Master research Internship

Bibliographic report

Deep Learning and Time Series

Domain: (examples) Data Structures and Algorithms - Logic in Computer Science

Author:
Arthur Le Guennec

Supervisor:
Romain Tavenard of your first supervisor
Simon Malinowski of your second supervisor
OBELIX

Abstract: write your abstract here

Table des matières

1 Introduction

Il y a beaucoup de papier sur la classification des séries temporelles, et il existe beaucoup de méthodes diférentes pour y parvenir. La classification des séries temporelles est présente dans beaucoup de domaine comme la médecine, la biologie, la reconnaissance audio, ... Nous allons utiliser içi les réseaux de neurones profonds. Dans ce modèle d'apprentissage, les réseaux convolutionnels (ou Convolutional Neural Networks) ont de très bons résultats dans le domaine de la classification d'image ([2]). Il existe cependant d'autres méthodes dans d'autres papiers ([1]) tels que le bag-of-words, ou encore les machines à vecteur de support. Comme dans le domaine des séries temporelles les nombres de données annotées est faible, on allons explorer des méthodes permettant de diminuer les conséquequences de ce manque de données.
Ce papier est organisé de la manière suivante. La section 2 concerne l'état de l'art dans ce domaine, la section 3 introduit les outils qui vont nous servir pour la classification, et la section 4 s'interessera aux problèmes qui nous concernent.

2 State of the art

La classification de série temporelle est un sujet qui est étudié depuis maintenant plusieurs années, donc différentes techniques existent et de nombreux articles sur le sujet. Dans cette partie nous allons voir quelques unes de ces différentes techniques.

2.1 Bag-of-Words

Le Bag-of-Words (BoW) est un modèle de description de document avec des descripteurs (des mots). Il fonctionne en attribuant des mots aux documents, chaques mots ayant un probabilité par document (voir figure **??**). [1] traite justement le sujet de la classification des séries temporelles avec les Bag-of-Words.

2.2 Support Machine Vector

Les SVM (Support Vector Machine ou Machine à Vecteur de Support en français) sont des techniques d'apprentissage supervisé.

3 Neurals Networks and Time Series

3.1 Neural Network

Les réseaux de neurones sont un système inspiré du modèle biologique du cerveau. Un réseau de neurone est composé de plusieurs neurones (voir figure 2) qui ont chacun une fonction dactivation (voir figure 3) et qui sont reliés à d'autres neurones par des poids.

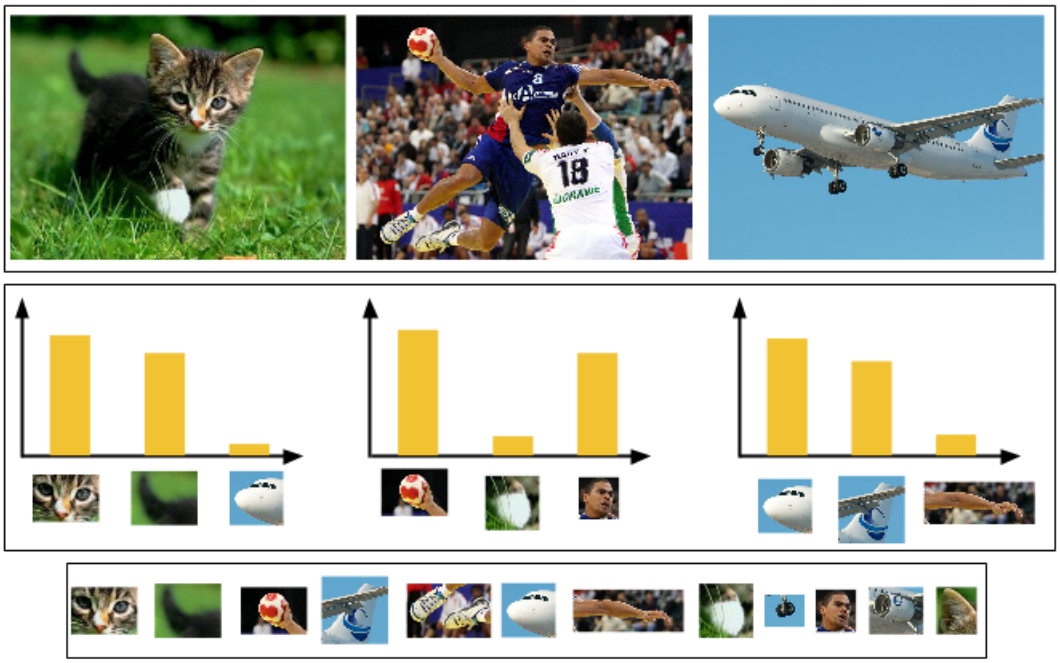
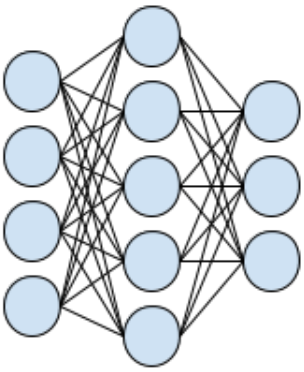
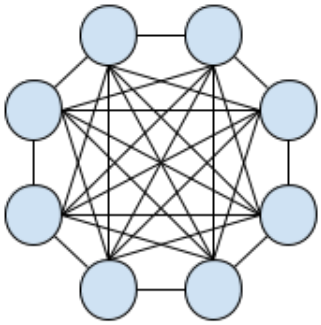


FIGURE 1: Le bag-of-words est en bas de la figure, il est composé de différents mots, ici des images, qui sont reconnus sur les documents (en haut), qui sont ici des images. Par exemple, pour la première image, on a une forte correspondance avec le premier et deuxième mot, alors que lon a une faible correspondance avec la dernière image.



(a) Réseau de neurones à couches



(b) Réseau de neurones à connexion complète

FIGURE 2: Il existe différents types de réseaux de neurones. Par exemple, la figure 1.a est un réseau avec une architecture à couche, alors que le réseau de la figure 2.b à une architecture à connexion complète. Dans le domaine de la classificatin, l'architecture à couche est souvent utilisé.

L'un des avantages d'un réseau de neurones est qu'il s'agit d'un système qui apprend par entraînement. Il y a donc deux étapes pour un réseau de neurone, dabord l'entraînement, puis la vérification. Un autre avantage est qu'il donne souvent de bons résultats, de nombreux papiers

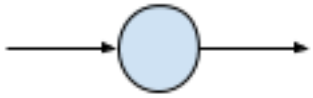


FIGURE 3: La fonction d'activation est la fonction qui va déterminer le comportement du neurone. Par exemple, si la fonction d'activation est un threshold, la sortie sera soit 1, soit 0. Si cest une sigmoide (le plus souvent utlisé), la sortie sera y = 1 / (1 + exp(-x)) avec y la sortie et x l'entrée.

sur le sujet obtiennent souvent des taux derreurs très faibles (mettre les références ici). Comme vu précédemment, il y a deux étapes essentielles pour le fonctionnement dun réseau de neurones : l'entraînement et le test. L'entraînement est là pour trouver les poids corrects entre les neurones, et le test pour vérifier que ces poids sont corrects. Dans les deux cas, le réseau a besoin d'une base de donnée sur laquel s'entraîner et tester. Cette base de donnée sera alors séparée en deux parties, l'une pour l'entraînement, une autre pour le test. L'efficacité d'un réseau de neurones dépend de son architecture. Selon le nombre de couche et le nombre de neurones par couche, le réseau peut être plus ou moins performants ([2][6]). La taille de la base de donnée utilisé influence aussi les résultats, Par exemple, [J'ai oublié la référence] montre bien que les performances dépendent bien de la taille des données utilisées. *Rajouter que les méthodes influencent aussi sur le résultats.*

3.2 CNN

Les CNN (Convolutional Neural Network, ou réseaux convolutionnels) sont un structure particulière des réseaux de neurones car les neurones d'une couche à une autre ne sont pas tous reliés entre eux comme dans un réseau de neurone à couche classique (voir figure 2b). Les connections entre neurones se font localement, ce qui fait des entrées d'une couche cachée un sous-ensemble de motif de la couche précédente. La figure 4 illustre bien l'architecture d'un CNN.

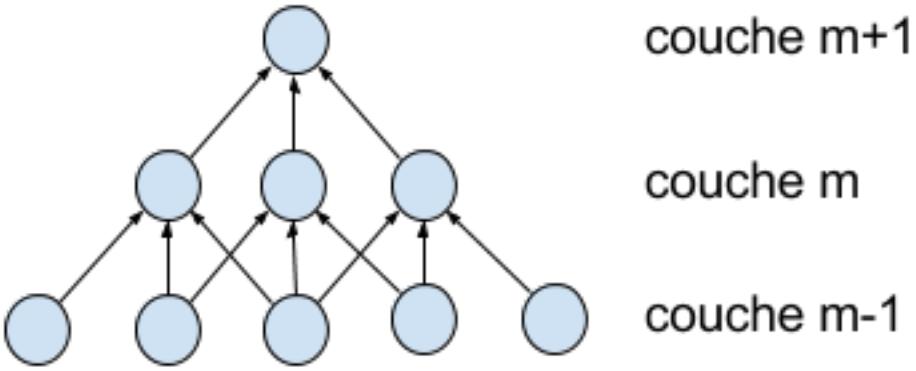


FIGURE 4: Ici, imaginons que la couche m-1 est une rétine où chaque neurone correspond à un pixel d'une image. Dans la couche m, on s'aperçoit que chaque neurone a 3 entrées qui correspondent chacune à un pixel. Chaque neurone de la couche m est donc un sous-ensemble de l'image de 3 pixels. De la même façon, chaque neurone de la couche m+1 est donc un sous-ensemble de l'image de 5 pixels.

3.3 Time Series

Les séries temporelles sont des données qui dépendent du temps. Par exemple, la taille d'une personne par rapport à son ge peut être vu comme une série temporelle. Le document [1] nous donne les définitions qui nous seront utile par la suite. Il existe de nombreux domaines où les séries temporelles interviennent. Elles peuvent, par exemple, servir dans la biologie, la médecine, la finance, ...

4 Data-augmentation for Time Series

4.1 Pourquoi chercher à augmenter la base de donnés ?

On a besoin de palier le manque de donnée car les structures CNN ont besoin de beaucoup de données pour pouvoir être entraînés efficacement. Les structures tels que MC-DCNN (voir [7]) peuvent être utilisé. De plus il y a peu, comparés aux images, de données sur les séries temporelles. On aura donc besoin d'augmenter ces données de différentes façons. Par analogie aux images, on regardera comment transformer une série temporelle. Dans les données pour les images, on applique certaines transformations, tel que la translation, l'inversion, le changement de couleur, de contraste, de luminosité, ... (voir figure 5). Ces transformations ne sont pas applicables directment aux séries temporelles.

4.2 Comment y parvenir ?

Pour les images, plusieurs techniques existent. Par exemple, [2] explique que pour réaliser une augmentation de données, ils perturbent une image avec des transformations qui n'invalide pas cette image, c'est à dire que si l'image d'origine montrait un chat, l'image transformée doit toujours montrer un chat. [5] explique que pour augmenter ses données, il sépare tout d'abord ses données en deux parties, l'une pour l'apprentissage, et l'autre pour les tests. Dans les données pour l'apprentissage, ils réalisent des sous-images en extractant des zones de 224x224 pixels de l'image d'origine et de sa réflection horizontale, qui a une taille de 256x256. On obtient donc une multiplication de 2048 de la base de données d'origine. Les données de test prendront 5 sous-images et leur réflection (donc 10 images) selon leurs prédictions. Toujours dans l'idée d'augmenter leurs données d'apprentissage, ils effectuent une ACP pour pouvoir, non pas réduire les dimensions, ajouter du bruit contrlé sur les composants principaux. Krizhevsky et al. [5] arrivent donc á baisser le taux d'erreur de plus de 1% ainsi. Dans le papier [6], Howard utilise le même procédé mais rajoute d'encore d'autres étapes.

Beaucoup de méthode de data-augmentation le sont pour les images, la question étant de savoir si parmi ces méthodes, on peut en réutiliser ou non. Par exemple, on ne peut pas réaliser une réflection sur une série temporelle. Cependant, lapproche de Krizhevsky dans [5] pour les données dentraînement avec une ACP pourrait être réalisable sur des séries temporelles. On pourrait peut-être aussi rajouter du bruit contrlé selon différents paramètres comme dans [5] et [3].

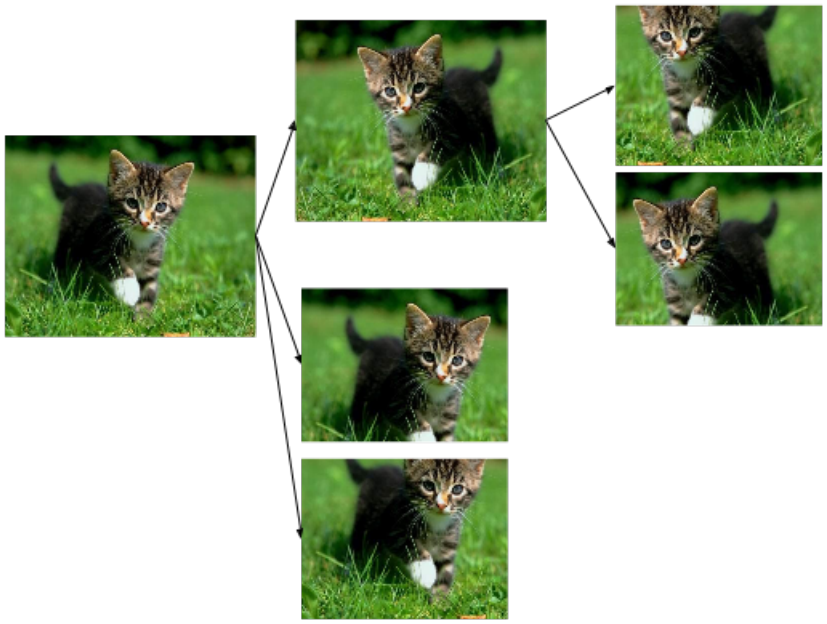


FIGURE 5: Exemple d'augmentation de donnée pour une image. Içi, on a obtenu six images à partie d'une seule. On a d'abord appliqué une inversion d'image et puis on a rogné les images obtenues de deux manières différentes. En réalité, beaucoup plus de transformations sont appliquées, permettant de multiplier la base de données des images de beaucoup (1024 dans [3]).

5 Conclusion

Pour conclure , il existe de nombreuses manières d'augmenter le volume des données. Cependant, ces méthodes dépendent du type de données. Avant de commencer à classifier des séries temporelles, il va donc falloir trouver comment augmenter ce volume. Nous pourrions, par exemple, utiliser l'agorithme ACP ou s'inspirer des méthodes utilisés sur les images. Pour la classification, les CNN semblent être un bon choix étant donnée les bons résultats obtenus dans [7]. Pour cela, nous allons utiliser le framework Caffe (voir [4]). Une fois mis en place, nous pourrions alors observer les résultats.

References

Références

[1] Adeline Bailly, Simon Malinowski, Romain Tavenard, Thomas Guyet, and Laetitia Chapel. Bag-of-temporal-sift-words for time series classification. In *ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data*, 2015.

[2] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details : Delving deep into convolutional nets. *arXiv preprint arXiv :1405.3531*, 2014.

[3] Andrew G Howard. Some improvements on deep convolutional neural network based image classification. *arXiv preprint arXiv :1312.5402*, 2013.

[4] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe : Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*, pages 675–678. ACM, 2014.

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[6] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout : A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1) :1929–1958, 2014.

[7] Yi Zheng, Qi Liu, Enhong Chen, Yong Ge, and J Leon Zhao. Time series classification using multi-channels deep convolutional neural networks. In *Web-Age Information Management*, pages 298–310. Springer, 2014.