Advanced Deep Learning

TP1 Exercise 4 : Regression Problem: Wind Production Prediction

Table des matières

1 Exercise 1: Impact of the architecture of the model

To perform this regression task, the model is composed of fully connected layers with variable activation functions, variable number of neurons and variable number of layers. The number of neurons is fixed for the entire architecture otherwise it would inflate too much the number of combinations. The output dimension is 1 and no activation function is applied on the last layer.

- Layers : 2,3,4,5,10
- Activation Functions : relu, sigmoid, tanh, CELU, hardtanh
- Number of neurons per layers : 5, 10, 20, 50, 100, 1000

By default the MSE is minimized through SGD with a learning rate of 0.01. The number of training epoch is 10 and the batch size is 10. A model is created for each combination of parameters and the latters are tested on an evaluation set of 1000 samples, the MSE is computed.

The following figures display the evolution of MSE for a few fixed combinations of parameters when the number of layers or the number of neurons increases.

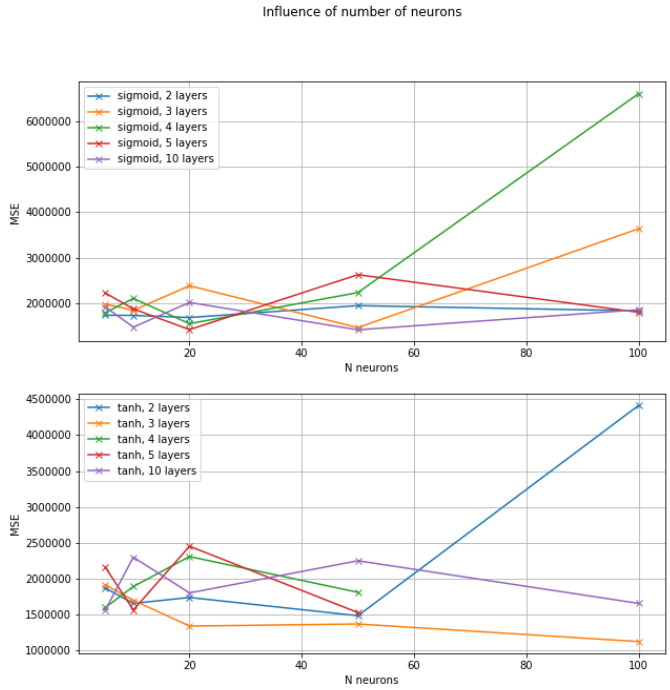


Fig. 1.1: Neurons

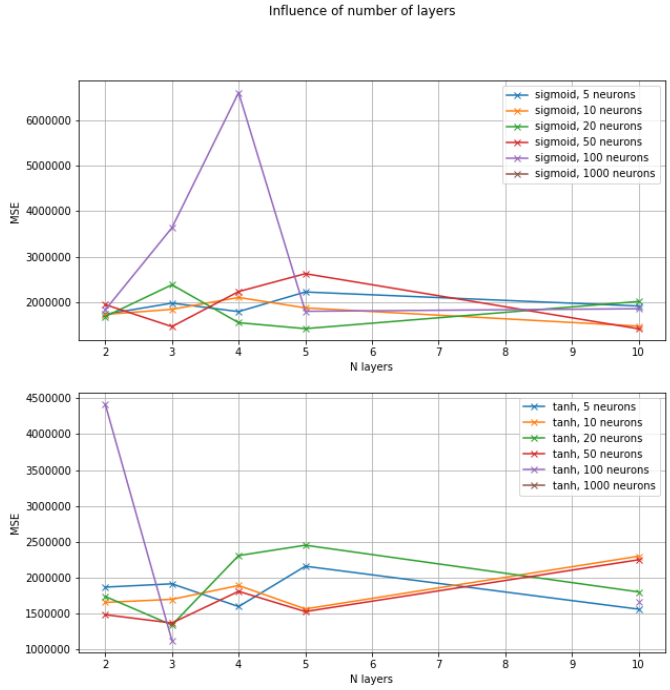


Fig. 1.2: Layers

From these plots we can conclude that complexifing the architecture of a model either by adding neurons or layers does not improve its performances at least under

10 epochs and the basic optimizer. Some combinations simply do not perform and yield nan values mainly with the activation function ReLU and CeLU. Under the current framework, the architectures with the best MSE are the following:

- MSE : $1,11e^6$,Layers : 2 ,Activation : Hardtanh, Neurons : 50
- MSE : $1,12e^6$,Layers : 3 ,Activation : tanh, Neurons : 100
- MSE : $1,30e^6$,Layers : 3 ,Activation : tanh, Neurons : 100
- MSE : $1,30e^6$,Layers : 3 ,Activation : Hardtanh, Neurons : 20
- MSE : $1,31e^6$,Layers : 3 ,Activation : Hardtanh, Neurons : 20
- MSE : $1,34e^6$,Layers : 4 ,Activation : Hardtanh, Neurons : 20

The loss per epoch are displayed for each of those combinations:

Fig. 1.3: Best combinations

There is no obvious convergence and the predictive power of our model on the evaluation set is poor. Overall, 10 train epochs is too few to expect convergence in this problem and the optimizer is not suited, thus the necessity of solving this issue.

2 Exercise 2: Impact of the optimizer

As modifying the architecture is not enough to yield decent performance we tackle the training step. We both test more subtle optimizing algorithms with different learning rates as well different training environments.

Firstly, we observe the impact of the learning rate for a standard architecture (3 layers, 20 neurons, tanh, 20 epochs, batch size 10). Given that the number of epochs will not be outrageously high We infer that a large learning rate (0.5-10) is preferable for all optimizers but SGD. Fig 2.1b) shows the impact of the batch size for a standard architecture (3 layers, 50 neurons, tanh, 25 epochs, lr 1) for different optimizers . For Adadelta it is critical to have a higher batch size whereas for other optimizers the batch size is not a significant hyperparameters (lower batch size tend to be slightly better). We will stick to 10 and 400 from now on. Fig 2.1c) the impact of the number of epochs for a standard architecture (3 layers, 50 neurons, tanh, batch size 400, lr 1) for different optimizers. Under this set of parameters, our evaluation error only significantly decreases when the number of epochs increases for the Adadelta optimizer. Nevertheless we will choose a number of epochs above 50 epochs to ensure convergence and below 100 to prevent overfitting.

Finally the parameter space having been roughly reduced we test all the following combinations to find the most efficient ones:

- Layers : 4,10
- Activation Functions : sigmoid, tanh, hardtanh
- Number of neurons per layers : 10, 50, 100
- Batch Size : 10,400
- Optimizer : Adadelta, Adam, SGD, RMSprop
- Learning Rate : 0.1, 0.5, 1, 10
- Epochs : 50,100

Fig 2.2 shows the evolution of the epoch per loss for the 6 best models on the evaluation set. The MSE has been substantially reduced to 300 000 compared to the first exercice. Adadelta with a learning rate of 0.5 is the best performing optimizer.

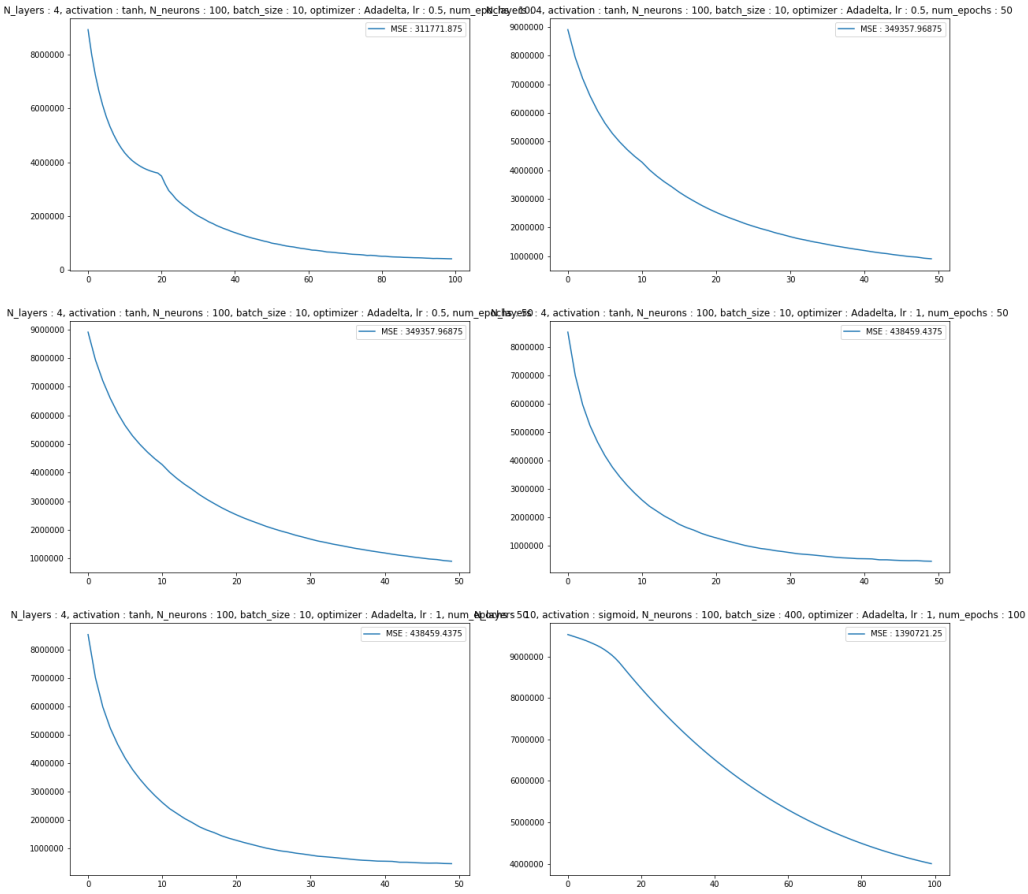
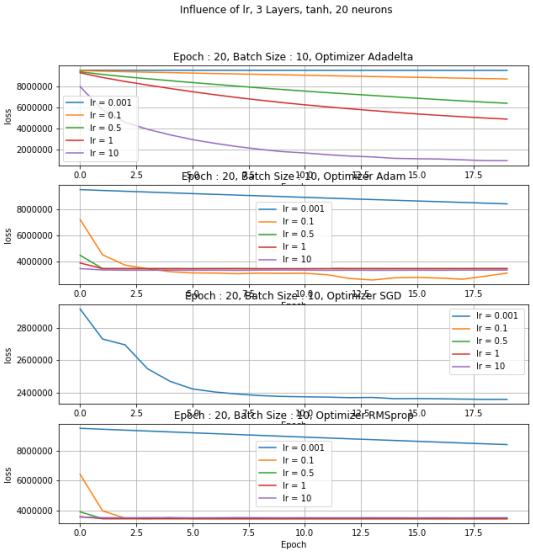
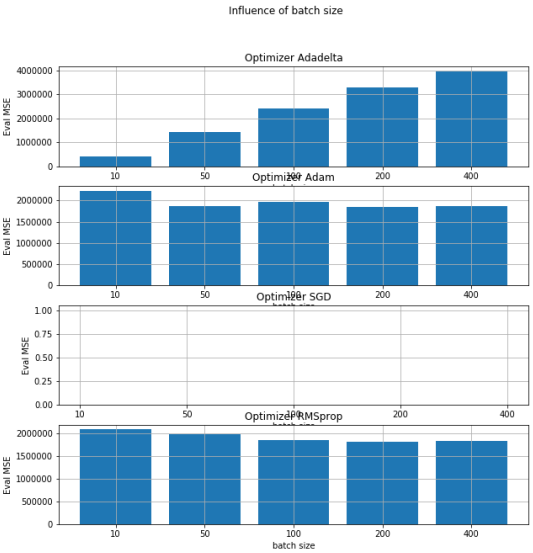

N_layers : 4, activation : tanh, N_neurons : 100, batch_size : 10, optimizer : Adadelta, lr : 0.5, num_epochs : 100
MSE : 311771.875
N_layers : 4, activation : tanh, N_neurons : 100, batch_size : 10, optimizer : Adadelta, lr : 0.5, num_epochs : 50
MSE : 349357.96875
N_layers : 4, activation : tanh, N_neurons : 100, batch_size : 10, optimizer : Adadelta, lr : 0.5, num_epochs : 50
MSE : 349357.96875
N_layers : 4, activation : tanh, N_neurons : 100, batch_size : 10, optimizer : Adadelta, lr : 1, num_epochs : 50
MSE : 438459.4375
N_layers : 4, activation : tanh, N_neurons : 100, batch_size : 10, optimizer : Adadelta, lr : 1, num_epochs : 50
MSE : 438459.4375
N_layers : 10, activation : sigmoid, N_neurons : 100, batch_size : 400, optimizer : Adadelta, lr : 1, num_epochs : 100
MSE : 1390721.25

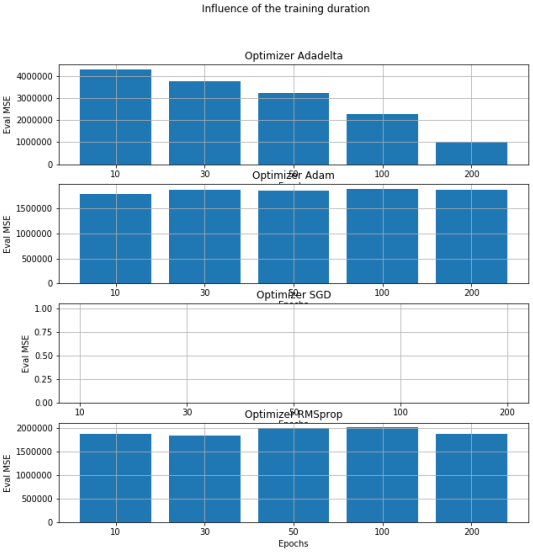
Fig. 2.1: Best



(a) Learning Rate



(b) Batch Size



(c) Epochs

3 Exercise 3: Impact of the loss function

Instead of using the standard MSE loss derived from the assumption that the conditional likelihood is gaussian with variance 1 and mean $f(x_i)$ we now use a Gaussian likelihood function where the model outputs gaussian distribution parametrized by $\sigma(x_i)^2$ and $\mu(x_i)$.

The conditional likelihood becomes:

$$p(y_i|x_i) = \frac{1}{\sqrt{2\pi\sigma(x_i)^2}} e^{-\frac{(y_i-\mu(x_i))^2}{2\sigma(x_i)^2}}$$

Hence we define the following loss function:

$$L = \sum_{i=1}^{N} \frac{1}{2}\log(2\pi\sigma_i^2) + \frac{1}{2\sigma_i^2}(y_i - \mu_i)^2$$

We therefore change our architecture to output both the $\log(\sigma(x_i)^2)$ and $\mu(x_i)$ still without applying any activation to the last layer. The following combinations are tested with more complex architectures to allow a better estimation $\log(\sigma(x_i)^2)$:

- Layers : 4,10,20
- Activation Functions : sigmoid, tanh
- Number of neurons per layers : 10, 50, 100
- Batch Size : 10
- Optimizer : Adadelta, RMSprop
- Learning Rate : 0.5
- Epochs : 100

Fig 3.1 shows the evolution of the average loss per epoch (from epoch 2 to last epoch) for the 6 best models on the evaluation set. The same behaviour is observed regardless of the architecture : between the first and second epoch, the average loss function drops drastically from 50k to 200 and then slowly decreases. This loss is unarguably better suited to the shape and prior information of the data than the MSE. The best loss obtained on the evaluation set is for the layers, , neurons, Adadelta w. 0.5 lr

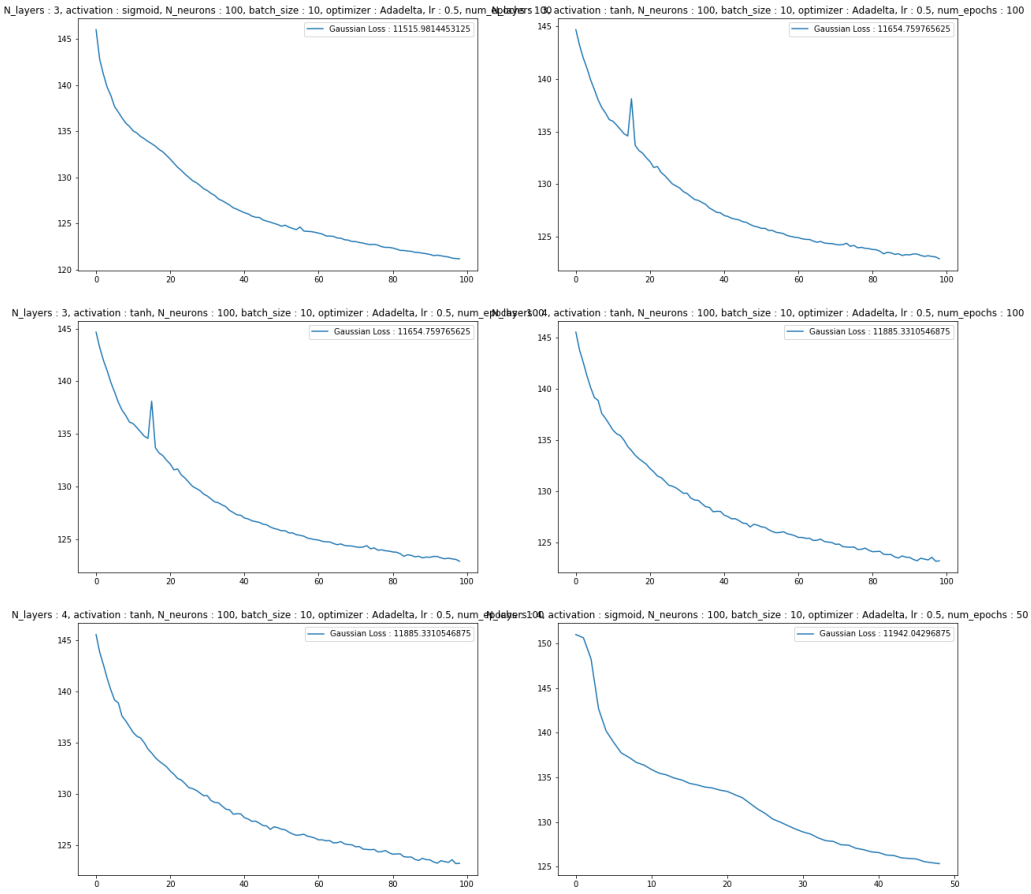

N_layers : 3, activation : sigmoid, N_neurons : 100, batch_size : 10, optimizer : Adadelta, lr : 0.5, num_epochs : 100
Gaussian Loss : 11515.9814453125
N_layers : 3, activation : tanh, N_neurons : 100, batch_size : 10, optimizer : Adadelta, lr : 0.5, num_epochs : 100
Gaussian Loss : 11654.759765625
N_layers : 3, activation : tanh, N_neurons : 100, batch_size : 10, optimizer : Adadelta, lr : 0.5, num_epochs : 100
Gaussian Loss : 11654.759765625
N_layers : 4, activation : tanh, N_neurons : 100, batch_size : 10, optimizer : Adadelta, lr : 0.5, num_epochs : 100
Gaussian Loss : 11885.3310546875
N_layers : 4, activation : tanh, N_neurons : 100, batch_size : 10, optimizer : Adadelta, lr : 0.5, num_epochs : 100
Gaussian Loss : 11885.3310546875
N_layers : 4, activation : sigmoid, N_neurons : 100, batch_size : 10, optimizer : Adadelta, lr : 0.5, num_epochs : 50
Gaussian Loss : 11942.04296875

Fig. 3.1: Best

4 Exercise 4: Prediction on test set

We train a final model using the following parameters and the gaussian likelihood loss function. The outputs of the model are still $\sigma(x_i)^2$ and $\mu(x_i)$.

- Layers : 10
- Activation Functions : sigmoid
- Number of neurons per layers : 200
- Batch Size : 10
- Optimizer : Adadelta
- Learning Rate : 0.5
- Epochs : 100

The gaussian likelihood loss on the test set is 21866.

We now plot the $\sigma(x_i)^2$ representing the uncertainty of the model regarding the prediction against the three input variables. The uncertainty regarding the prediction clearly increases when the windspeed increases, the overall uncertainty decreases when the horizontal radiation increases. When looking at the scatter plot of Radiation vs Production (fig 4.2) the distribution of production value gets narrower as the Radiation increases, the opposite is also true for WindSpeed vs Production. Thus our model captures well these behaviours. Nothing can be said about the temperature.

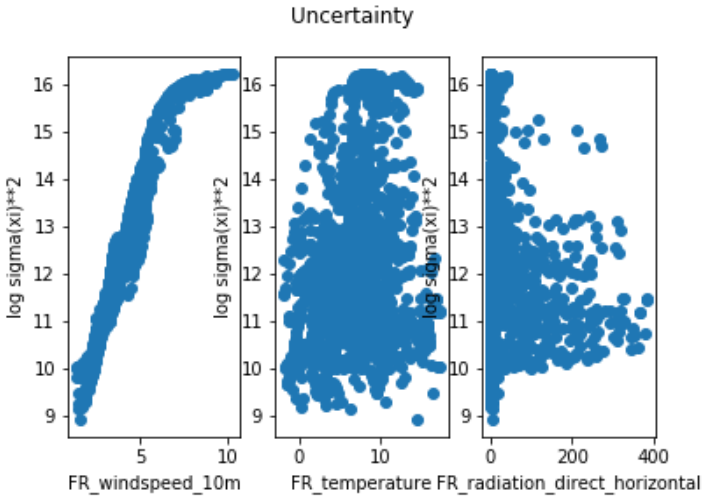


Fig. 4.1: Uncertainty

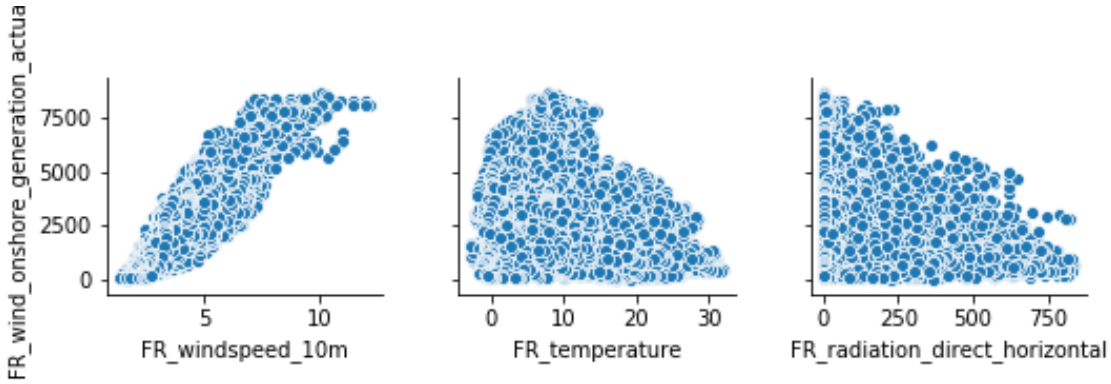


Fig. 4.2: Scatter