Practical work 12 - Recurrent Neural Networks

Students

- Liechti Matthieu
- Loup Olivia

Work to do and report

1.

Explore different number of LSTM units, different lengths of previous data (sequence length) and training epochs. Show the configuration that performed the best. Observe the resulting complexity of the network (e.g., number of trainable parameters).

We define the parameters :

- sequence length : 40
- epoch : 80
- units : 12

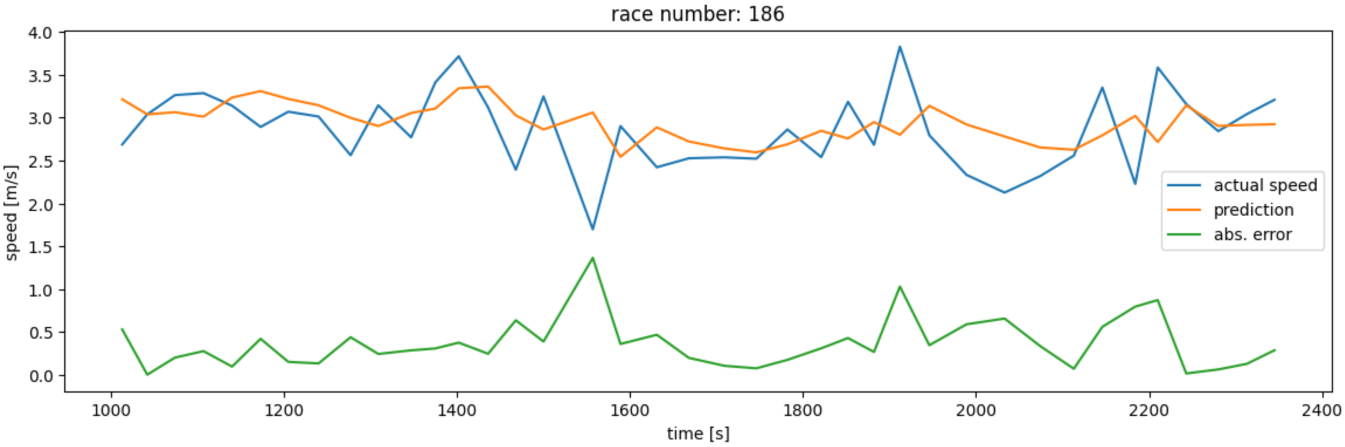
The *sequence length* have impact to increase correlation coefficient of train set but more decrease test length.

But this give also better result on predicition with values of error under 1.

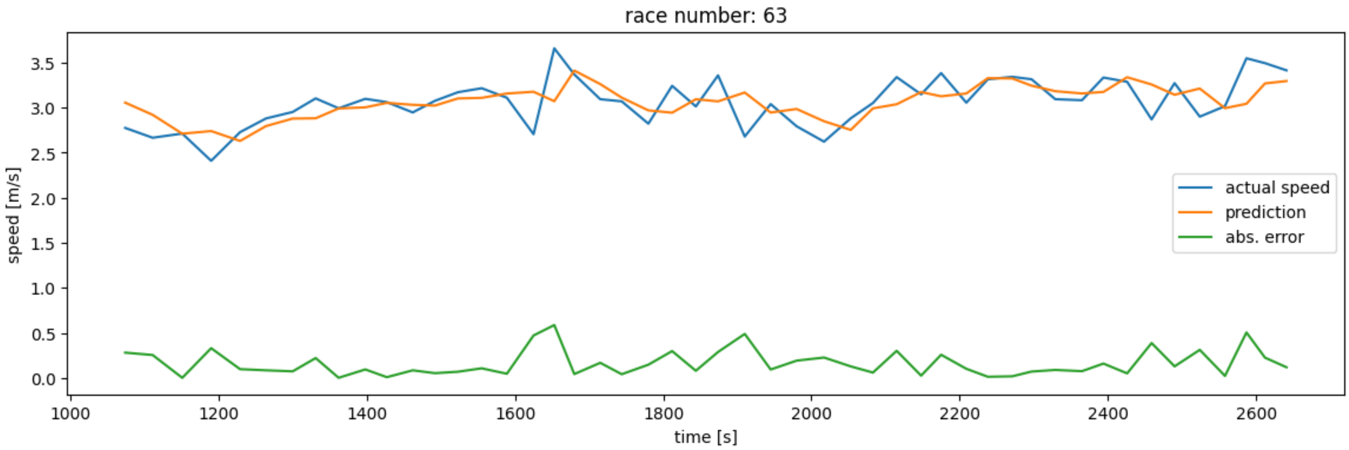
2.

What is the largest error (speed prediction) you observed? Do you observe that most of those large errors show up for high speeds ? or low speeds? Why?

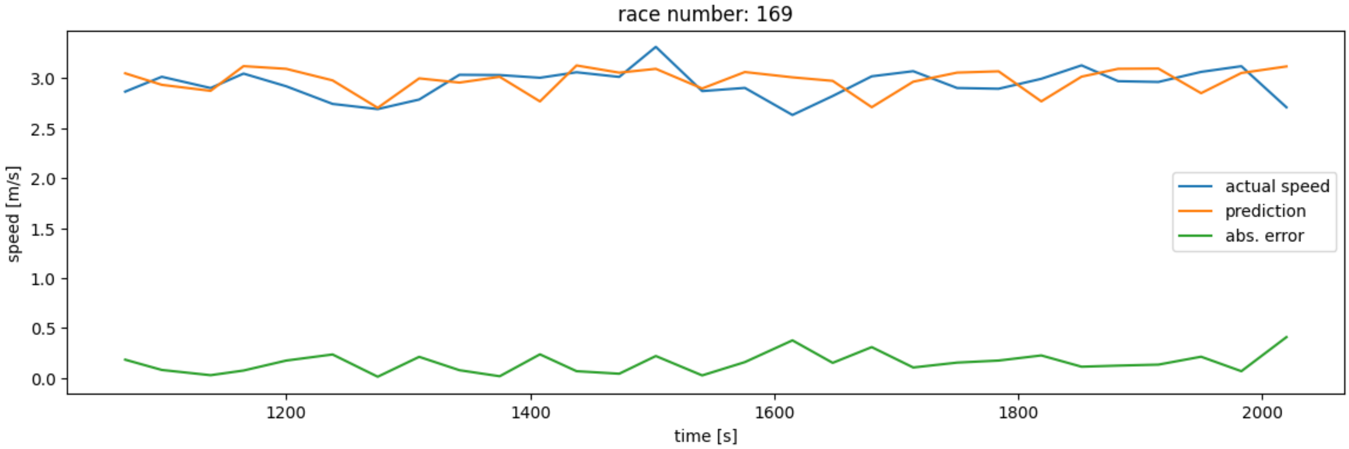
Largest error observed was less than one with parameters define in ex1. The bigger error was not dependend of fast or slow but of difference between previous speed. This could say the more source of larger error is the big difference of speed (acceleration or deceleration) than high or low speed. Big decceleration:



Big acceleration (and decceleration too):



And a case without big difference of speed:

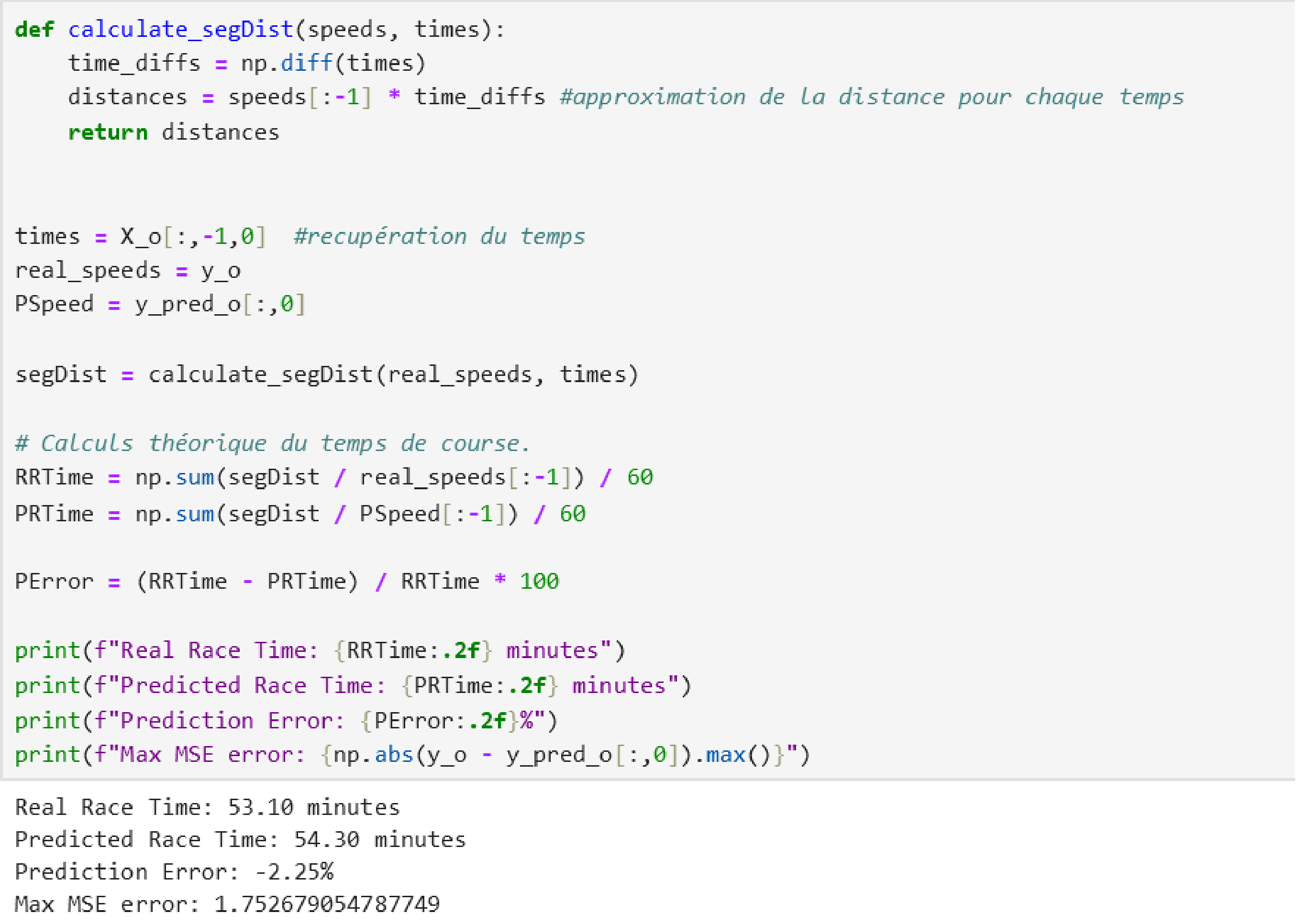


We could observe most of effect with big change.

3.

Using the predicted speeds for a given race, compute the expected time for a race and compute the difference between the real race time and the predicted race time in minutes. Provide the code of the cell that computes this prediction error.

Here is our code with the folowing result of the base notebook and the analysed cours was the 224.



```
def calculate_segDist(speeds, times):
    time_diffs = np.diff(times)
    distances = speeds[:-1] * time_diffs #approximation de la distance pour chaque temps
    return distances


times = X_o[:,-1,0]  #recupération du temps
real_speeds = y_o
PSpeed = y_pred_o[:,0]

segDist = calculate_segDist(real_speeds, times)

# Calculs théorique du temps de course.
RRTime = np.sum(segDist / real_speeds[:-1]) / 60
PRTime = np.sum(segDist / PSpeed[:-1]) / 60

PError = (RRTime - PRTime) / RRTime * 100

print(f"Real Race Time: {RRTime:.2f} minutes")
print(f"Predicted Race Time: {PRTime:.2f} minutes")
print(f"Prediction Error: {PError:.2f}%")
print(f"Max MSE error: {np.abs(y_o - y_pred_o[:,0]).max()}")
```

```
Real Race Time: 53.10 minutes
Predicted Race Time: 54.30 minutes
Prediction Error: -2.25%
Max MSE error: 1.752679054787749
```

When we observe the following result we can see a very little diffrence between the real and the predicted race time. Our model just under estimate the runner and predict a sligthly longer race time. This is why we have a negative prediction error of 2.25%. We also felt that repeating these measures across all courses would have been beneficial.