Hyppo: Efficient Discovery and Execution of Data Science Pipelines in Collaborative Environments

A. Kontaxakis, D. Sacharidis, A. Simitsis, A. Abelló, S. Nadal

ULB UNIVERSITÉ LIBRE DE BRUXELLES

ATHENA Research & Innovation Information Technologies

UPC UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Summary. This paper presents ***HYPPO***, a novel system to optimize and discover pipelines encountered in exploratory machine learning. HYPPO keeps previously executed pipelines in a History graph to facilitate ***discover*** and exploits ***sharing***, ***reuse*** and ***equivalence*** among tasks to optimize the execution. A thorough experimental evaluation shows that ***HYPPO*** results in plans that are typically **one order (up to two orders)** of magnitude faster and cheaper pipeline.

Discovery and Optimization

(1) Discovery

```
1 hyppo = Hyppo("catalog_path")
2 dataset = "HIGGS"
3 hyppo.best_metrics(dataset, num=3)
```

```
metric  low_score  high_score
  'F1'       0.58        0.88
   'P'       0.40        0.72
   'R'       0.61        0.90
```

```
4 hyppo.retrieve_best_pipeline(dataset, "F1", num=1)
```

```
[('scaler', StandardScaler()),
 ('pca', PCA(n_components=3)),
 ('svm', SupportVectorMachine()),
 ('F1', F1ScoreCalculator())]
```

```
5 hyppo.popular_operators(dataset, "Classifier")
```

```
[(42, 'rfc'), (23, 'svm'), (4, 'ridge')]
```

```
6 hyppo.view_dictionary('rfc')
```

```
{'rfc': ['sk.ensemble.RandomForestClassifier',
         'tfdf.keras.RandomForestModel',
         'cuml.ensemble.RandomForestClassifier']}
```

(2) Optimization

```
7 pipe = Pipeline([('scaler', StandardScaler()), ('pca',
      PCA(n_components=3)), ('rfc', cuml.ensemble.
      RandomForestClassifier()), ('F1',
      F1ScoreCalculator())], dataset)
8 pipe.run()
```

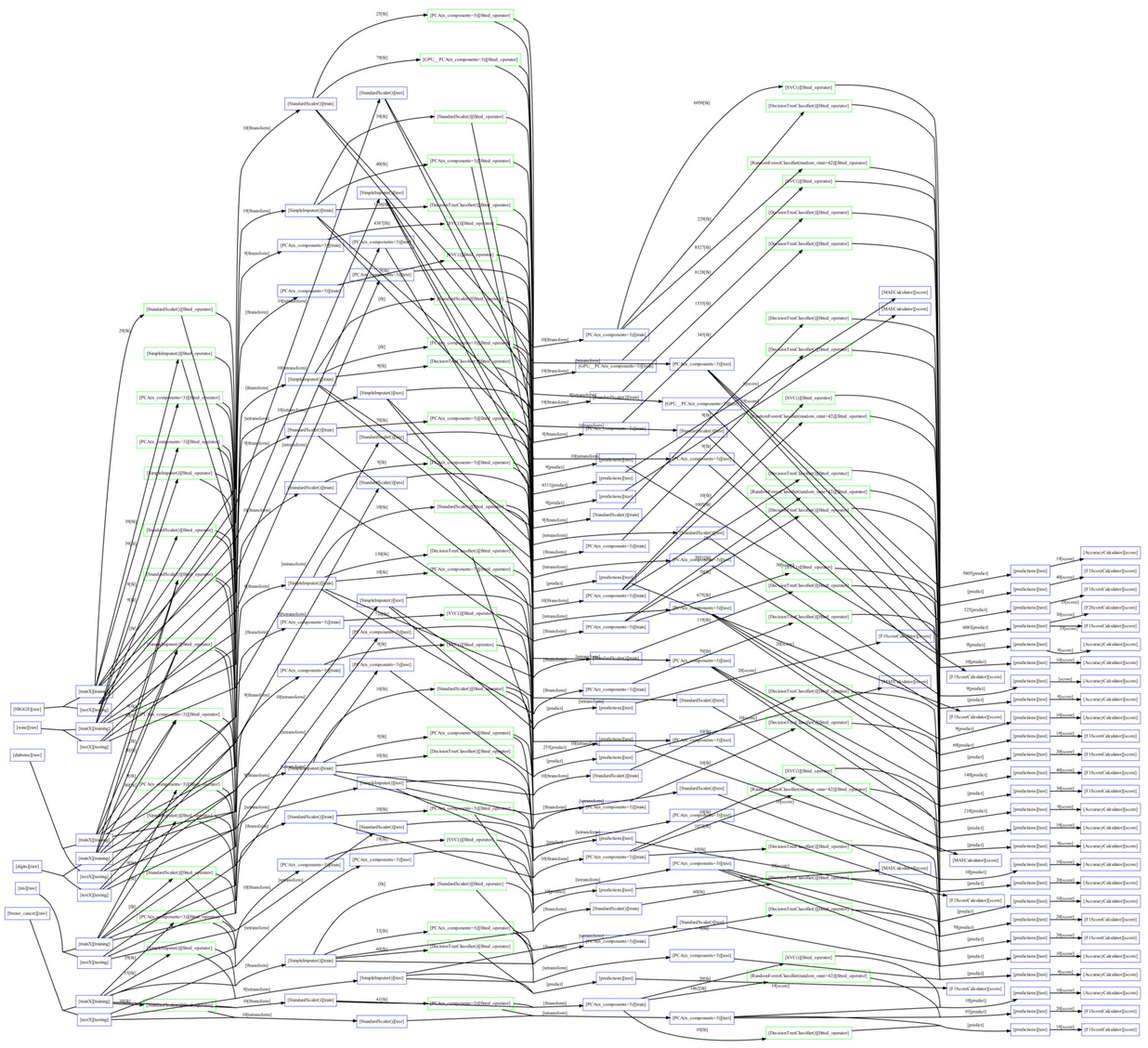
```
{'time': 1200ms, 'F1': 0.91}
```

```
9 opt_pipe = hyppo.optimal_plan(pipe)
10 opt_pipe.run()
```

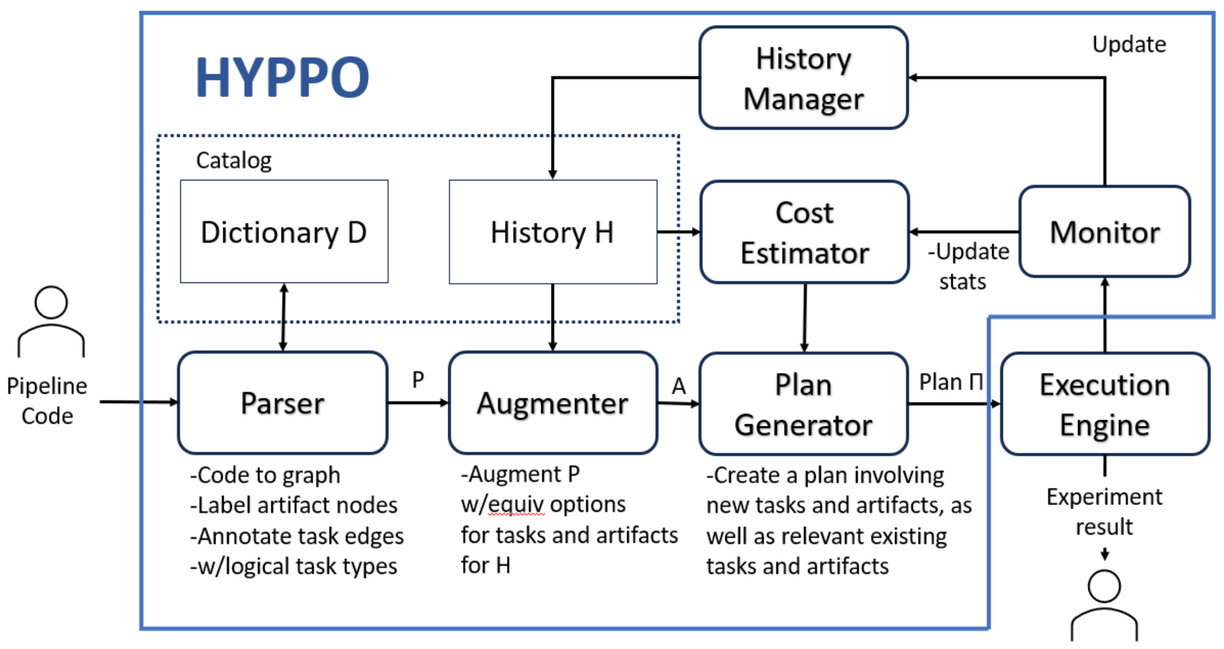
```
{'time': 700ms, 'F1': 0.91}
```

We allow users to:
Review evaluation metrics achieved on the specific dataset (Lines 2–3)
Retrieve the top-performing pipeline (Line 4).
Query Hyppo for popular operators of type Classifier that have been used in pipelines involving the HIGGS dataset (Line 5).
Look up the available implementations of rfc (Line 6).
Select an implementation a random forest classifier as the final step (Line 7).
Execute the pipeline as is (Line 8).
Optimize the pipeline by providing it as input to the optimizer of Hyppo, which returns an **optimal execution plan** (Line 9).
Executing that plan (Line 10) results in a considerable decrease in execution time.

History



System overview

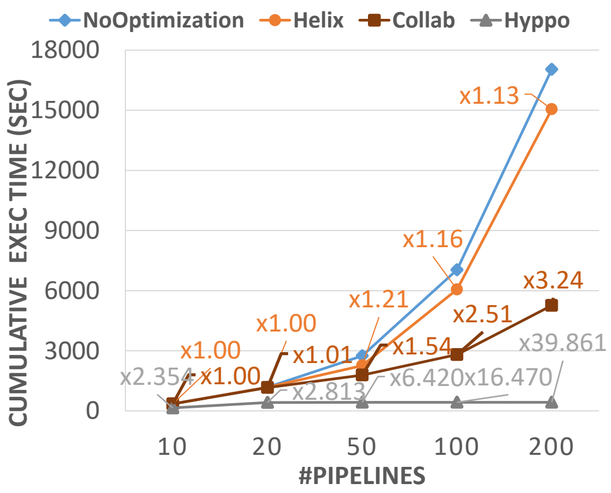


Evaluation

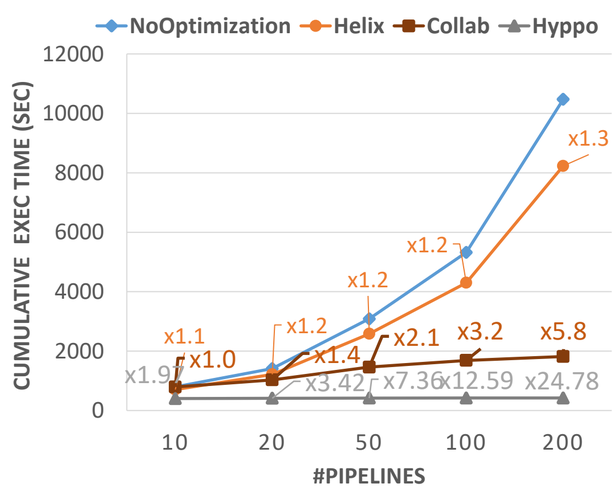
Method	None	Sharing	Reuse	Materialization	Equivalence
NoOptimization	○				
Sharing		○			
Helix [VLDB'18]		○	○	◗	
Collab [SIGMOD'20]		○	◗	○	
HYPPO[ICDE'24]		○	○	○	○

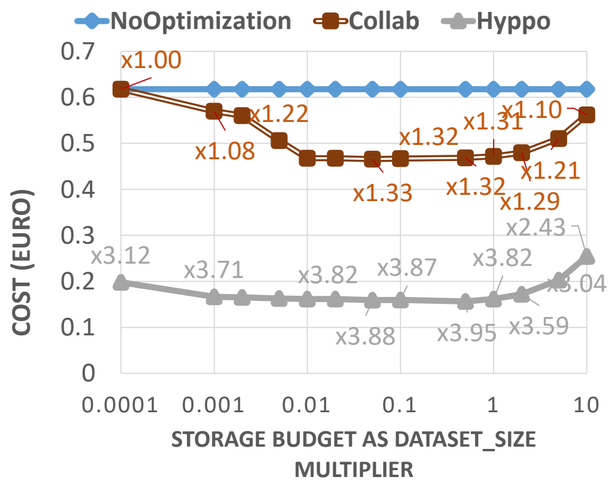
Iterative pipeline execution

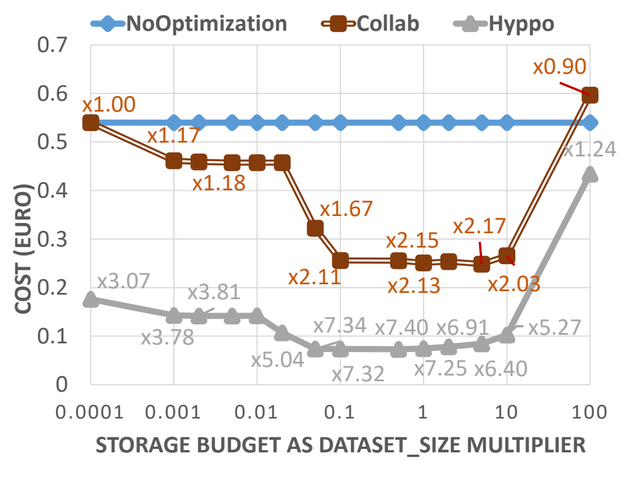
HIGGS



TAXI

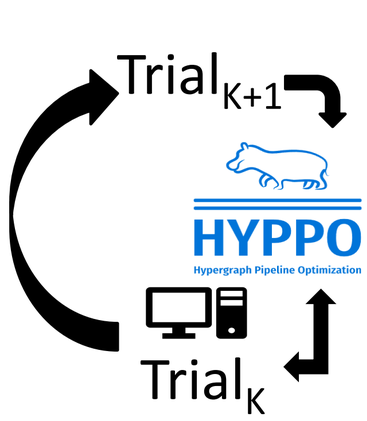






Given a pipeline code, HYPPO

a) searches for an **optimized execution plan**
b) decides what **artifacts to materialize**
c) results in plans **10x faster** and **cheaper** plans than SOTA methods



References

VLDB 18] D. Xin, S. Macke, L. Ma, J. Liu, S. Song, and A. Parameswaran, "HELIX: Holistic optimization for accelerating iterative machine learning"
[SIGMOD 20] B. Derakhshan, A. Rezaei Mahdiraji, Z. Abedjan, T. Rabl, and V. Markl, "Optimizing machine learning workloads in collaborative environments"
[ICDE'24] Antonios I. Kontaxakis, Dimitris Sacharidis, Alkis Simitsis, Alberto Abelló, and Sergi Nadal. "HYPPO: Using Equivalences to Optimize Pipelines in Exploratory Machine Learning"

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 955895.