PARTIE 1

Projet Intelligence Collective et Apprentissage Profond

Detection of abnormal events in oil wells

1. Expliquer en quoi l'usage d'un CNN 1D est pertinent pour traiter des données temporelles

Tout d'abord, dans les réseaux neuronaux convolutifs unidimensionnels (CNN 1D), le noyau glisse le long d'une seule dimension, ce qui les rend parfaitement adaptés pour le traitement de données de séries temporelles. En effet, ces données présentent une forte corrélation entre les points voisins dans le temps, et donc le noyau des CNN 1D, ne pouvant se déplacer que le long d'une dimension, celui-ci peut se déplacer le long de la dimension temps des données temporelles. Les CNN 1D peuvent alors extraire des motifs temporels en capturant les relations locales le long de la dimension temporelle.

De plus, les données temporelles sont généralement organisées en une séquence unidimensionnelle avec plusieurs variables à chaque pas de temps, ce qui correspond parfaitement à l'entrée des CNN 1D. Ces réseaux sont donc particulièrement efficaces pour analyser et modéliser des données temporelles, ainsi que comprendre les tendances et les schémas récurrents.

Les CNN 1D produisent aussi une sortie unidimensionnelle, ce qui facilite l'extraction de caractéristiques pour la prédiction et la classification, et ils peuvent être suivie d'une couche dense permettant de prendre en compte des motifs plus globaux ou de faire des prédictions à long terme.

Pour toutes ces raisons, l'usage des CNN 1D semble pertinent pour traiter des données temporelles.

2. Proposer une transformation des telles données de manière à ce que celles-ci puissent être traitées par des convolutions 1D.

Dans notre jeu de données (oil_wells_data.csv), chaque point temporel est accompagné d'un ensemble de variables (P-PDG, P-TPT, T-TPT…), il s'agit donc une série temporelle multivariée.

Afin de préparer ces données pour qu'elles puissent être traitées par des convolutions 1D, nous pouvons suivre les étapes suivantes.

Tout d'abord, nous allons séparer la cible de nos variables. Comme nous cherchons à prédire la colonne « class », c'est notre cible, nous allons la **séparer** des caractéristiques, les variables. La colonne « timestamp » doit également être traitée séparément car elle a un format différent, nous pouvons la transformer en format numérique afin qu'elle puisse être prise en compte par le CNN 1D.

Ensuite, nous pouvons voir que trois des caractéristiques (P-JUS-CKGL, T-JUS-CKGL, QGL) n'ont pas de valeurs, nous allons donc les **supprimer**. Les cinq autres caractéristiques (P-PDG, P-TPT, T-TPT, P-MON-CKP, T-JUS-CKP) sont numériques, il faut les **normaliser** pour qu'elles aient une échelle cohérente. Il ne semble pas y avoir de valeurs manquantes dans le fichier, si c'est le cas, il faudra les gérer.

Enfin, nous devons séparer nos données en **fenêtres temporelles**. Pour ces fenêtres, nous devons définir une taille fixe. Comme nos caractéristiques sont enregistrées chaque seconde, mais que notre cible « class » varie à une plus grande échelle, une fenêtre de plusieurs secondes, voire minutes pourrait être adaptée. De plus, ces fenêtres doivent se chevaucher afin de capturer les motifs de façon efficace et de ne pas en rater si la fenêtre n'est pas correctement positionnée.

Sources :

https://stats.stackexchange.com/questions/550769/why-cnn-is-suitable-for-time-series-data

https://towardsdatascience.com/how-to-use-convolutional-neural-networks-for-time-series-classification-56b1b0a07a57

https://machinelearningmastery.com/how-to-develop-convolutional-neural-network-models-for-time-series-forecasting/

https://laxfed.medium.com/convolutional-neural-networks-for-sequence-processing-part-1-420dd9b500

https://machinelearningmastery.com/machine-learning-data-transforms-for-time-series-forecasting/

https://www.kaggle.com/code/mersico/understanding-1d-2d-and-3d-convolution-network