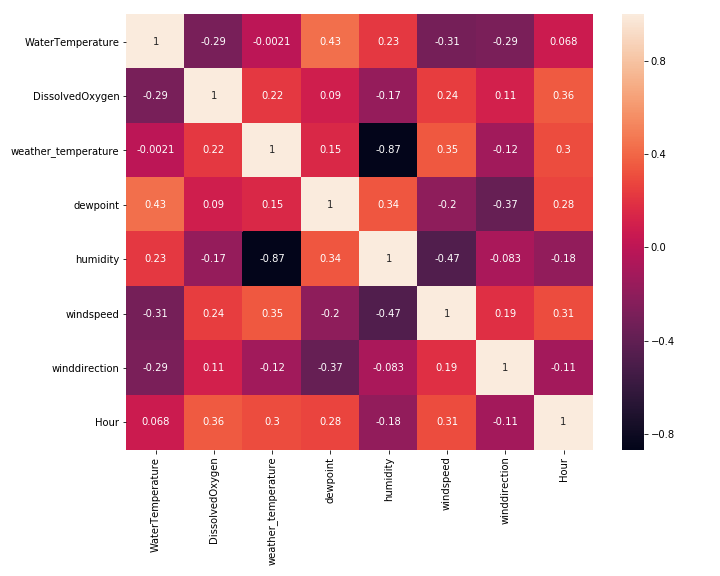
Biocéanor rapport Yann Cherdo

# Exploration de la donnée

1. 5 appareils sont répartis sur deux sites A et B,
2. Tous les appareils ont été enregistrés dans la même période du 15 au 26 juin, à quelques jours manquants,
3. Les séries temporelles ont des points manquants par échantillonnage,
4. Les séries temporelles ont des échantillons manquants suivant une période d’échantillonnage de 1h,
5. Les domaines dans lesquels les différentes variables évoluent peuvent varier d’un appareil à l’autre (exemple : DissolvedOxygen entre les appareils 1 et 2 : av2=64, av1=88),
6. La matrice de corrélation de l’ensemble des variables numériques montre que la variable à prédire (DissolvedOxygen) est relativement corrélée aux autres variables ainsi qu’à l’heure d’occurrence de l’échantillon.



# Préparation de la donnée

1. Les comportements des différents appareils pouvant dépendre des différents sites, ces appareils pouvant comporter des problèmes individuels de mesure ou être confrontés à des contraintes environnementales différentes voire des anomalies locales, les données sont séparées par appareil de manière à pouvoir tester des combinaisons restreintes le cas échéant,
2. Un ordonnancement chronologique des données est appliqué,
3. Les points manquants par échantillonnage sont interpolés lorsque la période d’échantillonnage d’1h est respectée.
4. Une variable indiquant l’heure de l’échantillon est rajoutée, étant corrélée à la variable à prédire, elle sera exploitée par le modèle de prédiction.
5. Utilisant un RNN (voir section suivante) comme modèle de prédiction, les séries temporelles seront découpées suivant une fenêtre glissante de 48 points et des 6 points suivants formant ainsi deux ensembles X et Y de séries temporelles. Les couples x et y des séries temporelles ne présentant pas une période d’échantillonnage uniforme d’1h seront rééchantillonnés par interpolation. Cela permet de garantir une période d’échantillonnage d’1h et que le modèle apprenne à prédire les 6 points correspondant aux 6 heures suivant les 48 heures les précédant.
6. Toutes les variables sont normalisées une à une dans [0, 1] de manière à correspondre aux domaines à dérivée non nulle des fonctions d’activations des cellules RNN.
7. Toutes les variables des séries temporelles sont conservées dans l’ensemble X tandis que seule la variable cible (DissolvedOxygen) est conservée dans les séries temporelles de l’ensemble Y.
8. Les variables de l’ensemble Y sont agrégées suivant l’usage many-to-many du modèle décrit dans la section suivante.

X\_train shape: (630, 48, 8)

Y\_train shape: (630, 6)

X\_test shape: (271, 48, 8)

Y\_test shape: (271, 6)

# Modèle de prédiction

Les signaux étant relativement périodiques, les périodes dominantes correspondant d’ailleurs généralement à celle d’un jour étant comprises dans la période de 48h proposée pour la prédiction, et n’utilisant que 48 points pour effectuer une prédiction, un RNN GRU, plus faible en complexité que le LSTM, semble approprié.

Une architecture many-to-many est utilisée, permettant à partir d’une série temporelle de taille a d’en prédire une de taille b.

La prédiction n’adressant que 6 points chacun de dimension 8, une couche dense et linéaire permet de produire directement la prédiction entière en produisant un vecteur de taille 6x8=48.

Le modèle utilisé a pour paramètres :

* Hidden\_units : spécifie le nombre de couches cachées (GRU) du réseau. Le modèle peut donc être testé ‘stacked’ ou simple,
* Epochs : le nombre de fois que le modèle doit apprendre successivement la donnée d’entraînement,
* Mini\_batch\_size : nous utilisons une mini\_batch gradient descent. La donnée d’entrainement est séparée en autant de mini batchs qu’elle peut contenir. Les loss sont agrégées au sein de chaque mini batch.
* Learning\_rate : le facteur d’apprentissage appliqué à chaque mise à jour des poids du modèle suivant la descente de gradient.

Le modèle est entraîné sur 50%, testé sur 20% et validé sur 30% des séries temporelles de tous les appareils. Les données d’entraînement sont mélangées aléatoirement à chaque génération de mini batch dans chaque époque.

Une optimisation Adam est utilisée, l’objectif étant de minimiser la Mean Square Error entre les séries temporelles prédites et réelles.

# Résultats

Paramètres :

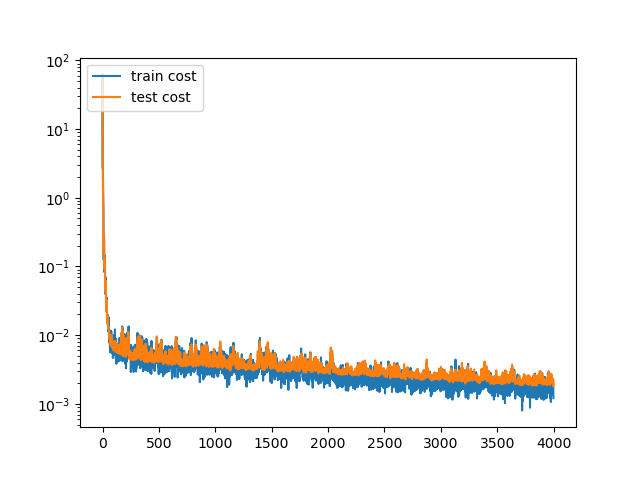
hidden\_units = [150, 50]

epochs = 200

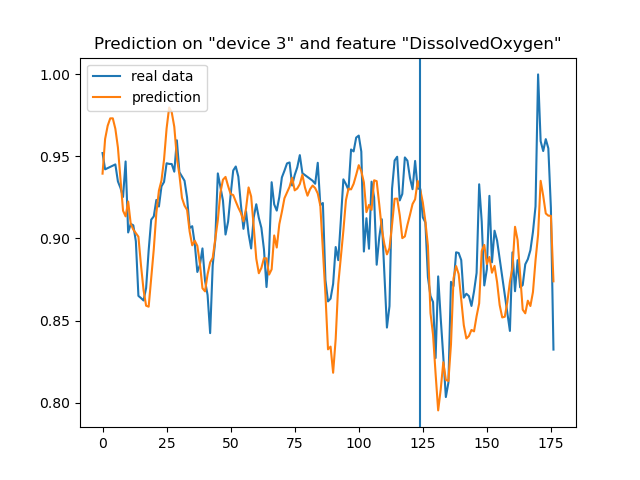
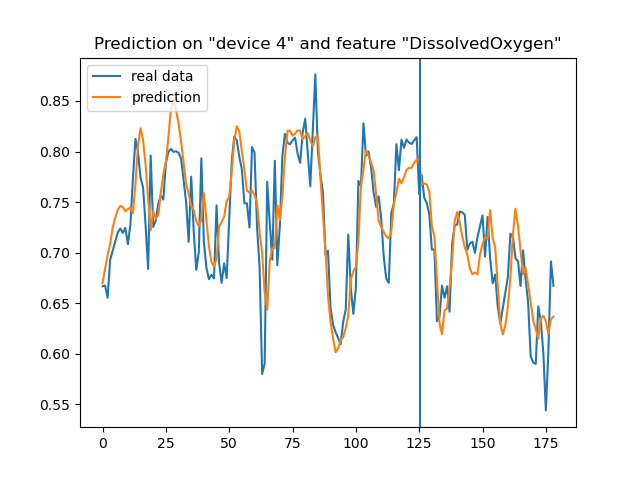
mini\_batch\_size = 32

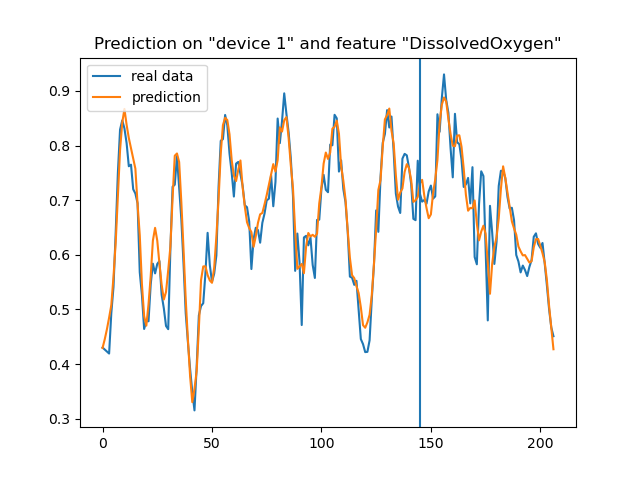
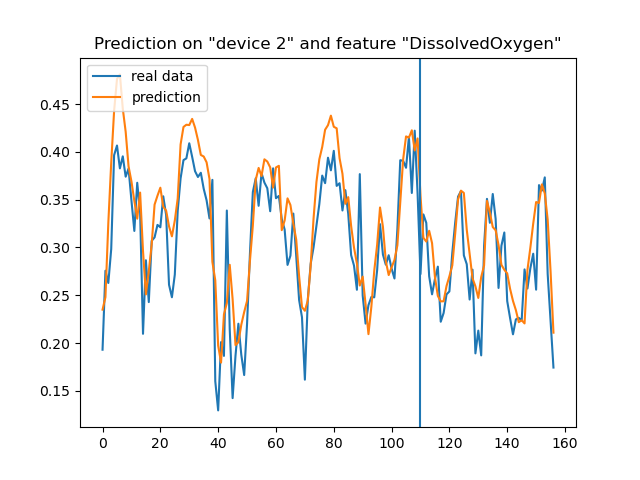
learning\_rate = 0.01

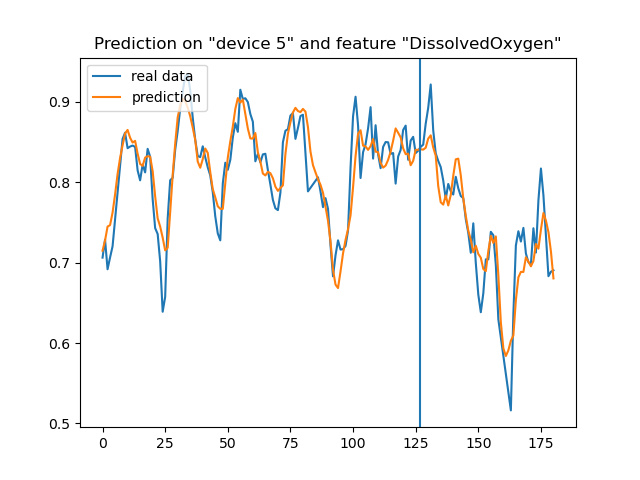
* Loss :



* Prédiction train/test/valid (train/test à gauche de la ligne verticale, valid à droite) :







MSE à la dernière époque : train\_cost=0.0016, test\_cost=0.0022

# Discussion

Les prédictions sont globalement satisfaisantes pour un premier essai.

La loss montre un très léger overfit qui, compte tenu des résultats pourrait être négligé. Un dropout pourrait être rajouté.

Des époques de 100 et 300 ainsi que des hidden layers plus volumineux et plus faible (non stacked) ont été essayés sans meilleur succès.