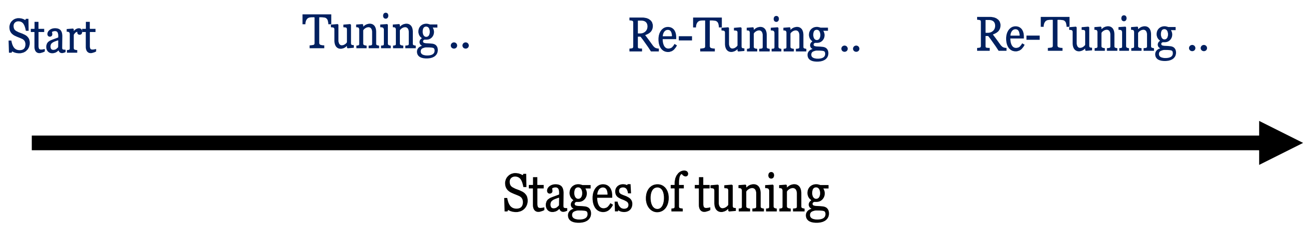
Sandeep Singh Sandha[1], Mohit Aggarwal[2], Igor Fedorov[2], Mani Srivastava[1]

[1] University of California, Los Angeles

[2] Arm Research

https://github.com/ARM-software/mango

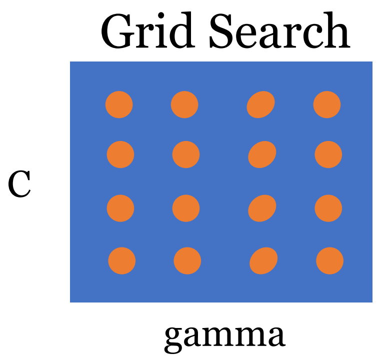
Tuning Hyperparameters

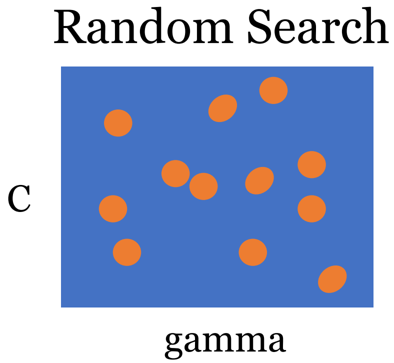


Approaches: Random search, Grid search, Bayesian optimization, Gradient-based, Evolutionary and Population.

ML algorithms are often very sensitive to choice of hyperparameters.

Example: SVM (C and gamma parameters)





Tuning Hyperparameters

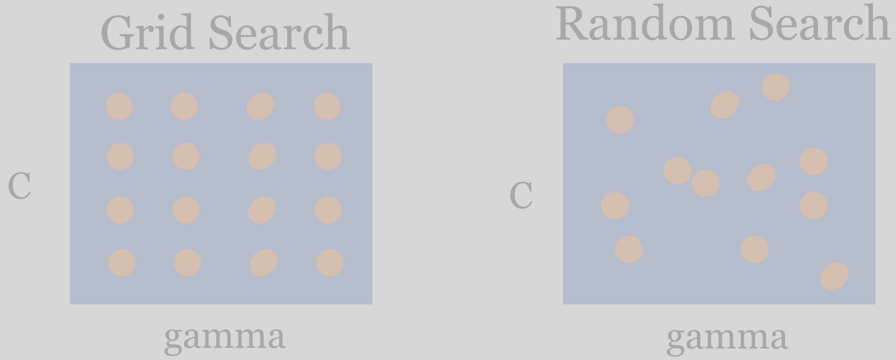
Start   Tuning ..   Re-Tuning ..   Re-Tuning ..

Approaches: Random search, Grid search, Bayesian optimization, Gradient-based, Evolutionary and Population.

Hyperparameter tuning is a tedious task and may involve a significant amount of software engineering.

ML algorithms are often very sensitive to choice of hyperparameters

Example: SVM (C and gamma parameters)



Hyperparameters: An Example

- SVM classifier for Iris Dataset
  - 150 examples, used for training
  - 50 examples used for validation
  - Used 2 features out of 4

- Hyperparameters:
  - kernel: rbf
  - gamma: [0.1 , 4]
  - C: [10^{-6} , 10^{8}]

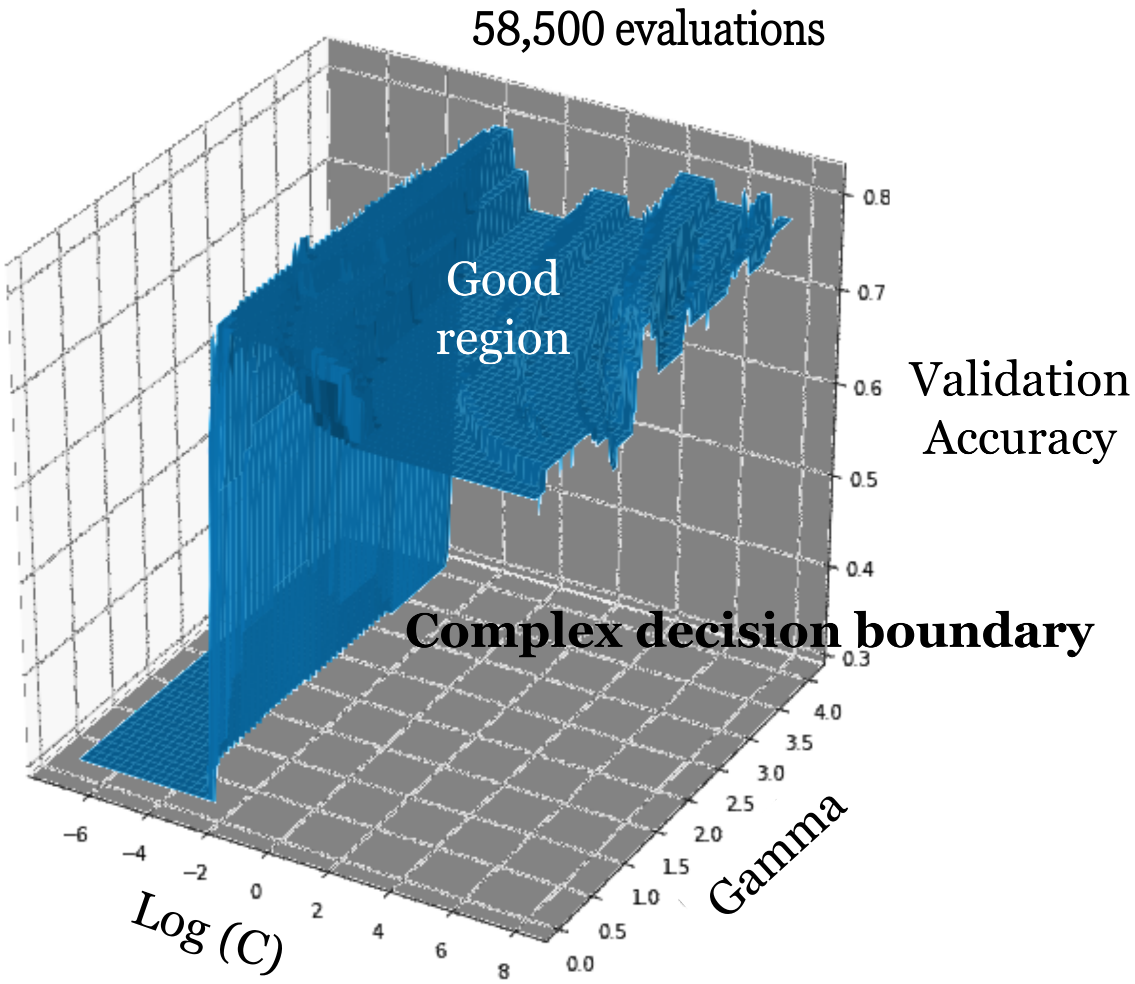
- **Best accuracy: ?**

Hyperparameters: An Example

- SVM classifier for Iris Dataset
  - 150 examples, used for training
  - 50 examples used for validation
  - Used 2 features out of 4

- Hyperparameters:
  - kernel: rbf
  - gamma: [0.1 , 4]
  - C: [10^{-6} , 10^{8}]

- **Best accuracy: 82%**



Hyperparameters: An Example

XGBoost classifier parameter search space: ~1 million with coarse grids.

```
param_space = {
     'n_estimators': range(10, 2001, 100),
     'max_depth': range(1, 15),
     'reg_alpha': loguniform(-3, 6),
     'booster': ['gbtree', 'gblinear'],
     'colsample_bylevel': uniform(0.05, 0.95),
     'colsample_bytree': uniform(0.05, 0.95),
     'learning_rate': loguniform(-3, 3),
     'reg_lambda': loguniform(-3, 6),
     'min_child_weight': loguniform(0, 2),
     'subsample': uniform(0.1, 0.89)
}
```

Hyperparameters: An Example

XGBoost classifier parameter search space: ~1 million with coarse grids.

```
param_space = {
    'n_estimators': range(10, 2001, 100),
    'colsample_bylevel': uniform(0.05, 0.95),
    'colsample_bytree': uniform(0.05, 0.95),
    'learning_rate': loguniform(-3, 3),
    'reg_lambda': loguniform(-3, 6),
    'min_child_weight': loguniform(0, 2),
    'subsample': uniform(0.1, 0.89)
}
```

Grid Search may not be optimal and often is very time consuming.

Does random search works?

What is Mango?

- Mango
  - Intelligent parallel search algorithms
  - Modular design: Ease of usability
  - Abstracts are compatible with Sklearn

- We Focus on classical ML algorithms
  - Designed for a production cluster

- Why Mango?
  - Existing widely used libraries: Hyperopt, Auto-sklearn, Auto-WEKA
  - Existing libraries were not designed for production cluster
  - Many have **missing feature**: Fault tolerance, Job scheduling challenges, Parallel search, Compatibility.
  - Assumes specific kind of platform/workers are running or are totally manual.

[1] James Bergstra, Dan Yamins, and David D Cox, "Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms," in Proceedings of the 12th Python in science conference. Citeseer, 2013
[2] Chris Thornton, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown, "Auto-weka: Combined selec- tion and hyperparameter optimization of classification algorithms," in Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, 2013
[3]. Auto-Sklearn, "Manual parallel spawing," https://automl.github.io/auto- sklearn/master/manual.html

What is Mango?

- Implementation of state-of-the-art parallel search optimizers
- Available opensource
- Continually tested on production scale datasets and is improved

https://github.com/ARM-software/mango

Mango: Simple Example

```python
1  from mango import scheduler, Tuner
2
3  param_space = dict(x=range(-10,10))
4
5  @scheduler.serial
6  def objective(x):
7      return x * x
8
9  tuner = Tuner(param_space, objective)
10 results = tuner.minimize()
11
12 print(results["best_params"],
13       results["best_objective"])
```

Search Space: 3

Objective Function: 5-7

Run Mango Tuner: 9-10

Results: 12-13

Mango Abstractions: Optimizer

- Multi-armed bandit bayesian optimizer
- Surrogate function: Gaussian Process **(Internal approximator)**
- Acquisition function: Created from Surrogate function **(Good regions to explore)**

Mango Abstractions: Optimizer

- Multi-armed bandit bayesian optimizer
- Surrogate function: Gaussian Process **(Internal approximator)**
- Acquisition function: Created from Surrogate function **(Good regions to explore)**

Simplified form of optimizer:

1. Sample the parameter search space using Monte Carlo.
2. Find a batch of next promising hyperparameter to evaluate.
   1. Maximize the acquisition function (exploration vs. exploitation).
   2. Implemented two batch selection approaches: **hallucination**, **clustering**.
3. Update the acquisition function based on results.

[1] Desautels, Thomas, et al. "Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization." JMLR 2014

Mango Abstractions: Optimizer

- Multi-armed bandit bayesian optimizer
- Surrogate
- Acquisitio

Evaluate promising regions in parallel

Simplified form:

Adaptive sampling

1. Sample the parameter search space using Monte Carlo.
2. Find a batch ... valuate.

Adaptive exploration

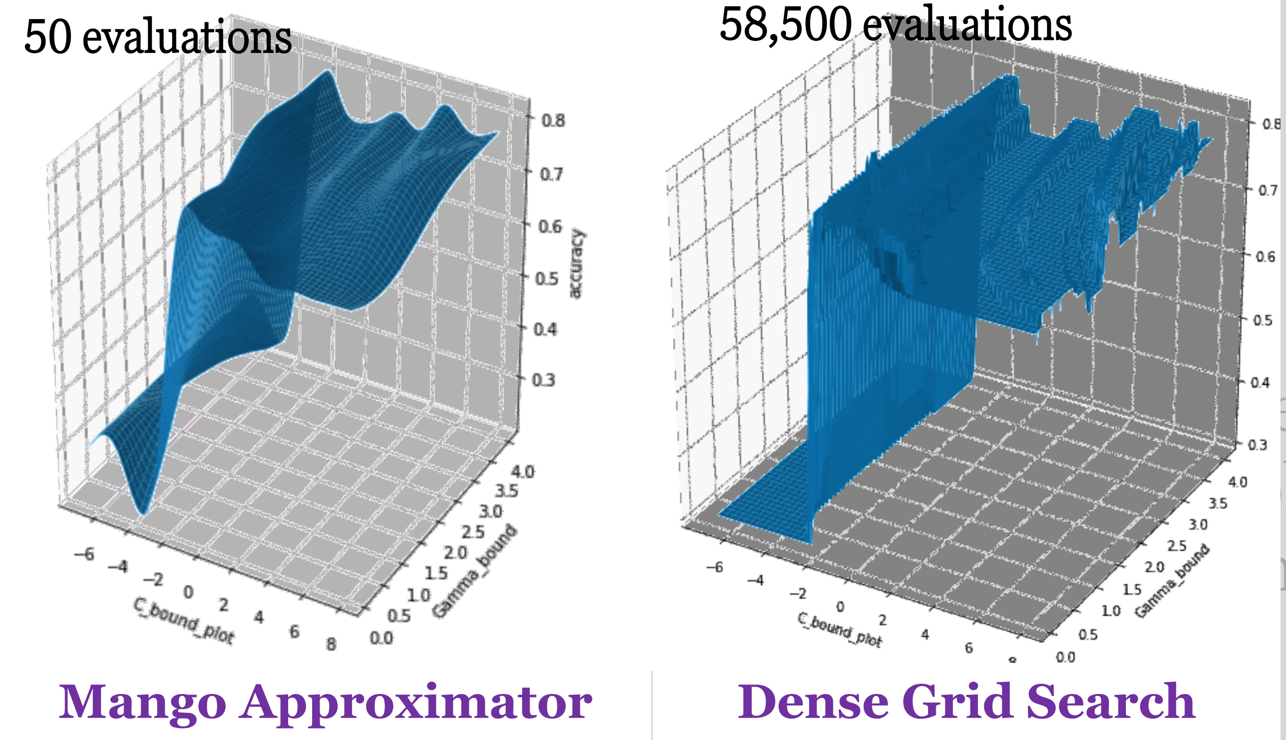
   1. Maximiz ... vs. exploitation)
   2. Implemented two batch selection approaches: hallucination, clustering.
3. Update the a

Fault tolerance

[1] Desautels, Thomas, et al. "Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization." JMLR 2014

Mango Abstractions: Optimizer

- Multi-armed bandit bayesian optimizer
- Surrogate fu
- Acquisition



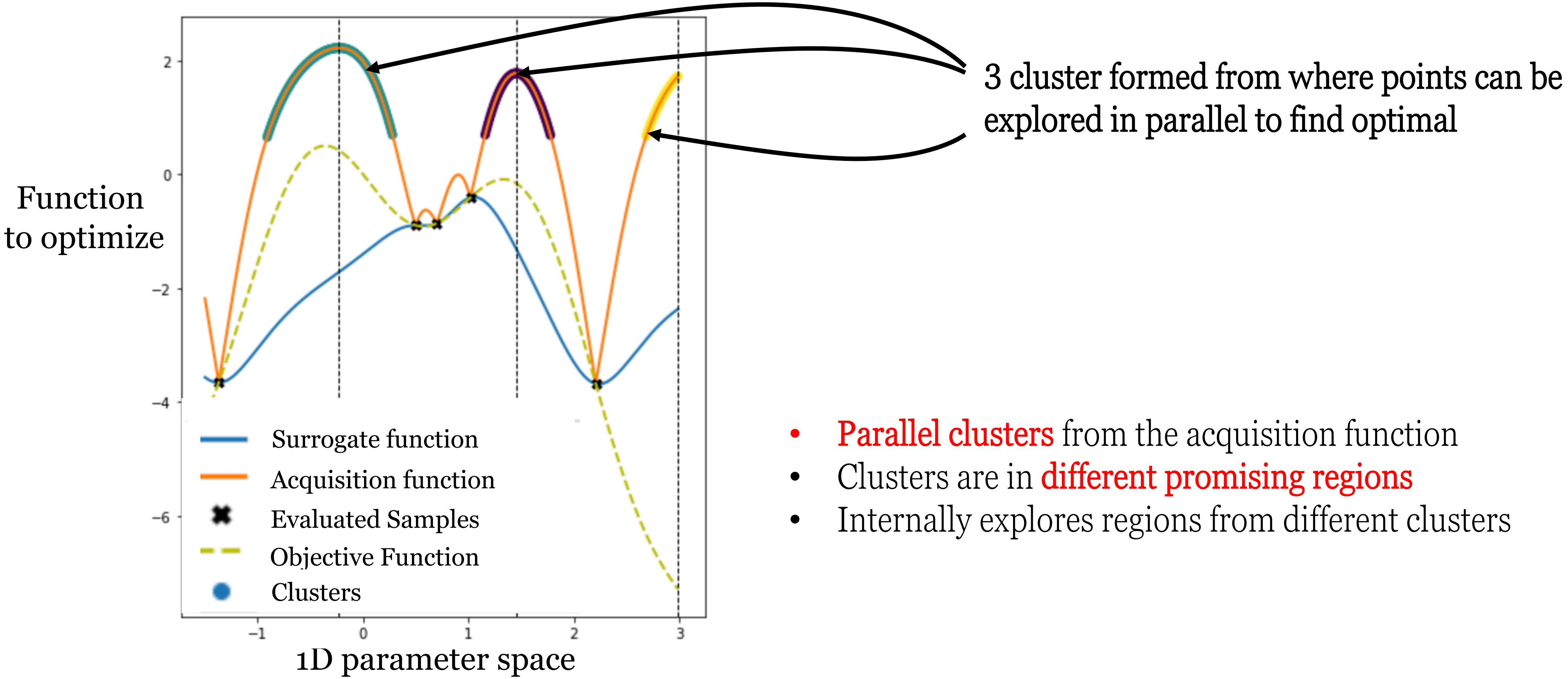
Simplified f

1. Sample
2. Find a b e.
    1. Max loitation)
    2. Imp nation, clustering.
3. Update

[1] Desautels, Thomas, et al. "Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization." JMLR 2014

Mango Abstractions: Optimizer

Visualizing clustering approach



- Parallel clusters from the acquisition function
- Clusters are in different promising regions
- Internally explores regions from different clusters

Mango Abstractions: Search Space

- Example: XGBoost Classifier

```
param_space = {
    'n_estimators': range(10, 2001, 100),
    'max_depth': range(1, 15),
    'reg_alpha': loguniform(-3, 6),
    'booster': ['gbtree', 'gblinear'],
    'colsample_bylevel': uniform(0.05, 0.95),
    'colsample_bytree': uniform(0.05, 0.95),
    'learning_rate': loguniform(-3, 3),
    'reg_lambda': loguniform(-3, 6),
    'min_child_weight': loguniform(0, 2),
    'subsample': uniform(0.1, 0.89)
}
```

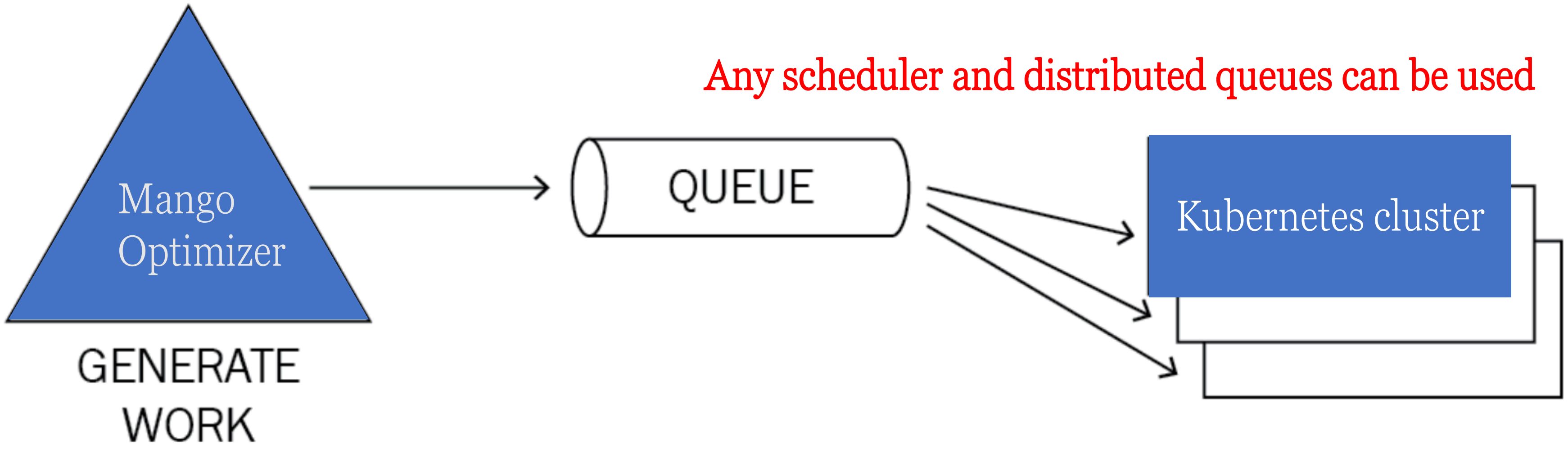
- Search space
  - We support all **70+ distributions of scipy.stats**
  - Python constructs lists, dicts, range.
  - **Compatible with sklearn** grid search
  - **Continuous**: distributions
  - **Discrete**: lists, strings
  - New distributions can be easily defined.

- Example: SVM Classifier

```
param_space = {
    'kernel': ['rbf', 'sigmoid'],
    'gamma': uniform(0.1, 4),
    'C': loguniform(-7, 8)
}
```

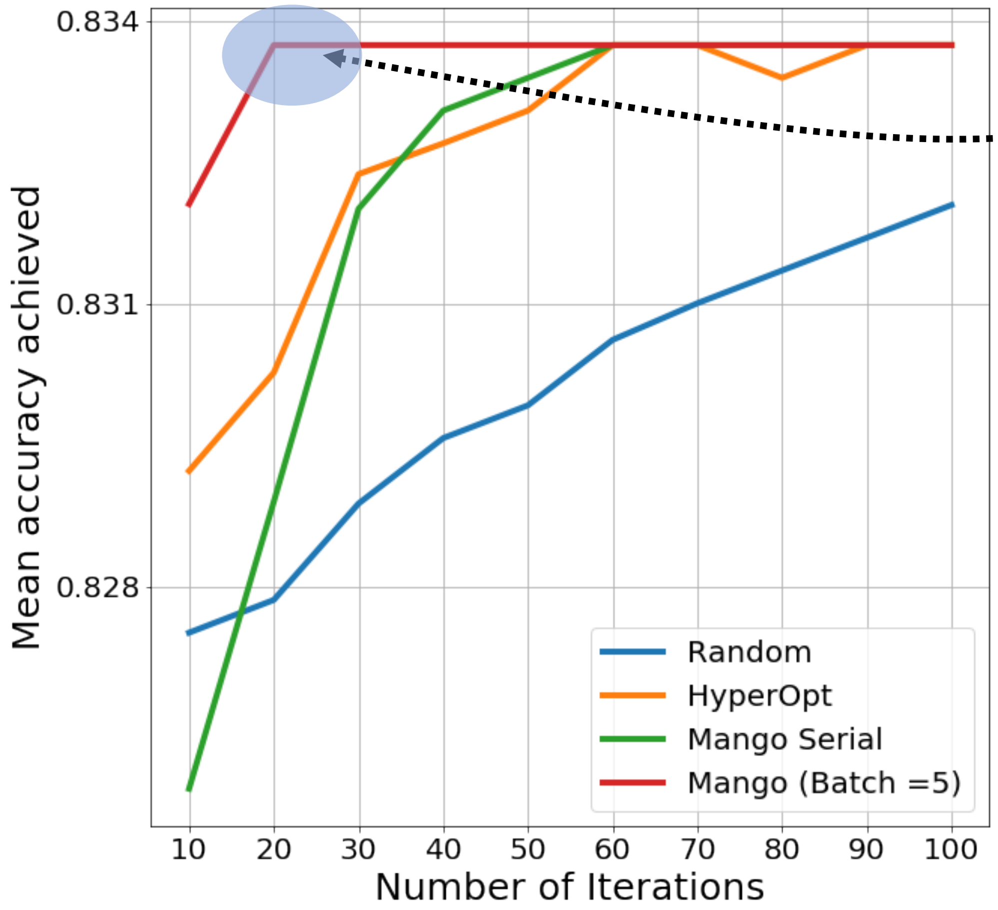
Mango Abstractions

- Objective Function & Scheduler
  - Objective function defined by user
  - Scheduler abstractions are kept separate from optimizer
- Objective Function
  - Objective function receives an hyperparameter to evaluate.
  - In case of failure, it can return empty results



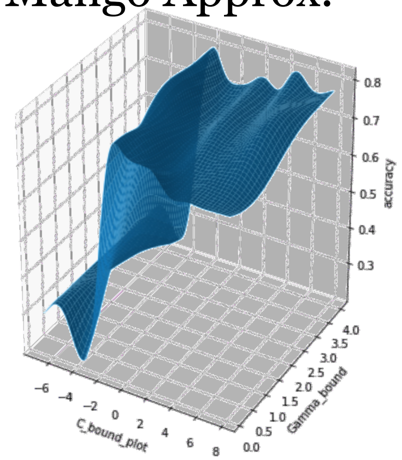
Celery workflow used in production

Mango Experiments



Batch Mango is much faster

Mango Approx.

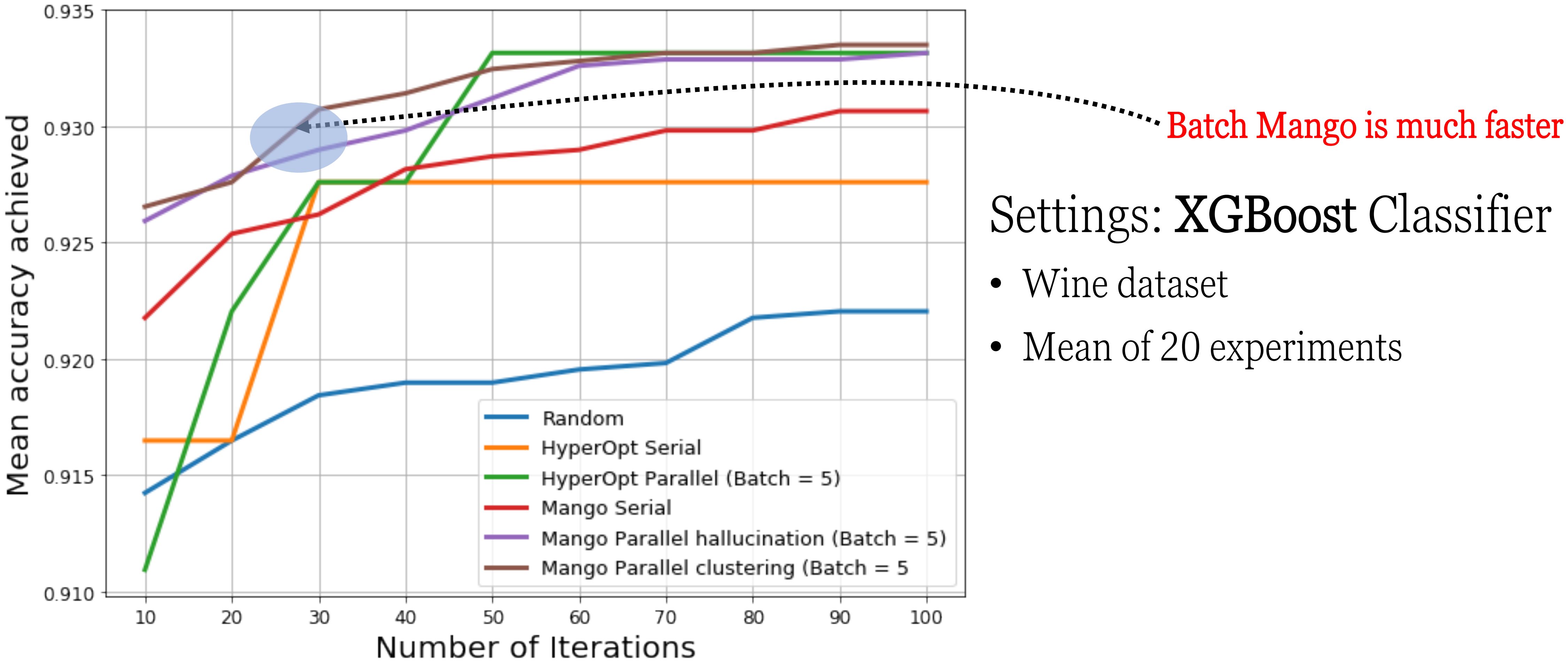


Settings: **SVM Classifier**

- Iris dataset
- Mean of 20 experiments
- HyperOpt: Another widely used library
- Mango in batch setting (hallucination)

https://github.com/ARM-software/mango/tree/master/examples/

Mango Experiments

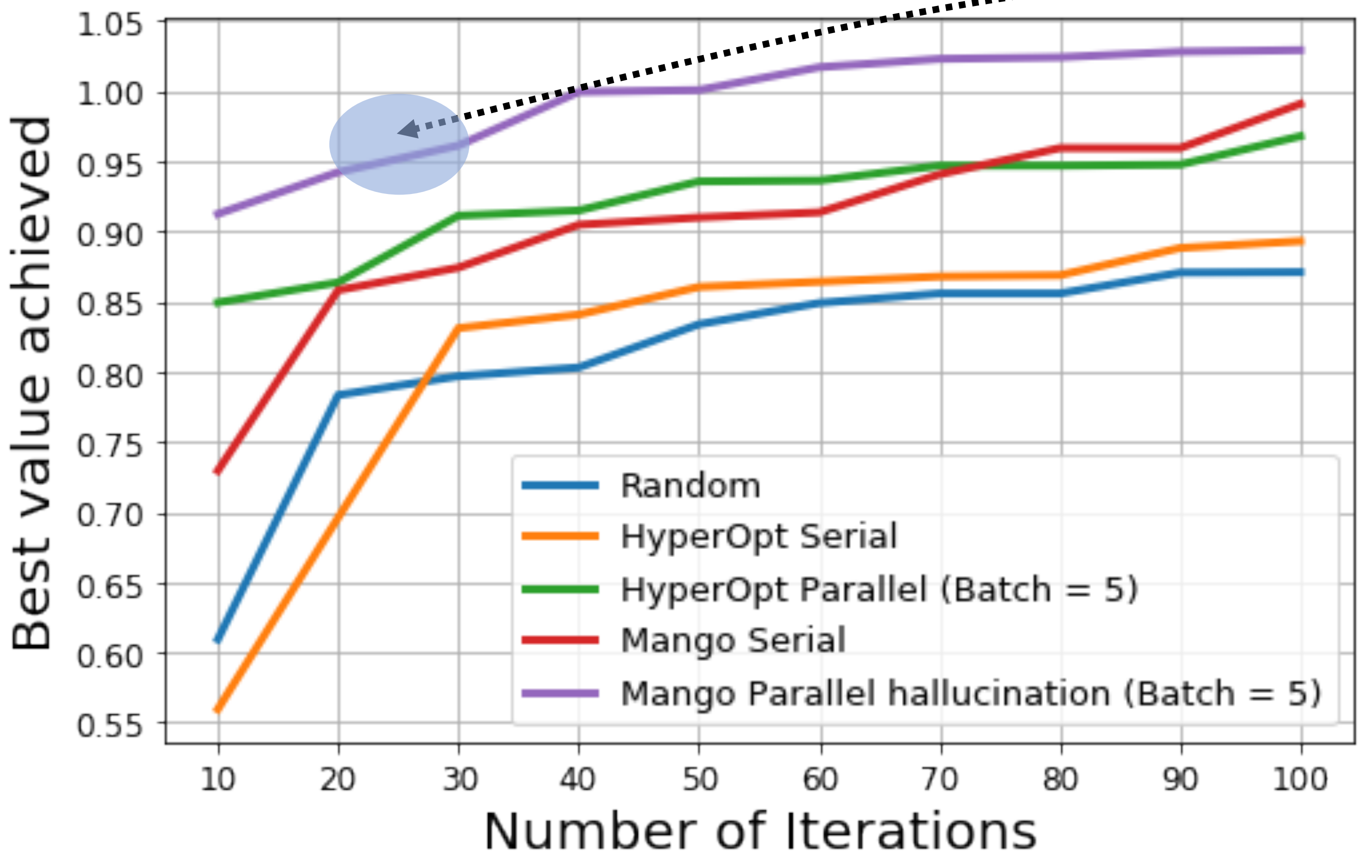


Settings: **XGBoost** Classifier

- Wine dataset
- Mean of 20 experiments

[1] Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016.
[2] https://github.com/ARM-software/mango/tree/master/examples/

Mango Experiments



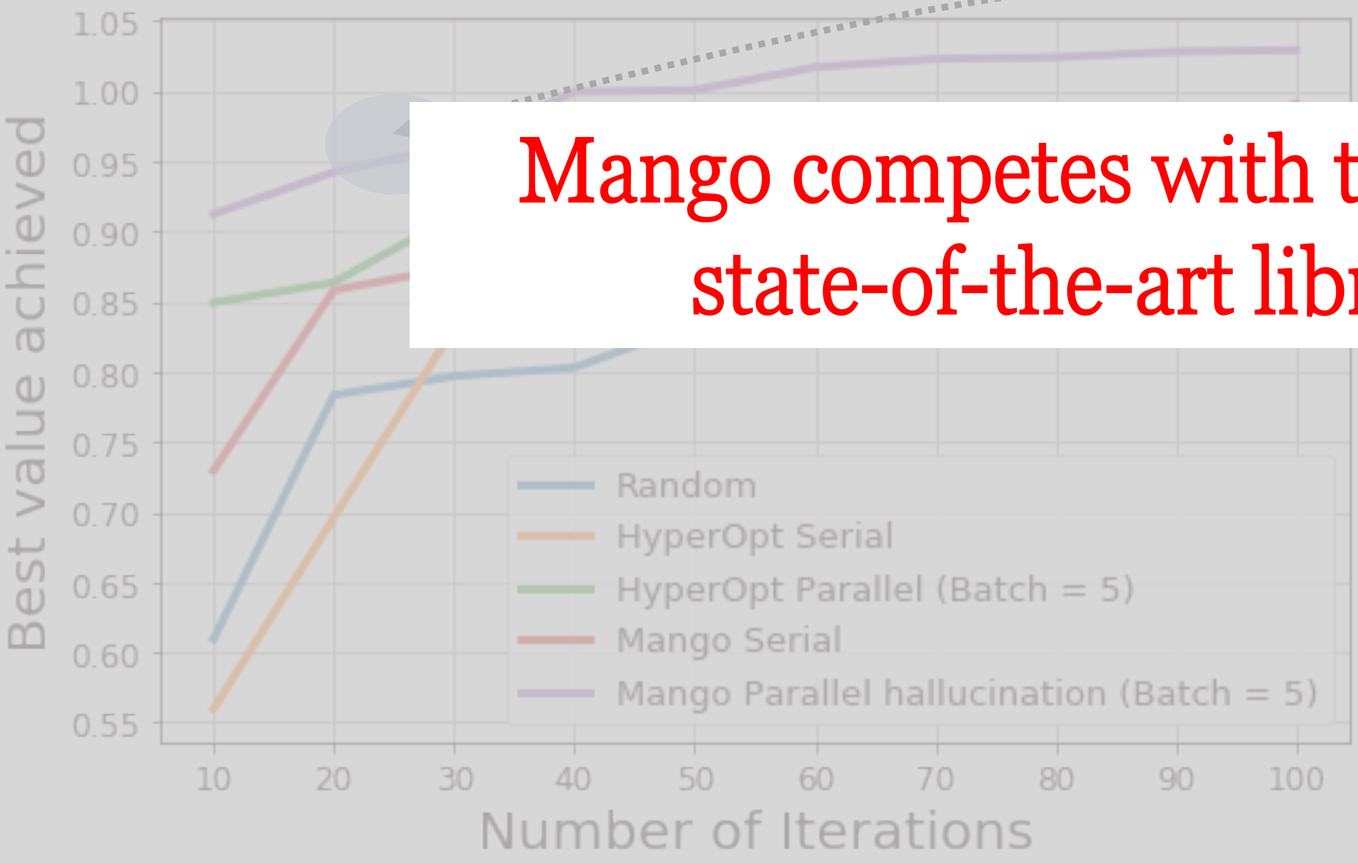
Batch Mango is much faster

Settings: **Branin function**

- Mixed discrete and continuous variables
- Mean of 20 experiments

https://github.com/ARM-software/mango/tree/master/examples/

Mango Experiments



Mango competes with the existing state-of-the-art libraries.

ngo is much faster

function

- Mixed discrete and continuous variables
- Mean of 20 experiments

https://github.com/ARM-software/mango/tree/master/examples/

Conclusion

- Parallel search speeds up the task of hyperparameter tuning.
- Mango achieves comparable performance to existing approaches:
  - Provide rich abstractions for search space.
  - Intelligent parallel search algorithms.
  - Flexibility to the choice of scheduler.
  - Designed with the ease of usability.

https://github.com/ARM-software/mango