MSE | MASTER OF SCIENCE IN ENGINEERING

TSM – Machine Learning

PW 12: Recurrent Neural Networks

11 December 2023

Daniel Ribeiro Cabral – Bastien Veuthey

MSE | Data Science and Computer Science

Hes·so
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts Western Switzerland

Race time prediction

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).

Voici les différents hyperparamètres utilisés pour avoir les meilleurs résultats selon nos tests :

```
TIMESTEPS = 25
BATCH_SIZE = 64          # Size of the batch for training
NB_EPOCHS = 60    # Number of times the training dataset is presented
NB_UNITS = 32         # Number of LSTM units
=====
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_5 (LSTM)               (None, 32)                4608

 dense_5 (Dense)             (None, 1)                 33

=================================================================
Total params: 4641 (18.13 KB)
Trainable params: 4641 (18.13 KB)
Non-trainable params: 0 (0.00 Byte)
```

Cela donne une corrélation à 65% pour l'entraînement et une bien meilleure à plus de 70% pour le test set :

```
Training correlation coefficient: 0.6546829706537176
Test correlation coefficient: 0.7156412199181862
```

Note : À la suite de nos essais, nous avons remarqué qu'au-dessus de 20 timesteps des warning lors de la création des inputs/outputs étaient générés pour certaines courses. Nous avons donc décidé de travailler avec des valeurs à ces environs pour éviter de ne dépasser un trop grand nombre de warning.

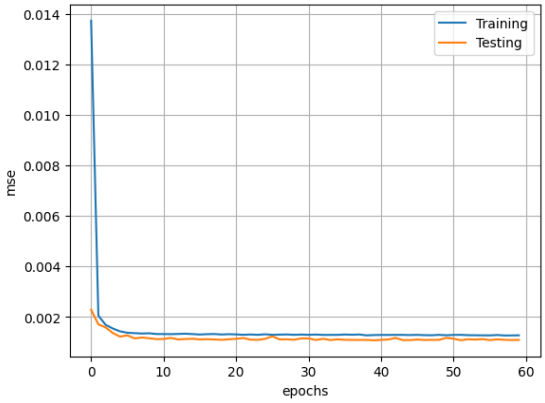

Figure 1 : MSE loss avec le nombre d'epochs

Différents éléments peuvent être observés grâce à ce graphique de la MSE loss au travers du nombre d'epochs :

- Les taux d'erreur de formation et de test chutent fortement au cours des premières époques, ce qui suggère que le modèle apprend rapidement au départ. Nous aurions pu nous arrêter à

un nombre plus petit (ex : 30) mais étonnement nos meilleurs résultats ont été fait aux alentours des 60.

- Après la chute initiale, la MSE se stabilise pour le training et le test. Cela indique que le modèle a atteint un point où la poursuite de l'apprentissage n'améliore pas de manière significative les performances sur l'ensemble d'apprentissage, et qu'il n'y a pas d'**overfitting** visible puisque la loss du test reste faible et proche de celle du training.
- La proximité entre les lignes suggère une bonne généralisation du modèle à des données inédites (unseen).

La figure 2 nous renseigne sur une race random (#148) et le traitement entre sa vitesse réelle et la prédiction de notre modèle (avec une comparaison par l'erreur) :

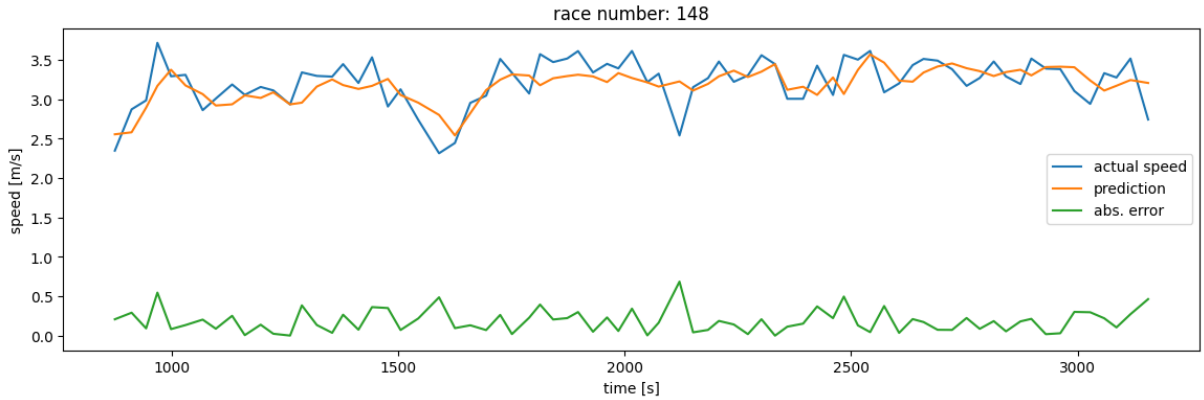

Figure 2 : Prédiction du modèle

Les vitesses réelles et prédites se suivent de près pendant la majeure partie de la course, ce qui indique que le modèle a une performance prédictive décente. Nous remarquons tout de même que la prédiction prend toujours un temps de latence pour s'adapter au changements brusques (et vient même à en louper un complètement). Elle semble indiquer la moyenne de la course tout du long.

L'erreur absolue semble être relativement faible et cohérente tout au long de la course, sans aucun pic significatif indiquant des erreurs de prédiction importantes (qui sortent de l'ordinaire du moins, peut-être excepté celle du milieu cf. 2.).

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?

En se basant sur la figure 2, un pic ressort parmi tous les autres :

On voit que la vitesse du coureur chute brusquement pour reprendre quasiment après à la même vitesse. Cela peut être dû à un facteur externe à la course comme à une chute du coureur, à un ralentissement involontaire ou à une caractéristique propre à la course. Cela devient quasi impossible à prédire pour le modèle. Nous n'avons pas remarqué de différences

fondamentales entre les vitesses hautes et faibles, à chaque évolution spontanée, l'erreur augmente pour ensuite se stabiliser.

L'erreur absolue moyenne sur toute la course a été calculée à 0.186.

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.

Voici le code que nous avons utilisé pour calculer l'erreur de prédiction :

```
def calculate_segment_distances(speeds, times):
    # Calculate the time difference between consecutive points
    time_diffs = np.diff(times)
    # Approximate distance for each segment
    distances = speeds[:-1] * time_diffs
    return distances

# Extract time and speeds from the dataset
times = X_o[:,-1,0]  # time in seconds
real_speeds = y_o
predicted_speeds = y_pred_o[:,0]

# Calculate distances for each segment
segment_distances = calculate_segment_distances(real_speeds, times)

# Calculate race times using these distances
real_race_time = np.sum(segment_distances / real_speeds[:-1]) / 60  # Convert to minutes
predicted_race_time = np.sum(segment_distances / predicted_speeds[:-1]) / 60

prediction_error = (real_race_time - predicted_race_time) / real_race_time * 100  # Percentage
error

print(f"Real Race Time: {real_race_time:.2f} minutes")
print(f"Predicted Race Time: {predicted_race_time:.2f} minutes")
print(f"Prediction Error: {prediction_error:.2f}%")
```

Avec ces résultats :

```
Real Race Time: 38.03 minutes
Predicted Race Time: 38.59 minutes
Prediction Error: -1.47%
Max MSE error: 0.6864103995451498
```

La difference est quasi minime (à la minute près). Il est cependant bon de noter que notre modèle a prédit que le coureur prendrait plus de temps qu'il n'en a réellement pris (d'où la valeur négative). Cela n'indique pas un mauvais résultat simplement que notre modèle a sous-estimé pour cette course. Cela serait intéressant de compiler les résultats du même coureur au fil de ses courses et ainsi de continuer à améliorer notre modèle.