Workflow ETA

Dominykas Asauskas, Jean-Roch Vlimant

Workflow ETA Prediction

Overview and Goals

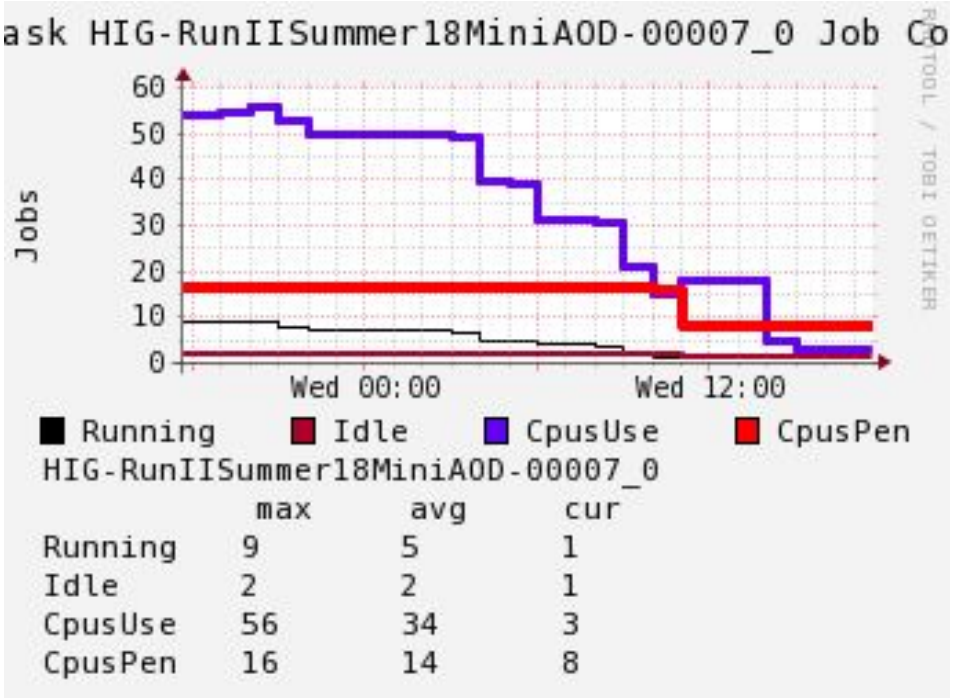
- Task* summary timestamps are taken every day and create daily plots

- Given multiple timesteps of each task in the workflow, the time for workflow to finish (ETA) can be hard to predict

- The aim is to predict the time completion (ETA) of a workflow based on the visual monitoring information

* - Single workflow consists out of multiple tasks

Example of Task Graph at Single Timestep



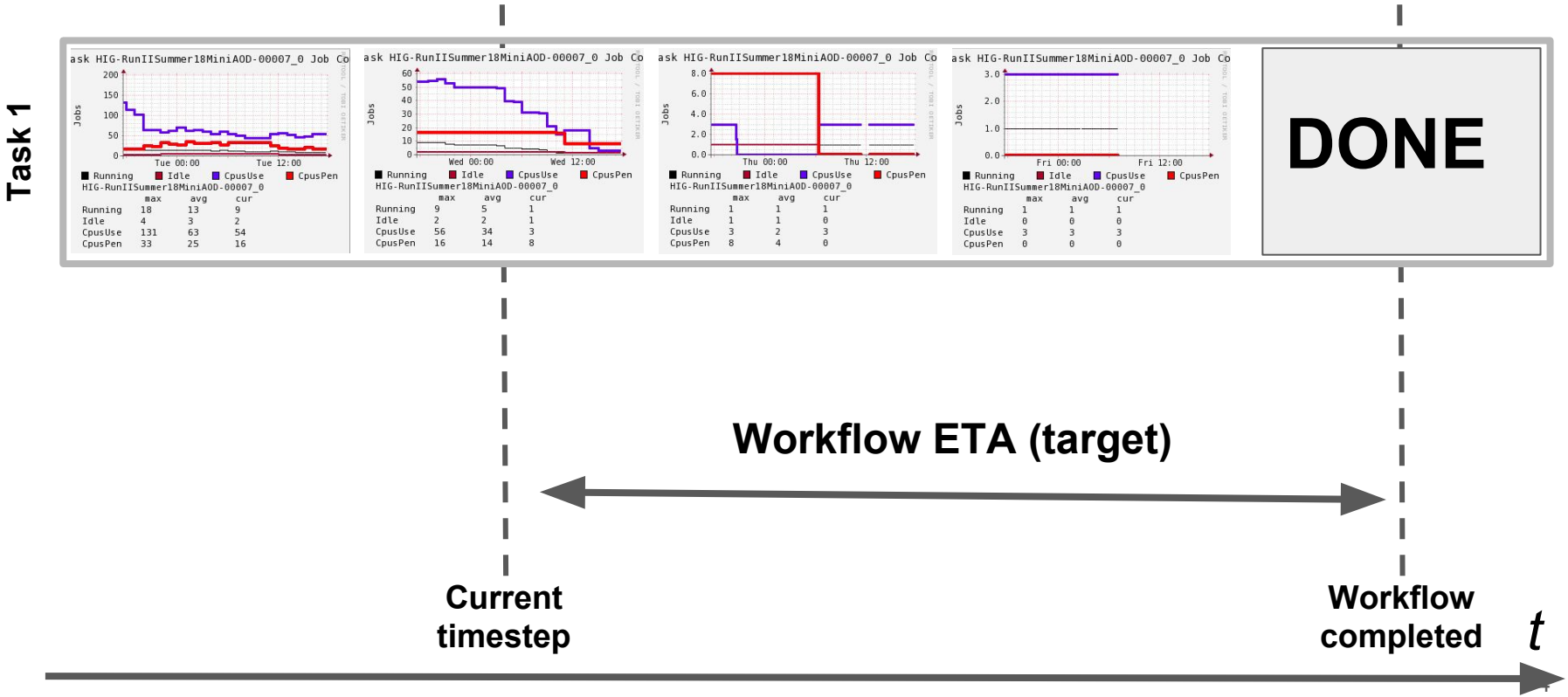
Has info about how many jobs are:

- Running
- Idle

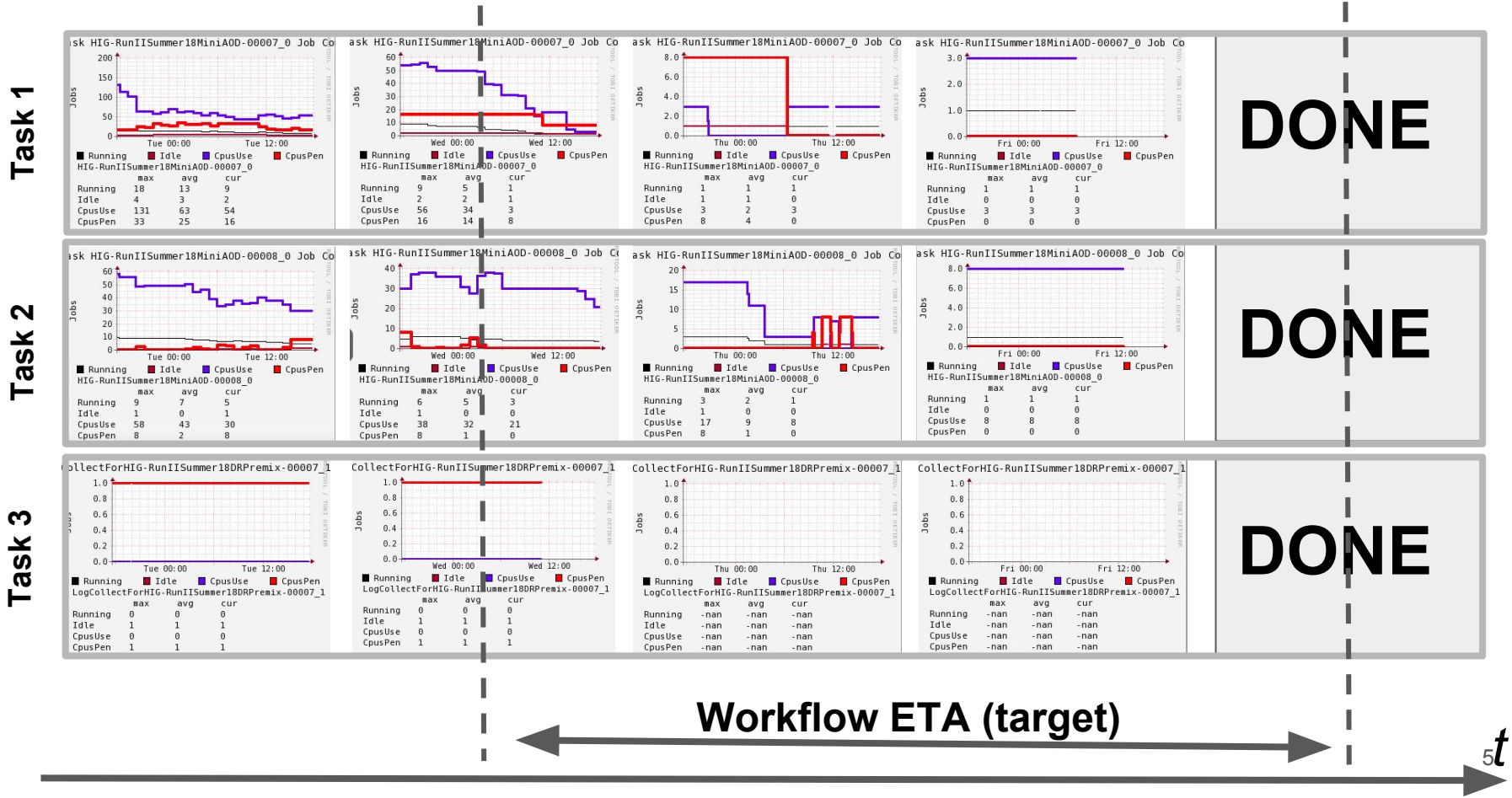
How many CPU’s are:

- In use
- Pending

Timeline of a Single Task



Workflow ETA Prediction (w/ multiple tasks)



Dataset Statistics

2 weeks datasets available:
from [2017|05|01 - 2017|05|07] and [2018|07|02 - 2018|07|08]

- Total images: **8875 + 12341**
- Total workflows: **1179 + 1132**

Datasets are filtered with following filters:

- Workflow in json must have “completed” timestamp
- Workflow must have at least one non empty graph
- Workflow files must exist and be in json

Filtered Dataset Statistics

Statistics after filtering:

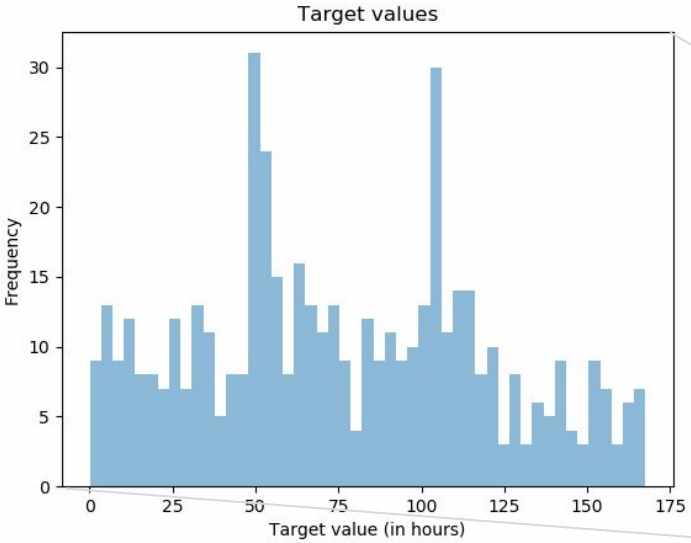
- Total images: **7108 + 423** (64% loss)
- Total workflows: **477 + 21** (78% loss)

Losses occur because:

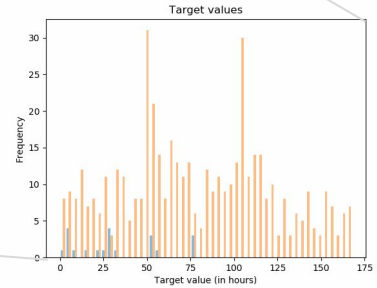
- Workflow graphs created after completion: **111 + 863** (42%)
- Not completed workflows: **591 + 128** (31%)
- Total number of empty graphs: **8415 + 12007** (96%)

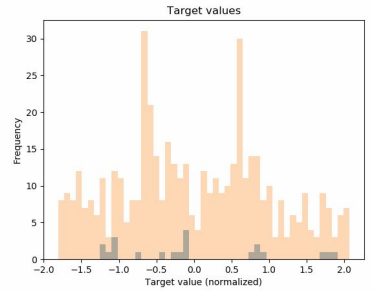
Even after filtering **6932/7531** (92%) of graphs are empty

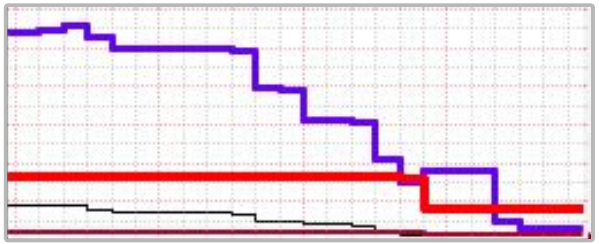
Dataset Preprocessing



All targets are normalized (target-mean)/std



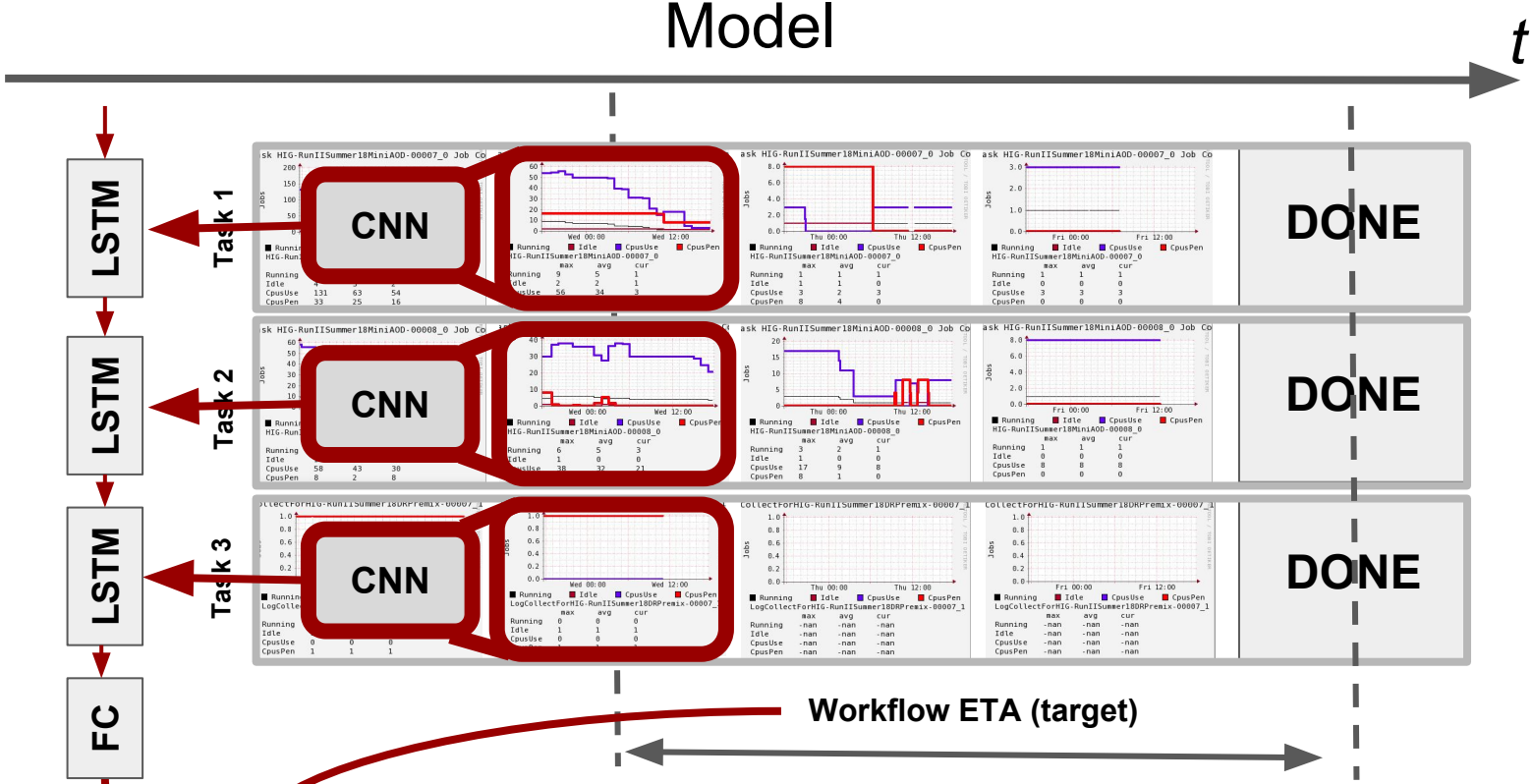




All images are cropped

All images divided by 255

Model



$$L_\delta(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{for} |y - f(x)| \le \delta, \\ \delta\,|y - f(x)| - \frac{1}{2}\delta^2 & \text{otherwise.} \end{cases}$$

(Huber loss)

Hyperparameters

test split - **0.2**

train and validation data is cross validated accross **5** kfolds

Huber loss threshold - **1.0**

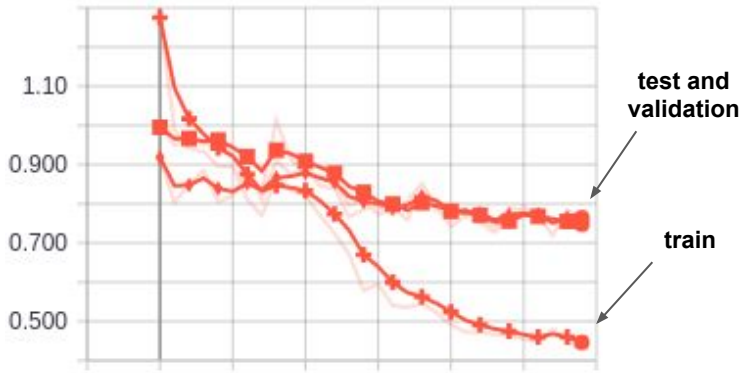
CNN: layers - **3**, first layer channels - **5**, increase of channels every layer - **3**

CNN: kernel - **(3,3)**, after each convolution max pooling is made with kernel - **(2,2)**

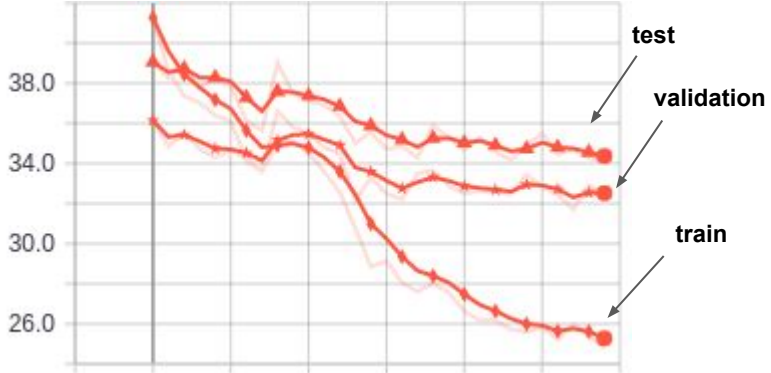
LSTM units - **100**, fc layers after LSTM - **1** w/ **30** units w/ **0.33** Dropout

Cross Validated Results (5 folds)

huber loss

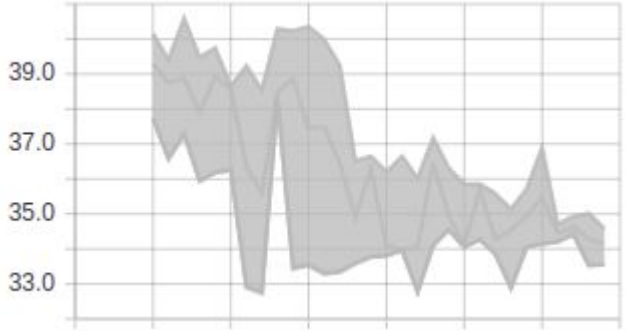


error std



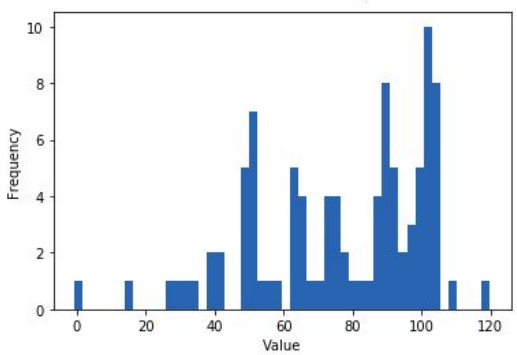
distribution across 5 folds (valid.)

distribution across 5 folds (valid.)

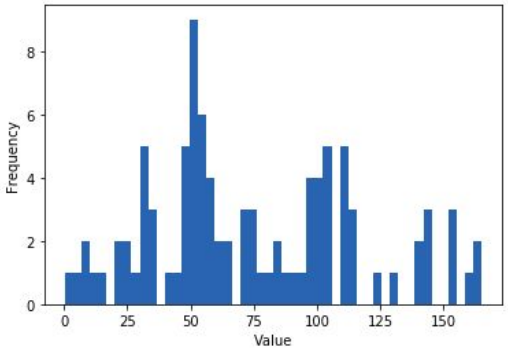


Test Dataset Results

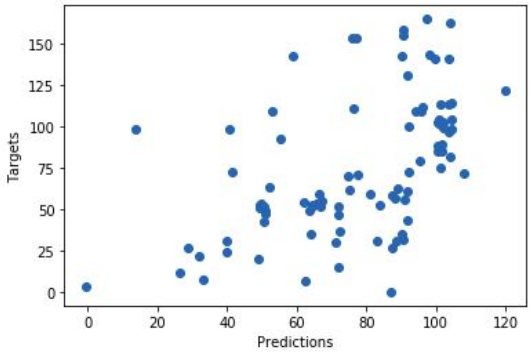
Prediction histogram (Hours)



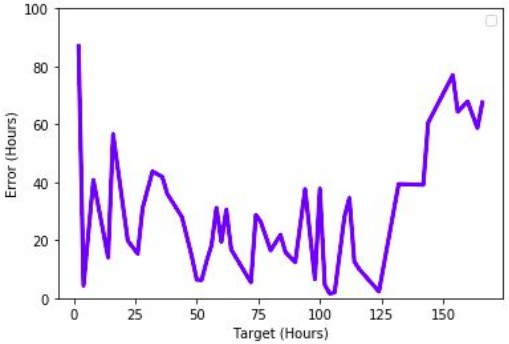
Target histogram (Hours)



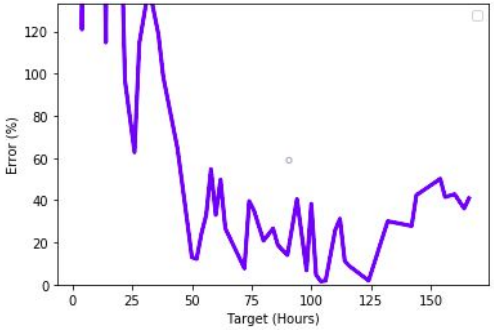
Predictions vs Targets



prediction - target



(prediction - target) / target



Next Steps

- That affect data:

Collecting more data

Collecting scalar data

Look for ways to filter raw data that gives better results

- That affect model:

Trying to predict how long each task may take (possible by looking for empty graphs)

Create a model that doesn't throw away previous timestep information