**Two-stage Approach for Predicting Samtool-view Execution Time**

1. **Overview**

In this report, we will build machine learning models to predict the execution time of Samtool-view. The strategy used is a two-stage approach which consists of two stages: 1) prediction of runtime parameters and 2) prediction of execution time. This approach is able to improve the performance of machine learning to predict the execution time of Wien2K, Povray, Blender, Montage compared to the single-stage approach [1]. However, it turns out that in the case of the prediction of the Samtool-view execution time, the performance of this approach is worse than the single-stage approach with a difference in the MAE score of 7%.

1. **Method**

First, I’ll explain how to build a two-stage approach to predict the execution time of Samtool-view. In this explanation, I will use a random forest. The first stage of the model predicts runtime parameters using pre-runtime parameters. Building this machine learning model consists of three steps:

1. Determine the best K parameters for predicting elapsed time
2. Determine the best K pre-runtime parameters for predicting the runtime parameters determined in step 1
3. Build machine learning model for predicting elapsed time

* Step 1: Determine the best K parameters for predicting elapsed time

The way to determine the best K parameters is to try all possible K values where K is in interval 1 until the total number of parameters. First, the correlation between elapsed time and each parameter is calculated. Then, a list is created that contains features that have been sorted from the most correlated to the least correlated as shown in Table 1. A machine learning model is built for the first K features of the list. By using random forest regression algorithm, the model with the first 7 features has the lowest mean absolute error, so we will only use the first 7 features.

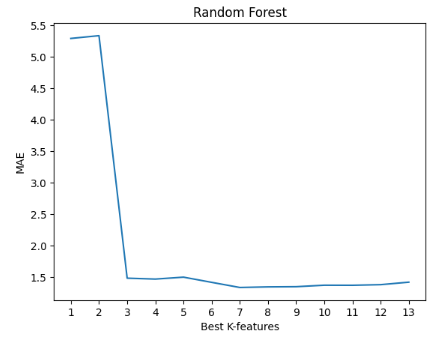


Figure 1. The random forest model for predicting execution time can achieve a minimum MAE score when using 7 features.

| Output size | **0.76** |
| --- | --- |
| Input size | **0.65** |
| Average CPU | **-0.57** |
| Average virtual memory size | **-0.35** |
| Write bytes | **0.19** |
| Peak cpu | **0.17** |
| Packing size | **0.15** |
| Read bytes | 0.11 |
| Number of allocated cpu | -0.10 |
| Peak virtual memory size | 0.07 |
| Peak RSS | 0.05 |
| Number of allocated memory | -0.04 |
| Average RSS | 0.03 |

Table 1. Correlation of all features to the elapsed time that has been sorted in descending order

* Step 2: Determine the best K pre-runtime parameters for predicting the runtime parameters

Of the 7 best parameters, 2 of them are pre-runtime parameters and the other 5 are runtime parameters. Because there are runtime parameters in it, we need to predict these 5 parameters using pre-runtime parameters. By using the same method as in step 1, we will choose the best K pre-runtime parameters by sorting the features in descending order based on their correlation to each runtime parameter and building a machine learning model using a random forest. It should be noted that we will not only evaluate the pre-runtime parameters which are part of the 7 best parameters. Based on Figure 2, successively the best K values for the output size, average cpu, average vsz, write bytes, and peak cpu prediction models are 4, 3, 2, 3, and 2. Details of the correlation values can be seen in Table 2.

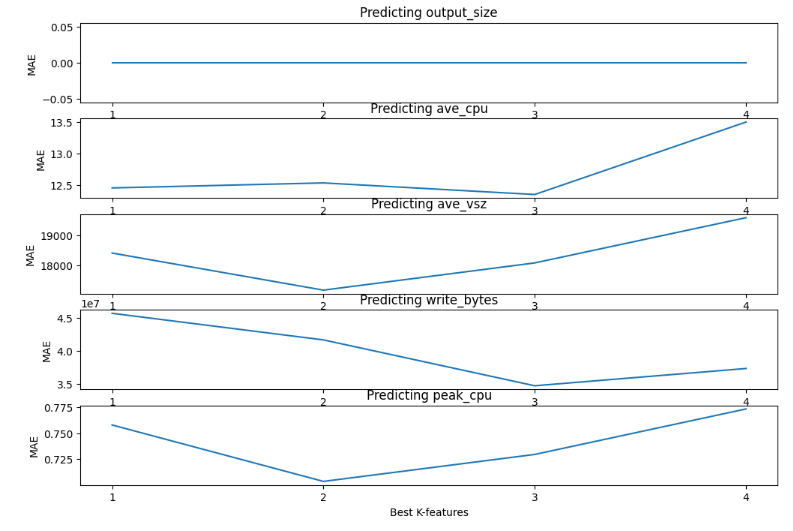


Figure 2. MAE plot over best-k features

|  | Input size | Packing size | Alloc Mem | Alloc CPU |
| --- | --- | --- | --- | --- |
| Output size | **0.89** | **0.02** | **-2.3e-05** | **4.31e-05** |
| Ave CPU | **0.10** | **-0.23** | 0.06 | **0.12** |
| Ave VSZ | **0.15** | **-0.18** | 0.06 | 0.09 |
| Write bytes | **0.31** | **-0.34** | 0.09 | **0.41** |
| Peak CPU | **0.15** | **-0.10** | 0.02 | 0.08 |

Table 2. Correlation of pre-runtime parameters with runtime parameters (the bolded value is the best parameter value)

* Step 3: Build machine learning model for predicting elapsed time

This model is built by utilizing the 7 features obtained in step 1. By using the 5 machine learning models from the second step we will predict the train data and test data for the 5 runtime parameters. The train data and test data are combined with the train data and test data input size and packing size to obtain 7 columns of train data and test data. The combined train data is used to train the elapsed time prediction model and the combined test data is used to evaluate it.

| Pre-runtime parameters | Input size |
| --- | --- |
| Number of allocated CPU |
| Number of allocated memory |
| Packing size |
| Runtime parameters | Output size |
| Peak CPU |
| Average CPU |
| Average RSS |
| Peak RSS |
| Average virtual memory size |
| Peak virtual memory size |
| Read bytes |
| Write bytes |

Table 5. Parameters classification.

1. **Result**

Models with a two-stage approach compared to single-stage approach models. The single-stage approach is built with pre-runtime parameters, which are packing size and input size. These two parameters are the best K pre-runtime parameters which produce the lowest MAE in the random forest regression model.

It turns out that the evaluation score (Figure 3) shows that the single-stage approach is better than the two-stage approach. So, the model chosen to predict the execution time of Samtool-view is the single-stage approach.

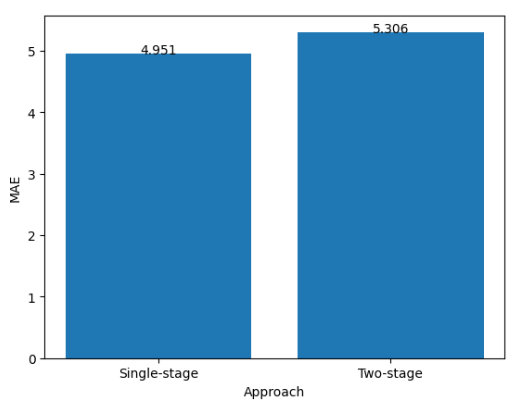
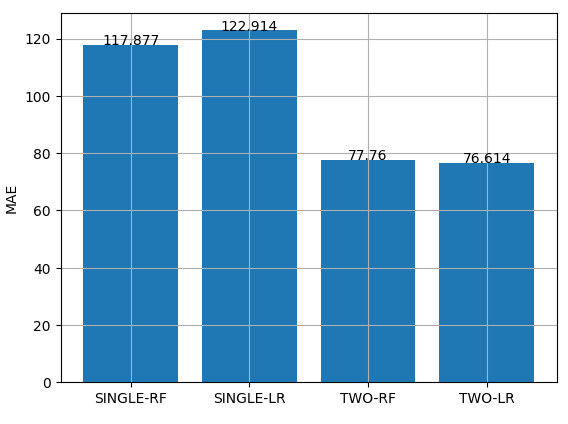
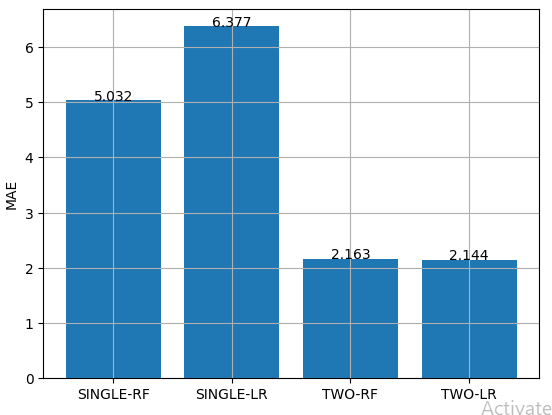
****

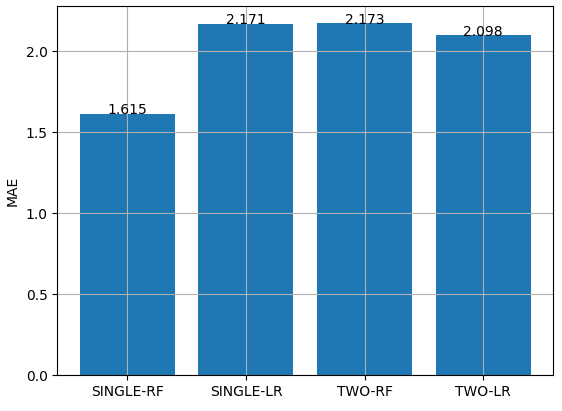
Figure 3. The single-stage approach model has better performance with a difference in MAE to the two-stage approach being 7%.



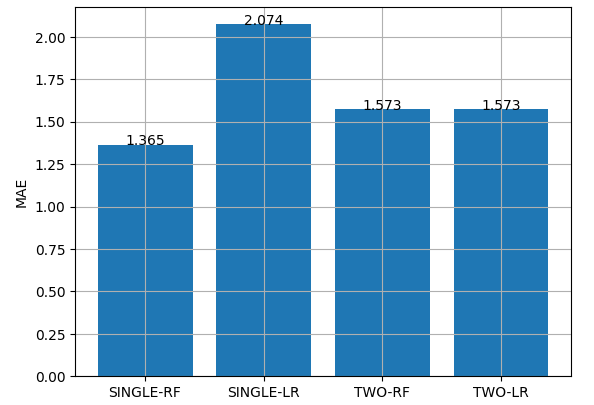
BWA-Mem



SAMTOOL-View



Picard-SortSam



PICARD-MarkDuplicates

**References**

[1] T.-P. Pham, J. J. Durillo and T. Fahringer, "Predicting workflow task execution time in the cloud using a two-stage machine learning approach", IEEE Trans. Cloud Comput., vol. 8, no. 1, pp. 256-268, Jan.-Mar. 2020