IMT Atlantique
Technopôle de Brest-Iroise - CS 83818
29238 Brest Cedex 3
Téléphone : +33 (0)2 29 00 13 04
Télécopie : +33 (0)2 29 00 10 12
URL : **www.imt-atlantique.fr**

Mini project report

Machine Learning : Binary Classification

CIOCARLAN Alina
PYTHOUD Axel

Date d'édition : 26 novembre 2020
Version : 1

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Sommaire

1. Introduction

Rapport en anglais ? (je pense qu'il vaut mieux perso) Présenter notre mission à travers ce projet, présenter rapidement les datasets, dire qu'on va étudier une classification basée sur du deep learning et que nos but et d'optimiser les paramètres tout en créant un workflow qui s'adapte à plusieurs datasets. Donner les principales tâches que l'on s'est fixés (pre process, feature selection, param tuning via grid search, k fold cross validation and evaluate the best model obtained)

2. Data preprocessing

Exemple de note de bas de page : OFDM [1].

2.1. Étude ou expérience 1

Exemple d'édition d'équation :

$$Y(f) = \frac{1}{\sqrt{T}} \sum_{p=0}^{P-1} \sum_{k=-\infty}^{+\infty} \sum_{n=0}^{N-1} \frac{1}{a_k^{(p)}} e^{-j2\pi \frac{n}{T} \tau_0^{(p)}} e^{j\Phi_0^{(p)}} e^{-j2\pi \frac{f}{a_k^{(p)}} (kT + \tau_0^{(p)})} d_k^{(n)} G_e\left(\frac{f - f_{d,k}^{(p)}}{a_k^{(p)}} - \frac{n}{T}\right). \tag{1}$$

Autre exemple d'édition d'équation :

$$\mathrm{E}\{|y_k^{(n,p)}|^2\} = \underbrace{\mathrm{E}\{|\gamma_k^{(0,p)}|^2\}\mathrm{E}\{|d_k^{(n)}|^2\}}_{signal} + \underbrace{\sum_{\substack{n'=0 \\ n' \neq n}}^{N-1} \mathrm{E}\{|\gamma_k^{(n'-n,p)}|^2\}\mathrm{E}\{|d_k^{(n')}|^2\} + \mathrm{E}\{|w_k^{(n,p)}|^2\}}_{bruit}. \tag{2}$$

Exemple d'édition d'équation avec surlignage (mise en évidence d'un résultat) :

$$Y(f) = \sum_{p=0}^{P-1} \frac{1}{a^{(p)}} e^{j\Phi_0^{(p)}} e^{-j2\pi \left(\frac{f - f_d^{(p)}}{a^{(p)}}\right) \tau_0^{(p)}} X\left(\frac{f - f_d^{(p)}}{a^{(p)}}\right). \tag{3}$$

Exemple d'édition d'une expression matricielle :

$$\mathbf{\Gamma_k^{(p)}} = \begin{pmatrix} \gamma_k^{(0,p)} & \gamma_k^{(1,p)} & \cdots & \gamma_k^{(N-1,p)} \\ \gamma_k^{(-1,p)} & \gamma_k^{(0,p)} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_k^{(-(N-1),p)} & \gamma_k^{(-(N-2),p)} & \cdots & \gamma_k^{(0,p)} \end{pmatrix}. \tag{4}$$

Exemple d'édition d'expression vectorielle :

$$\mathbf{y_k^{(p)}} = e^{j\Phi_0^{(p)}} \left(\mathbf{\Gamma_k^{(p)}} \otimes \mathbf{\Phi^{(p)}}\right) \mathbf{d_k} + \mathbf{w_k^{(p)}}, \tag{5}$$

$$\mathbf{w_k^{(p)}} = [w_k^{(0,p)}, ..., w_k^{(n,p)}, ..., w_k^{(N-1,p)}]^T. \tag{6}$$

1. Orthogonal Frequency Division Multiplexing

2.2. Étude ou expérience 2

Exemple de figure au format `*.png` (de préférence), possiblement `*.jpg` (Figure 1) :

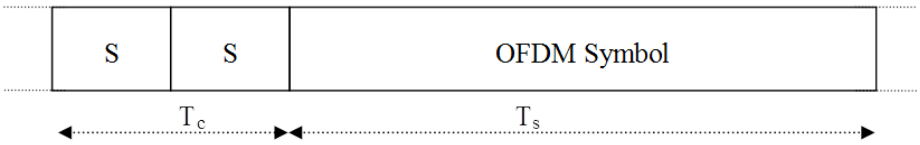


FIGURE 1 – Structure de la trame à l'émission

2.3. Étude ou expérience 3

Exemple de définition éditée dans un bloc de style *Beamer* :

Définition

$B = (B_t)_{t \in \mathbb{R}_+}$ à valeurs réelles est un **mouvement brownien** (ou **processus de Wiener**) issu de x si

1. $B_0 = x$
2. $0 \leq t_i \leq t_j \Rightarrow B_{t_j} - B_{t_i} \sim \mathcal{N}(0, \sigma^2(t_j - t_i))$
3. $0 \leq t_i \leq t_j \leq t_k \leq t_l \Rightarrow \mathbb{E}[(B_{t_j} - B_{t_i})(B_{t_l} - B_{t_k})] = 0$

Exemple d'équations éditées dans 2 blocs adjacents de style *Beamer* :

Coarse Cross-correlation

$$A_c^{(m)} = \sum_{i=0}^{N_c-1} u_k^{(m+i)} u_k^{*(m+i+N_c)}. \quad (7)$$

Fine Cross-correlation

$$A_f^{(m)} = \sum_{i=0}^{N_c-1} u_k^{(m+i)} u_k^{*(m+i+N_{f'})}. \quad (8)$$

2.4. Étude ou expérience 4

Exemple d'un texte édité sur deux colonnes adjacentes :

Voici un exemple de texte édité sur 2 colonnes dans le modèle de document de rapport de recherche IMT Atlantique. Ce modèle est édité en langage LaTeX adapté à la rédaction de documents scientifiques contenant, notamment, des équations et des figures.

2.5. Étude ou expérience 5

Exemple d'insertion de tableau :

Nom	Type	Nombre d'heures
FIG MTS 203P (com. num.)	Module	10
FIP RT323 (codage)	Module	10
FIP MGP320 (projet S5)	UV	15
FIG F4B301 (codage)	UV	20

TABLE 1 – Exemple de tableau (données factices)

3. Machine learning workflow

We are going to describe the Machine Learning workflow we chose. We will in particular explain our learning strategy and the metrics we used to evaluate our models.

3.1. Feature selection

As we can see on our datasets, especially the Chronic Kidney Disease's (CKD) one, there are a lot of features. Indeed, there are 24 columns for the features. This may be a problem for our neural network (or any classification algorithm) : having a high search dimension increases the complexity of the algorithm and its run-time. The neural network will also require a huge amount of data to converge without under-fitting. Therefore, it is essential to define a strategy to reduce the features' space. We have already dropped one column during the preprocessing phase, as more than 30% of its lines where empty. We will use a Principal Component Analysis (PCA) algorithm to drop redundant columns. We already have a standardize dataset thanks to the pre-processing step, which is essential to apply a PCA algorithm without suffering from high variance.

PCA is an unsupervised technique which aims to make the high variability of the data more visible by rotating the axes. It consists in the calculation of the eigenvalues and eigenvectors of our feature matrix. After that, it calculates the variance ratio of each rotated feature and ranges it in a decreasing order. We then select n features, depending on how much relevant information we want to keep. In our case, we defined a threshold of 90% of cumulative variance ratio. Performing a PCA on the CKD dataset yields to the selection of 10 columns, which allows us to reduce a lot the complexity of our dataset. For the Bank Note Dasaset, it reduces the features' dimension of 1.

3.2. Neural network model

As we deal with numerical data (nor images or temporal data), one of the most appropriate layers to use here is the dense layer (also called fully-connected layer). In order to get a better precision in the predictions, we are going to use 3 dense layers. The first 2 layers will use "ReLU" as an activation function, which is the most commonly used activation function in hidden layers. The last dense layer, composed of only one neuron (binary classification), will be followed by the common sigmoid activation function.

In our model (see annex X), you can notice 2 particular layers : the batch normalization one and the dropout one. Batch normalization standardizes the input (batch) of the layer, while the dropout layer allows us to ignore the prediction of a part of the neurons (only during training), 15% of them here. They both help our algorithm to avoid overfitting, to be more stable and to better generalize the results on another dataset.

Some parameters will be chosen later through a gridsearch (see next part). We decided to perform this strategy on the number of epoch, the batch size, and the optimizer. We won't perform gridsearch on the activation function for example as they may not have a very significant impact compared with the number of epochs or the batch size.

3.3. Training and testing strategy

A common practice is to split the dataset into a training, a validation and a test set. Each dataset has a precise function :

- Training set : train the weights of the neural network model chosen.

- Validation set : used as a test set to fine-tune the model (ie choose the best parameters or training strategy in order to get the best results).

- Test set : only used to test the final model, once we had optimize the training parameters with the validation test. It allows use to confirm the actual performance of our network.

Our strategy is here to divide the dataset in two : 0.75% for training/validation set, and 25% for the test set. We decided to take a bigger test set in order to challenge our model and to see if it can easily generalize

its results on an unseen dataset. On the training/validation set, we are going to train and fine-tune a model. We will do that in 2 main steps :

- Step 1 : K-Fold cross-validation and Gridsearch (Sklearn function) in order to find which parameters lead to the highest accuracy and the lowest loss. We will split the set in 10 folds, as it is a common value used for this kind of cross-validation. The Gridsearch will be performed on 3 relevant parameters : the number of epochs, the batch size and the kind of optimizer. Playing on the number of epochs and on the batch size will have an impact on the quality of the convergence and also on its speed. The choice of the optimizer depends on the dataset's characteristics.

- Step 2 : K-Fold cross-validation using the best parameters found in first step. We will keep the weights of the model which achieved the best accuracy and the lowest lost.

Once we have found our best model, we are going to train it on the whole dataset (it was trained on only 90% of the training/validation set) and then we are going to test it on the unseen test set to evaluate it.

3.4. Metrics

One of the most common metrics is the accuracy. In our network, this metric is easy to access through the sklearn functions. The highest the accuracy of a network is, the more its answers are correct. This metric is simple and intuitive. However, it has many drawbacks when we consider a dataset with unbalanced classes. Therefore, we need to look at other metrics to evaluate our model. We will look at the loss of our network, and at the precision, recall and F1-score. A small lost indicates that the network has a good convergence, so we want to minimize this parameter. The other 3 metrics' meaning depend on the numbers FP (false positive), FN (false negative), TN (true negative) and TP (true positive).

For example, in the CDK dataset, we want to minimize the FN, ie the number of person not considered as sick, while they really are sick. Therefore, it is essential for us to maximize the recall (which is equal to TP/(TP+FN). It is also good to maximize precision (ie minimize FP). As F1-score is a mean of precision and recall, it's also important to maximize F1-score.

4. Results : best model validation

We have tested our workflow using a RTX 2060 and having 8 Gb of memory. As our CPU is limited in memory, there are some tasks that we couldn't parallelize some tasks, for example the Gridsearch. Therefore, the fine-tuning phase takes a longer time. Applying our workflow takes approximately 10 minutes for each dataset.

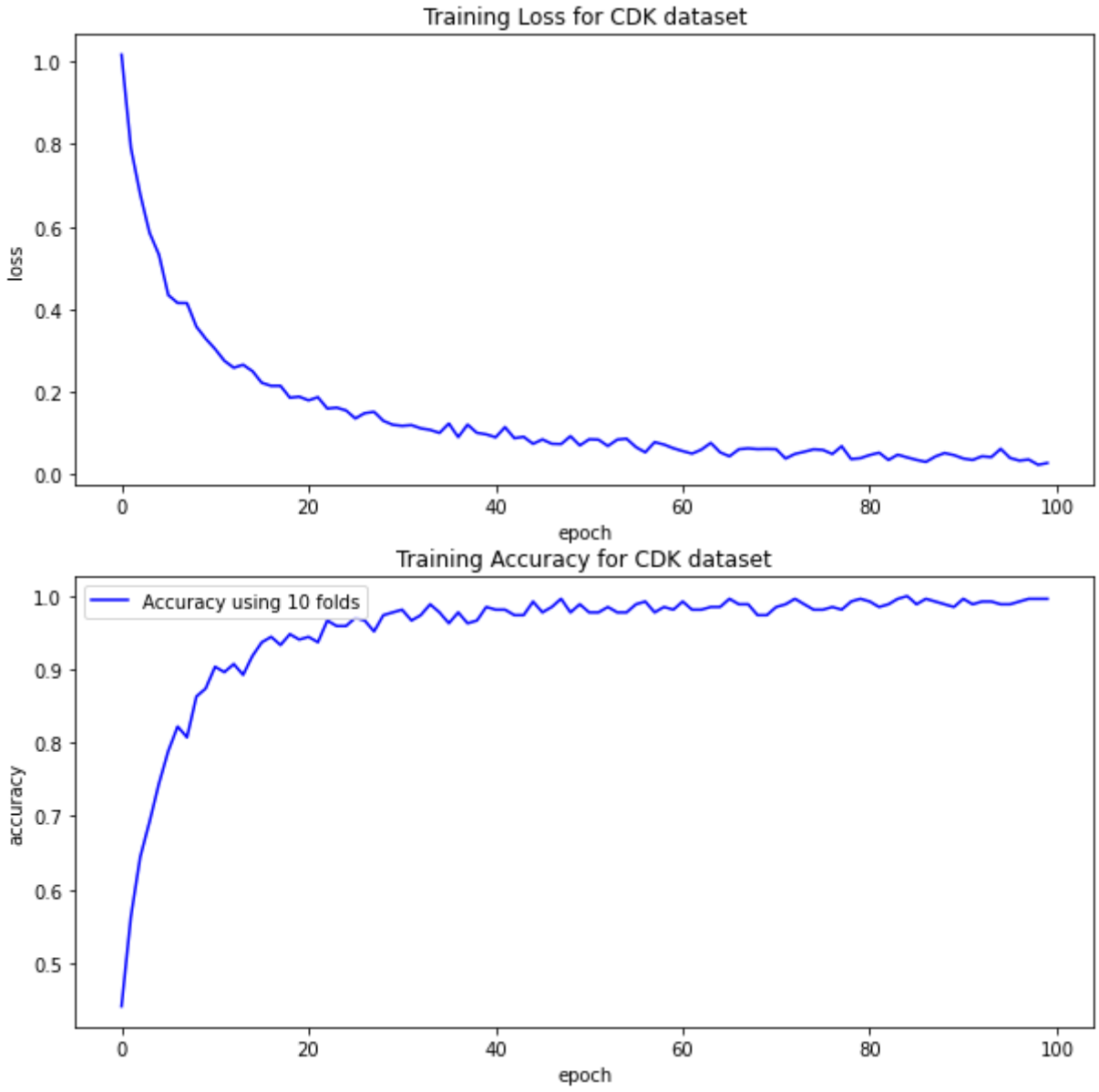
4.1. CKD Dataset

For each step of our training phase, we obtained the following results :

1. Fine-tuning : 100 for the best epoch number, 40 for the batch size and the best optimizer is 'Adamax'.
2. Best model : 99% of accuracy and a loss of 0.038. We also obtain the following scores :

	precision	recall	f1-score	support
0	0.98	1.00	0.99	64
1	1.00	0.97	0.99	36
accuracy			0.99	100
macro avg	0.99	0.99	0.99	100
weighted avg	0.99	0.99	0.99	100

Our neural network performed very well on this dataset. The accuracy is very high. Moreover, when we look at the row 0 of the classification report, which corresponds to the label "ckd", we notice hat the recall is equal to 1, which is a good news because this means that we never misclassified someone that has a CKD. Thus, we can conclude that this neural network is very powerful for this dataset.



If we take a look at the learning curves, we can see that both loss and accuracy have converged to their final value. That means that we do not need to train our network on this dataset anymore (we won't get significantly better results by doing so).

4.2. Banknote Dataset

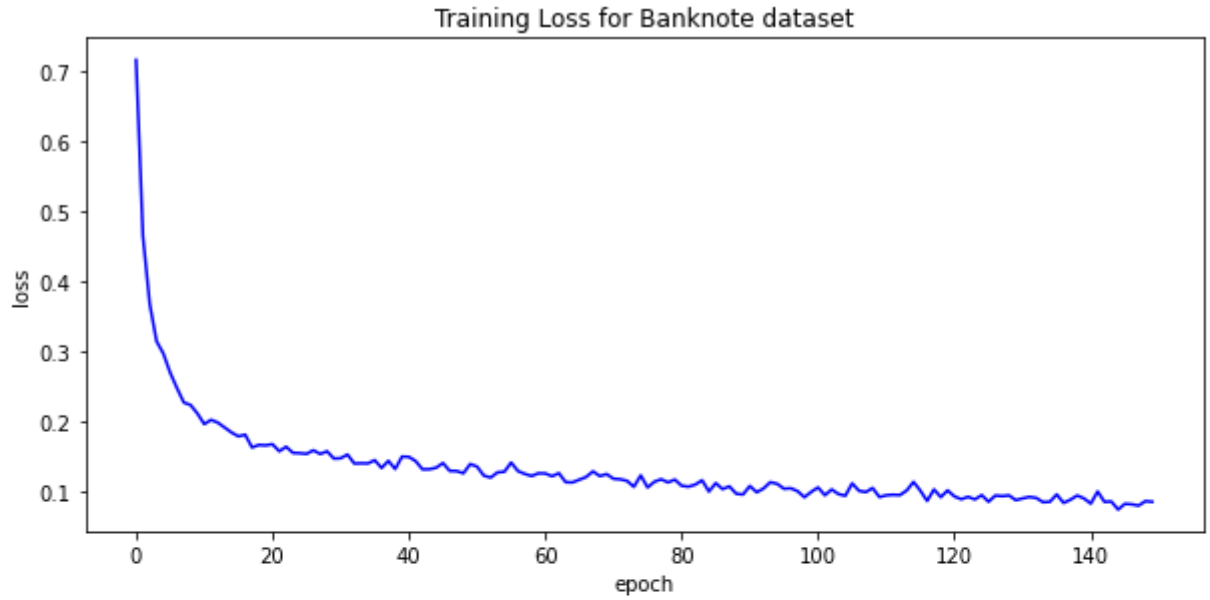
For each step of our training phase, we obtained the following results :

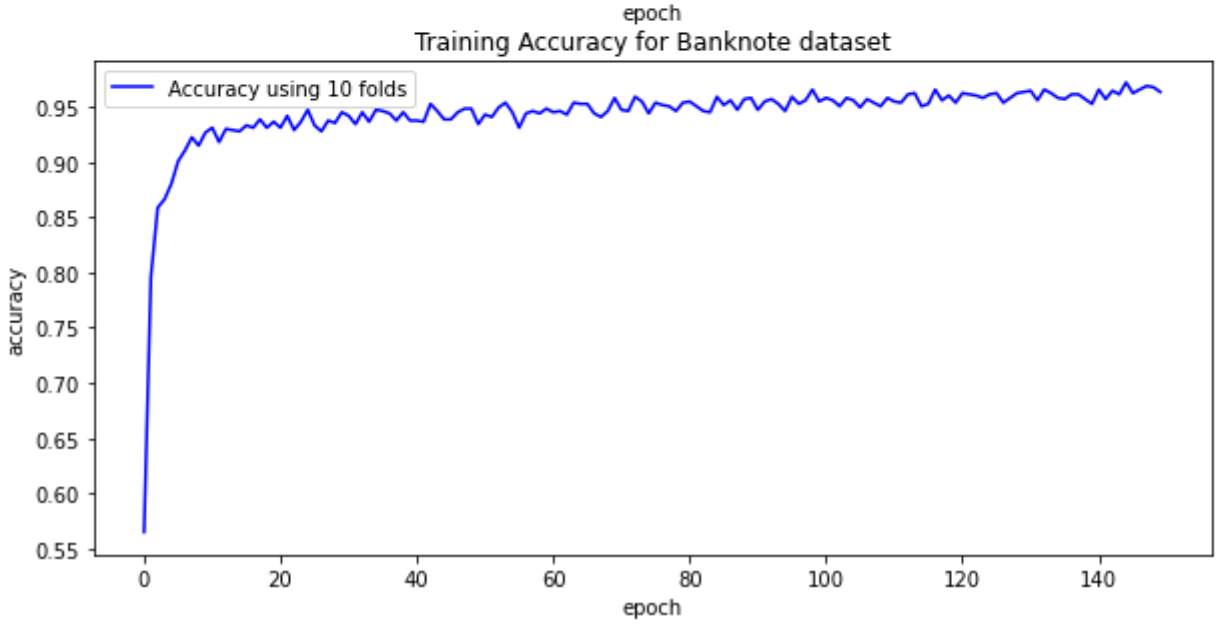
1. Fine-tuning : 150 for the best epoch number, 100 for the batch size and the best optimizer is 'Adam'.

2. Best model : 95% of accuracy and a loss of 0.08. We also obtain the following scores :

	precision	recall	f1-score	support
0	0.98	0.93	0.95	191
1	0.91	0.98	0.95	152
accuracy			0.95	343
macro avg	0.95	0.95	0.95	343
weighted avg	0.95	0.95	0.95	343

Our neural network did not perform that well on this dataset. We expected a better accuracy, even if it is not that bad. The scores from the classification report are correct, but it could have been better.

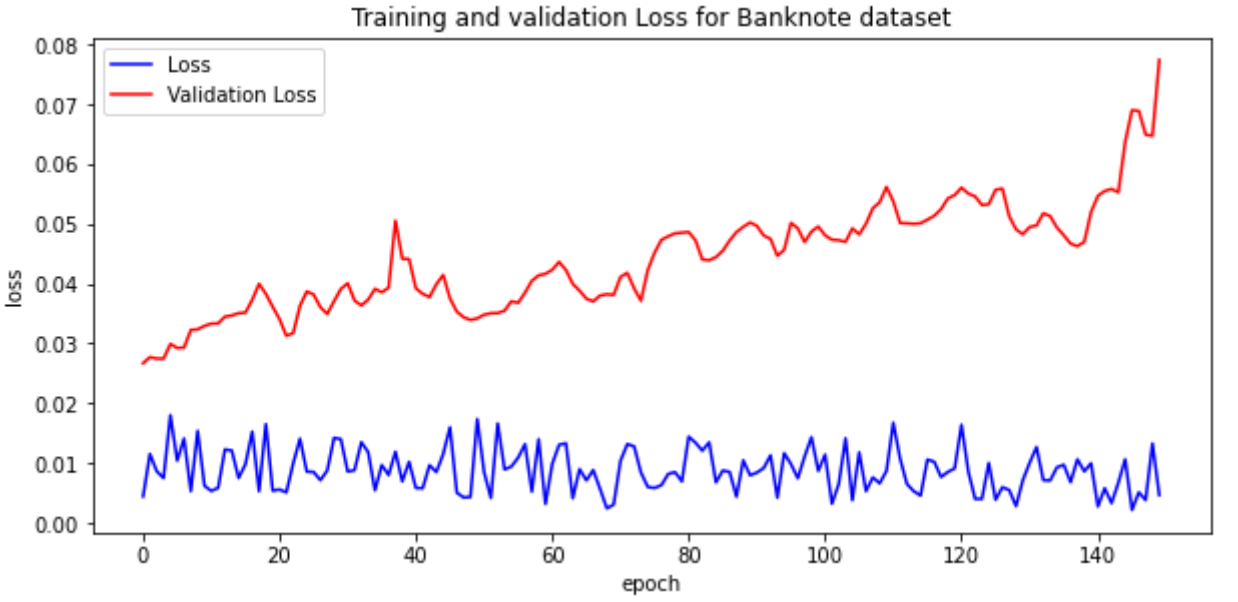




If we take a look at the learning curves, we can see that both loss and accuracy haven't totally converged. Indeed, the curves are still increasing when approaching epoch 150. This means that our network may need a little more time to learn.

Therefore, we tried to launch again the training phase for 1000 epochs. The accuracy is better (97,3%) and the scores are also better. This confirms the fact that the network needed more time to learn. However, we think that the network can be modified in order to make it converge faster. Moreover, even if we keep increasing the number of epochs, we don't get a better accuracy than 97%. We can probably find a model that will be more powerful for this dataset.

We can also improve the analyse of our network for this dataset. Indeed, we can see that the training accuracy (99%) is higher than the test accuracy. We can wonder whether our network is overfitting or not. To do that, we need to modify our function *best-model-fit-eval()* and add a validation step (so we also need to create a validation set). Then, we need to compare the training and the validation curve. When the validation loss begins to increase, that means that the model is overfitting. Let's apply this to our banknote strategy. We obtain the following curves :



That is exactly what is happening here. We can see that before training on the whole dataset, the validation loss already is above the training loss. Thus, our model is severely overfitting.

This highlights one of the major problems of our model. Indeed, it may lead to overfitting. Maybe we can solve this by adding more dropouts or by making a simpler model (less layers for example).

5. Conclusion and good practices to adopt

<Placer le texte ici>

5.1. Résultat 1

D'après [**?**], la structure d'une présentation de résultats est composée des éléments suivants :

1. description du résultat (ce qu'il faut observer) ;
2. discussion du résultats (commenter les observations) ;
3. conclusions sur le résultat (que peut-on en déduire ?).

Annexes

Annexe 1 – Exemple d'annexe

Un exemple d'annexe.

1.1. Première partie

1.1.1. Sous-section

1.1.2. Sous-section

1.2. Deuxième partie

1.2.1. Sous-section

1.2.2. Sous-section

Annexe 2 – Autre annexe

Un autre exemple d'annexe.

2.1. Première partie

2.1.1. Sous-section

2.1.2. Sous-section

2.2. Deuxième partie

2.2.1. Sous-section

2.2.2. Sous-section

OUR WORLDWIDE PARTNERS UNIVERSITIES - DOUBLE DEGREE AGREEMENTS

3 CAMPUS, 1 SITE

IMT Atlantique Bretagne–Pays de la Loire – **http://www.imt-atlantique.fr/**

Campus de Brest
Technopôle Brest-Iroise
CS 83818
29238 Brest Cedex 3
France
T +33 (0)2 29 00 11 11
F +33 (0)2 29 00 10 00

Campus de Nantes
4, rue Alfred Kastler
CS 20722
44307 Nantes Cedex 3
France
T +33 (0)2 51 85 81 00
F +33 (0)2 99 12 70 08

Campus de Rennes
2, rue de la Châtaigneraie
CS 17607
35576 Cesson Sévigné Cedex
France
T +33 (0)2 99 12 70 00
F +33 (0)2 51 85 81 99

Site de Toulouse
10, avenue Édouard Belin
BP 44004
31028 Toulouse Cedex 04
France
T +33 (0)5 61 33 83 65

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Imprimé à IMT Atlantique
Dépôt légal : Septembre 2017
ISSN : 2556-5060