INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING
ICPP/2021/CHICAGO/USA
acm In-Cooperation
sighpc
AUGUST 9–12, 2021

PREP: Predicting Job Runtime with Job Running Path on Supercomputers

Longfang Zhou[1], Xiaorong Zhang[1], Wenxiang Yang[2], Yongguo Han[1], Fang Wang[2], Yadong Wu[3], Jie Yu[2*]

[1]Southwest University of Science and Technology, Mianyang, Sichuan, China.

[2]China Aerodynamics Research and Development Center, Mianyang, Sichuan, China.

[3]Sichuan University of Science and Engineering, Zigong, Sichuan, China.

E-mail: longfang_zhou@163.com, 29102239@qq.com, yangwenxiang10g@nudt.edu.cn, hansir@swust.edu.cn, wangfangcardc@163.com, wyd028@163.com, yujie@nudt.edu.cn.

Contents

- Problem
  - Background
  - Objective
- Motivation
  - Job Running Path
  - Clustering Path
- PREP Framework
  - Overview
  - Data Pre-processing
  - Runtime Prediction
- Experiment and Evaluation

Problem

- Background
  - Backfilling is to select a queuing job with estimated runtime and make it run ahead. However, users often overestimate the runtime.
  - Machine learning lacks enough features to describe the characteristic information of the job.

- Objective
  - We explore new features to add information describing the characteristic of job.
  - We apply the feature to runtime prediction to improve prediction accuracy.

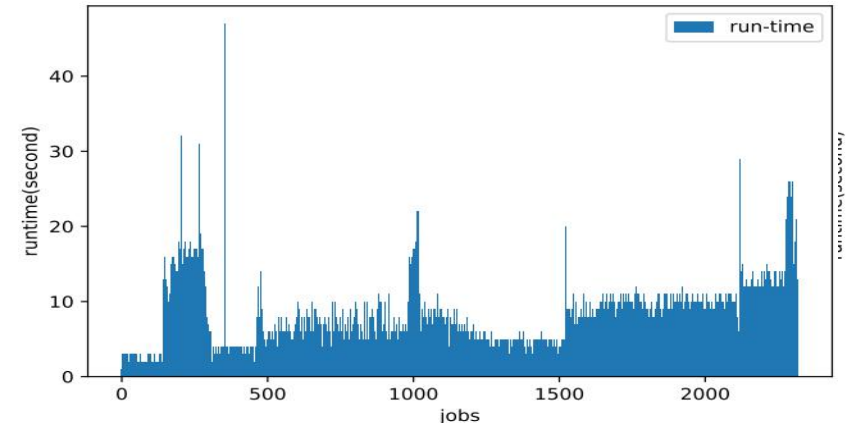
Motivation

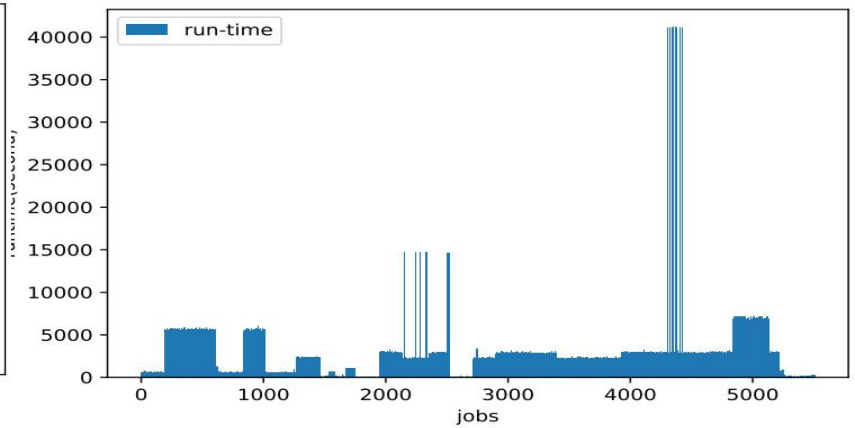
- Job Running Path
  - Definition: The path refers to the path where the user submits the jobs.
  - The path includes abundant semantic information, such as the project of jobs, data sets and parameters, etc.

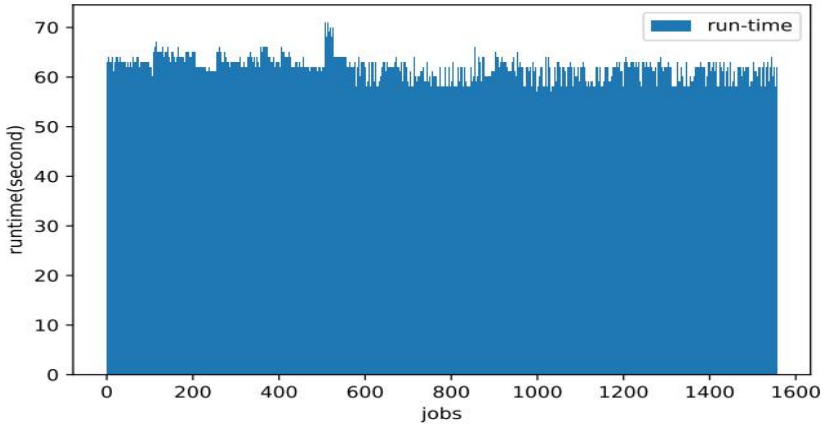
	Example of Job Running Path
1	/home/Alisa/fairness/88reward/12_4_8_6procssors-a/goal_545/time_plan/strategy_version82
2	/home/Bob/colleague/2018/178/teamwork/FastRun/accurate/effctive/a5b10_3meter_math_l2.9
3	/home/Carole/Car/2department/tyre-significant-20181123/a532-32open/konwledge/luminosity/graceful/a09b12
4	/home/Devere/Language/12effetive28optimistic_12/1.2-upper1/20/illustration/peaceful/Jordan/Tower
5	/home/Eros/symmetry/2018labor/97/100-901/100-991-refine/improvement/hole76/a95b45

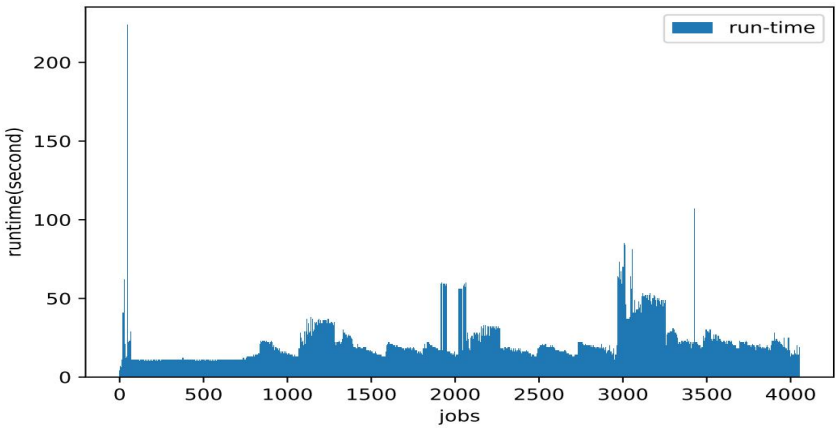
Motivation

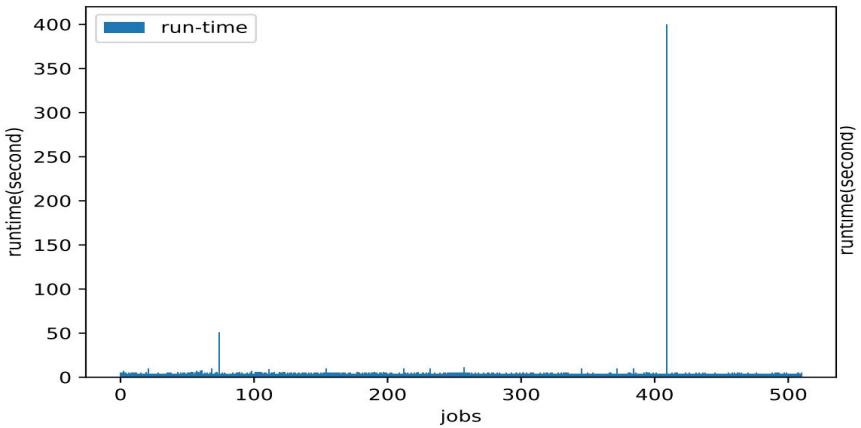
- Job Running Path
  - Jobs with the same paths have similar runtime.

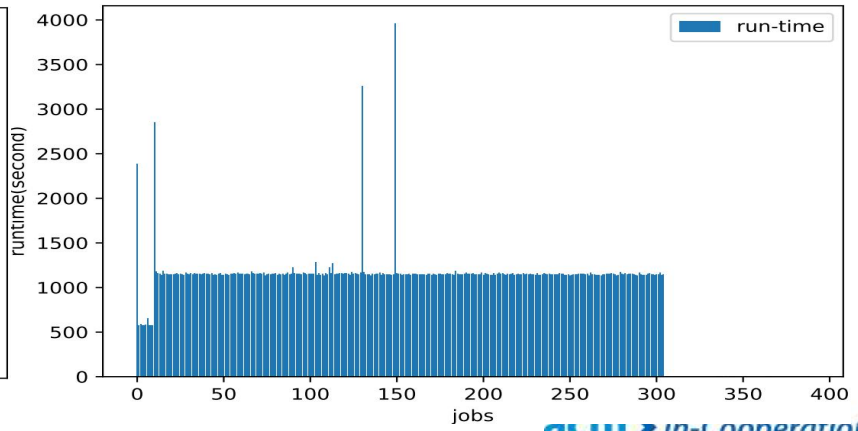










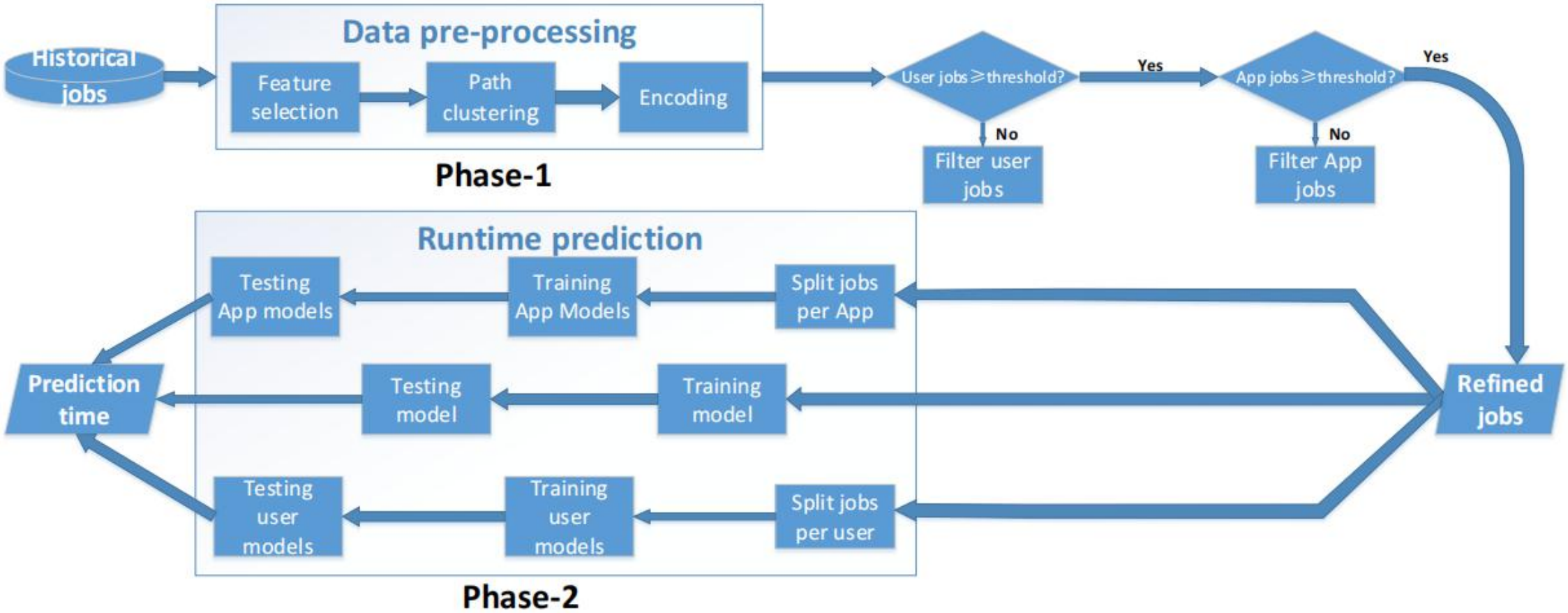


Motivation

- Clustering Path
  - To improve the quality of data, we group all jobs with similar path**, and one application contains all jobs with similar path.**
  - We make runtime prediction by adding the new feature into traditional prediction model.
  - We make prediction based on the weighted average of runtime prediction of multiple applications.

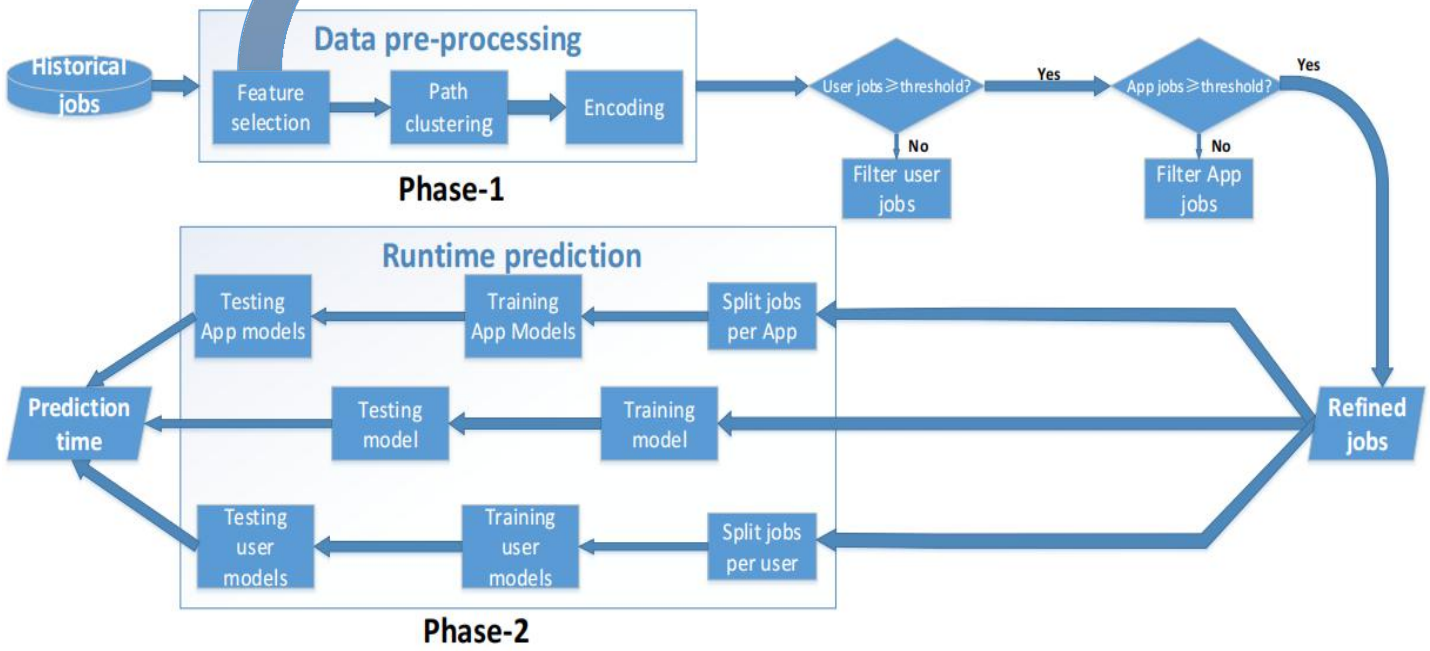
PREP Framework

- Overview



PREP Framework

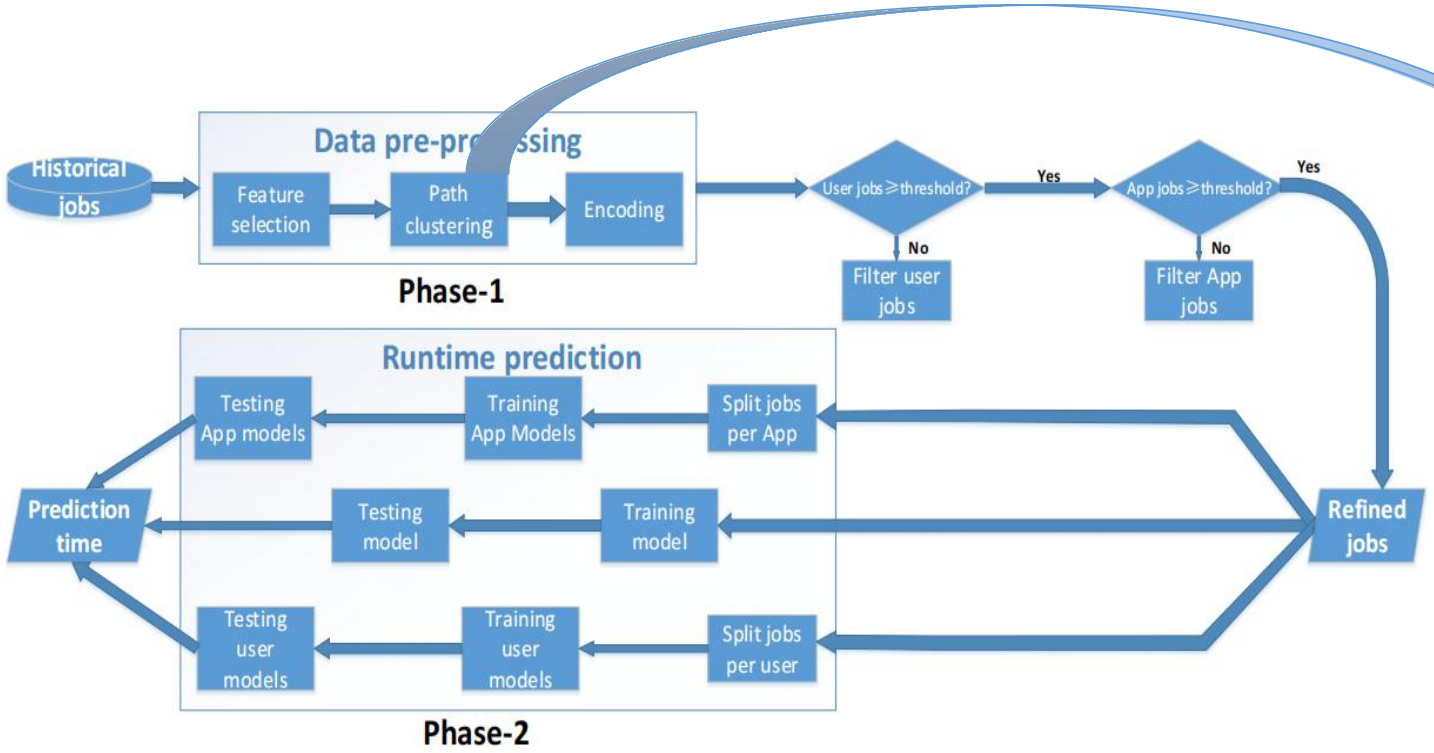
- Data Pre-processing



- **Feature Selection**
  - Method: Pearson's product-moment coefficient
  - Features: UID, Submit, ReqCPUs, JobName, Path
- **Path Clustering**
  - Aim: Group all similar paths (P)
  - Metric: Weighted Value Levenshtein Distance
  - Clustering: K-Means + 0-1Matrix
- **Encoding**
  - Hash Encoding

PREP Framework

- Data Pre-processing



- **Feature Selection**
  - Method: Pearson's product-moment coefficient
  - Features: UID, Submit, ReqCPUs, JobName, Path
- **Path Clustering**
  - Aim: Group all similar paths
  - Metric: Weighted Value Levenshtein Distance
  - Clustering: K-Means + 0-1Matrix
- **Encoding**
  - Hash Encoding

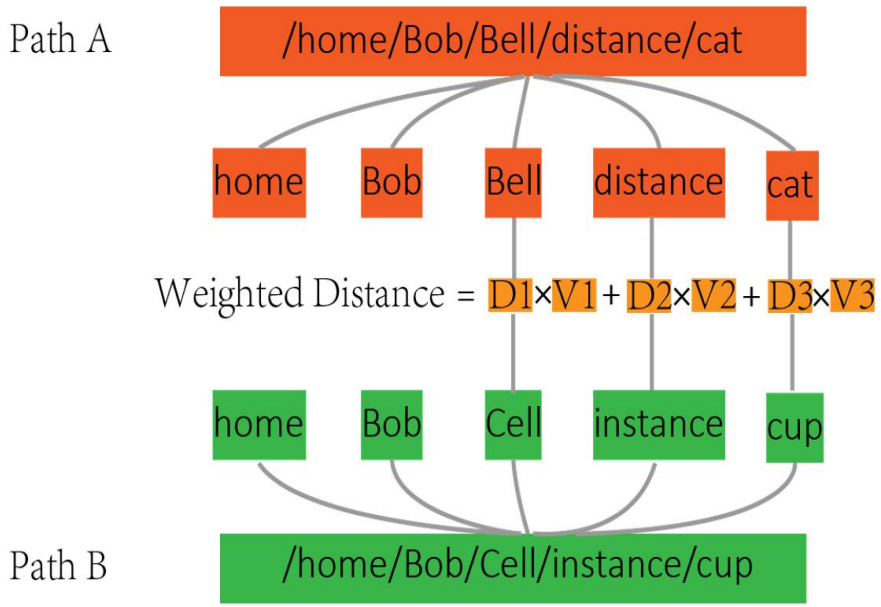
PREP Framework

- Path Clustering
  - The Levenshtein Distance represents the shortest edit distance between two strings and we use this distance to measure the similarity of two paths.

$$lev_{a,b}(ij) = \begin{cases} max(i,j), & \text{if min(i,j)=0} \\ min \begin{cases} lev_{a,b}(i-1,j)+1, \\ lev_{a,b}(i,j-1)+1, \\ lev_{a,b}(i-1,j-1)+1. \end{cases} & \text{otherwise} \end{cases}$$

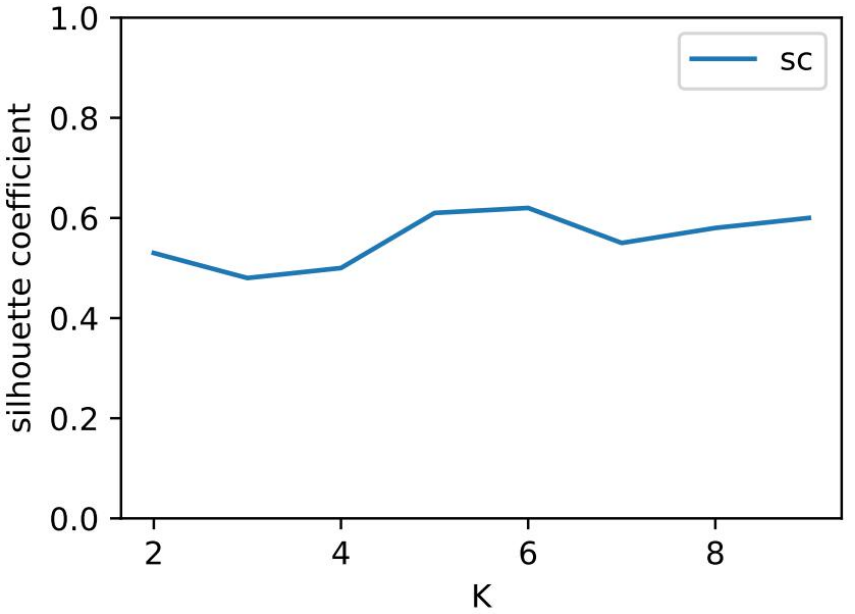
  - Weighted Distance: We assign different weights to each section of a path to increase the distance between short paths.

	Dissimilar Path
1	/home/Alisa/bell/A/2hour/F
2	/home/Bob/road/B/2019_TV2/G
3	/home/Carole/C/Calm/angle/H
4	/home/Devere/way-12/D/labor/I
5	/home/Eros/check/a2/E/b22c3/J



PREP Framework

- Path Clustering
  - Clustering Algorithm: The Silhouette Coefficient Method with K-Means is often difficult to find the most suitable K.



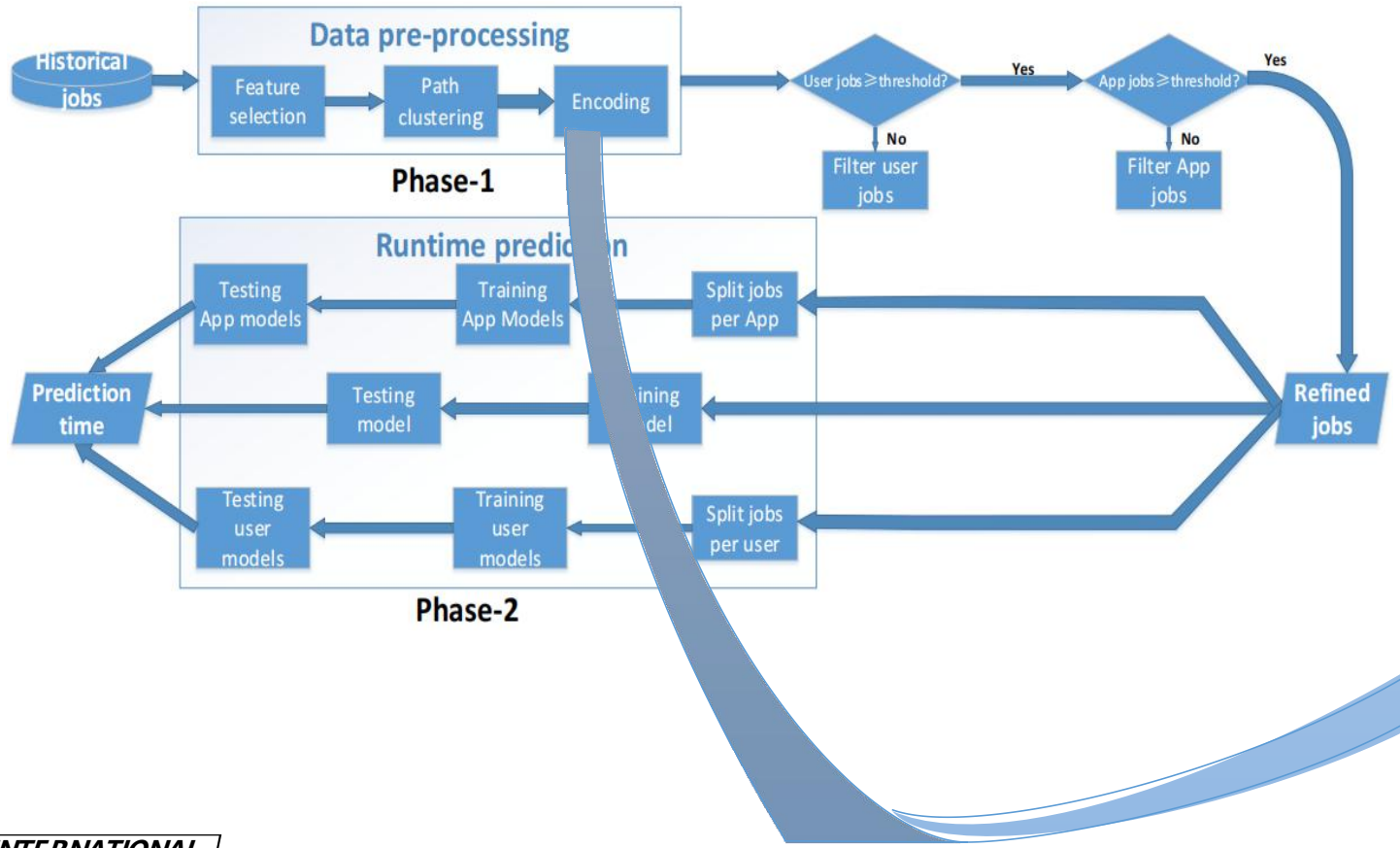
  - 0-1 matrix: We find that the weighted distance between similar paths is always less than 5. We convert the value to 0 or 1 according to whether the value is greater than 5.

	pathA	pathB	pathC	pathD	pathE
pathA	0	1	0	0	1
pathB	1	0	1	0	0
pathC	0	1	0	1	0
pathD	0	0	1	0	1
pathE	1	0	0	1	0

  - We use the K-Means method to cluster until the sum of the distances between all paths in each cluster is 0.

PREP Framework

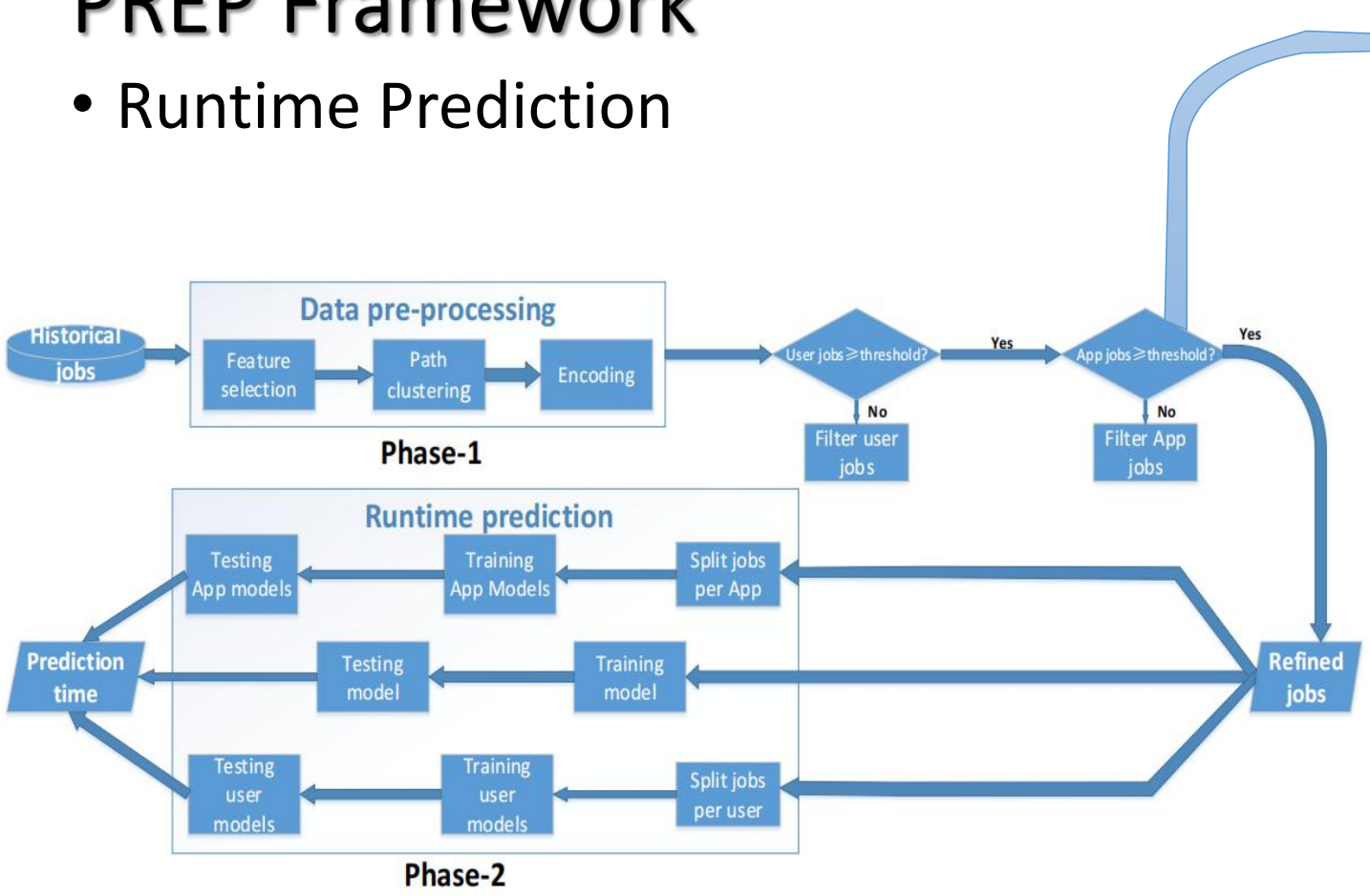
- Data Pre-processing



- **Feature Selection**
  - Method: Pearson's product-moment coefficient
  - Features: UID, Submit, ReqCPUs, JobName, Path
- **Path Clustering**
  - Aim: Group all similar paths (P)
  - Metric: Weighted Value Levenshtein Distance
  - Clustering: K-Means + 0-1Matrix
- **Encoding**
  - Hash Encoding

PREP Framework

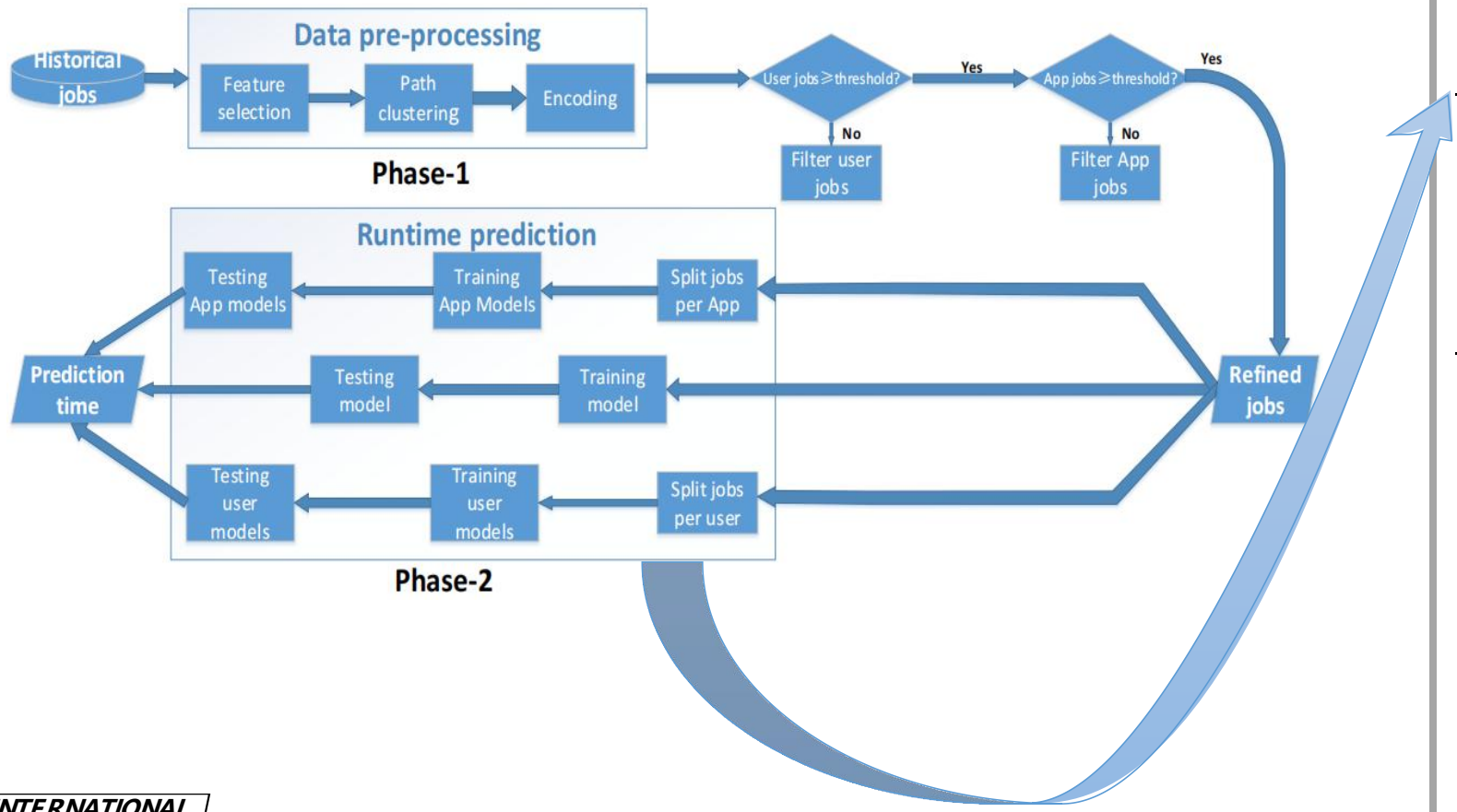
- Runtime Prediction



- **Refined Jobs**
  - Filter Jobs: User, Application.
- **Prediction**
  - Model: Support Vector Regression(SVR), Decision Tree(DT), Random Forests(RF).
  - 3 approaches: Comprehensive Model, User sub-model, App sub-model

PREP Framework

- Runtime Prediction



- **Refined Jobs**
  - Filter Jobs: User, Application.
- **Prediction**
  - Model: Support Vector Regression(SVR), Decision Tree(DT), Random Forests(RF).
  - 3 approaches: Comprehensive Model, User sub-model, App sub-model

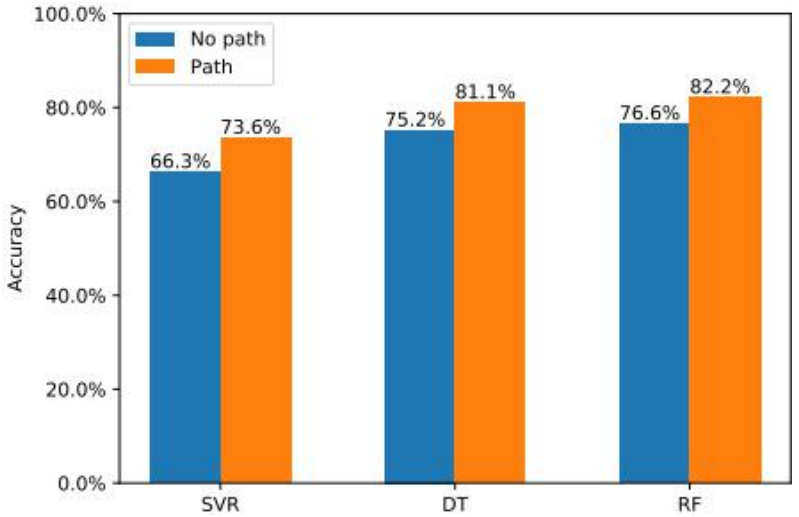
Experiment and Evaluation

- Training model
  - After data processing, the job characteristics we put into the model are shown in the table on bottom. The UID, ReqCPUS, Submit, PathType, NameType are used to training model and runtime is the training goal.
  - The data size is 53,332, and we randomly divide the job into a training set and a test set at a ratio of 3:1, namely 40,000 jobs for training model and 13,000 jobs for testing model.

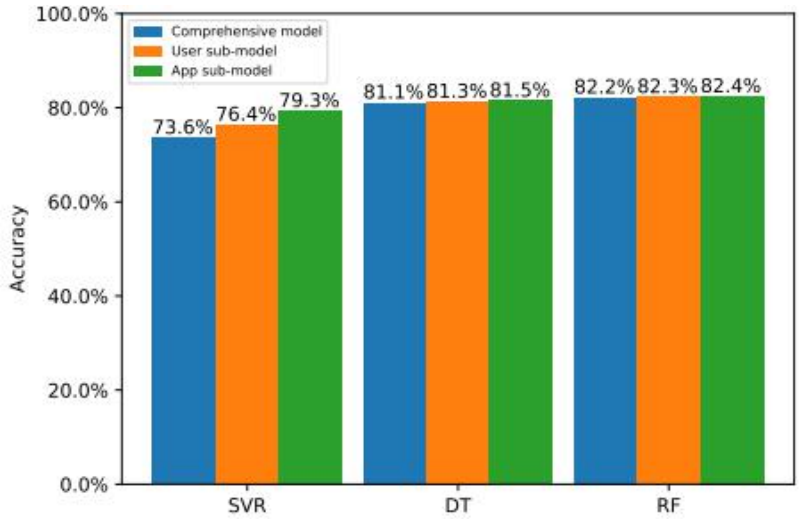
UID	ReqCPUS	Submit	PathType	NameType	runtime
3224625485180113950	512	69446	10146494682662368832	17727889271148596029	100
157560009062443987O8	1024	74215	52588008077877162l0	75576635195364l6698	150
128009643727882183O1	512	144502	162981138327777392O8	75576635195364l6698	90
429414866605090304	96	12893257	147346889031143233 31	72208862090398384 32	50
23243625432851834113950	1024	16895257	14734689032314323331	95723889271122596029	200

Experiment and Evaluation

- Evaluation
  - We put all training set in one model and try to put all training set only with 3 features: UID, ReqCPUS, Submit.

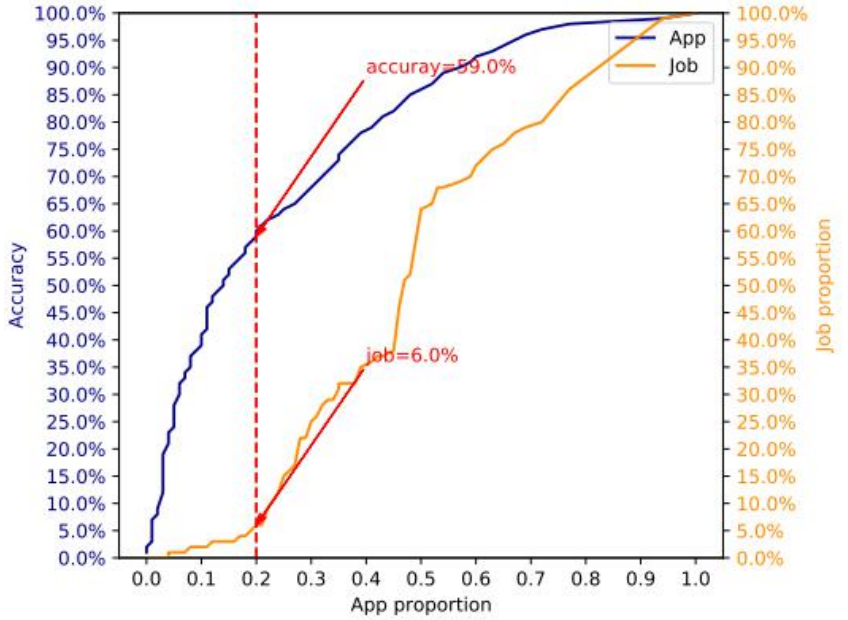


  - We assign an independent predictive model to each user and application, and the results is a weighted average of the accuracy of these sub-models.

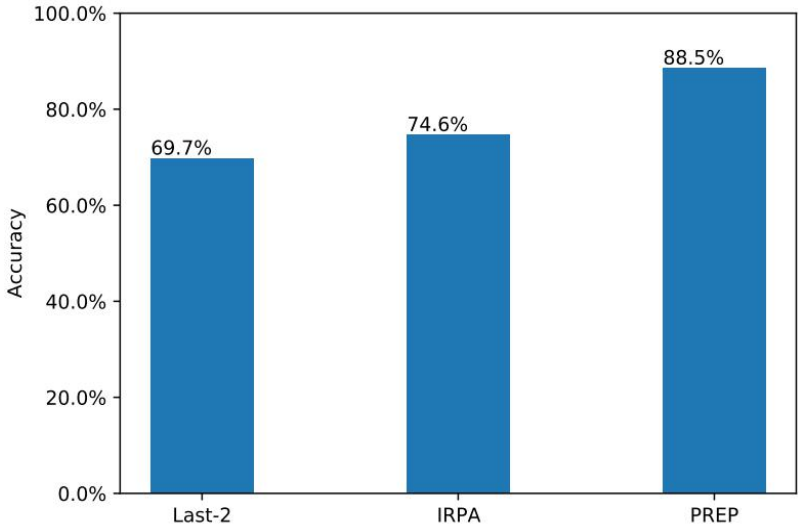


Experiment and Evaluation

- Evaluation
  - Because some applications with low accuracy contain a small number of jobs, our prediction is focused on the applications with large amounts of data.



  - This is a comparison of the two methods Last-2 and IRPA, PREP has a better effect.



Conclusion

- What we do?
  - We add a new feature named job running path to improve the prediction accuracy of job runtime.
- Results?
  - We cluster all similar path into several applications.
  - The new feature can improve the average accuracy on different models and the prediction based on each application can achieve the 88.5% accuracy.

Thanks for listening !