Authors: Burca Horia, Arias Rodrigo
Date: 19/11/2018

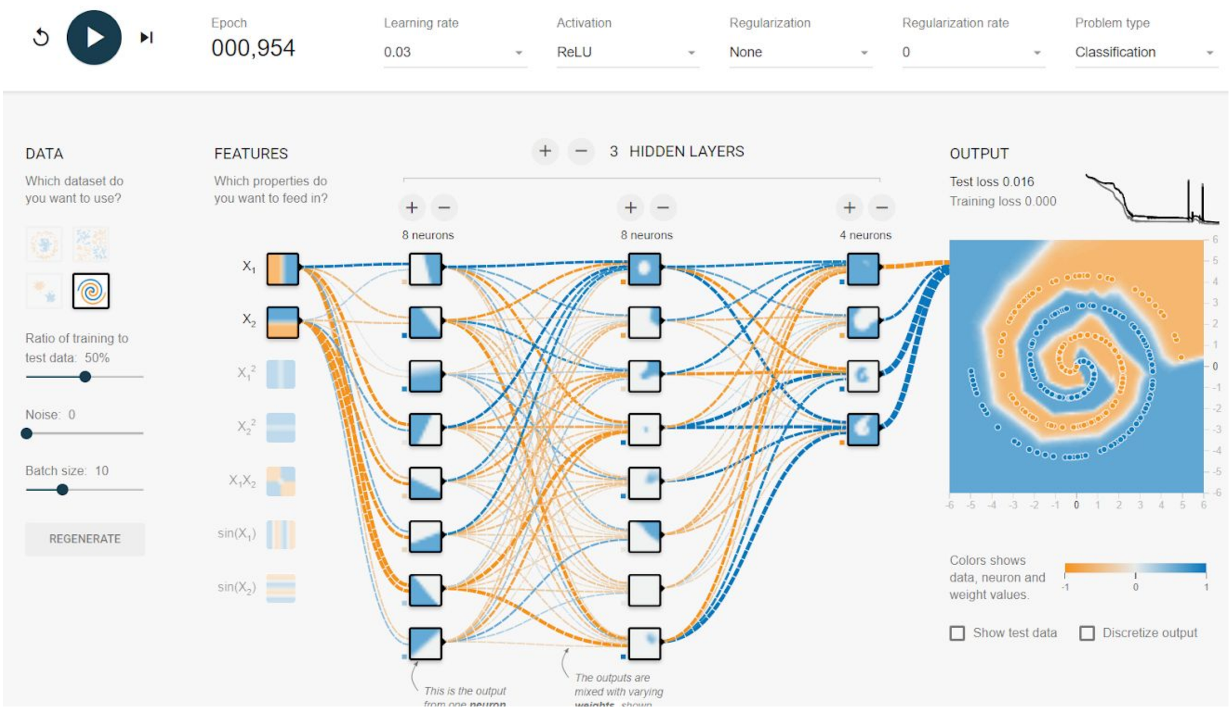
Lab 8: Get started with DL hyperparameters

Approach

In this lab we try to construct an efficient neural network for the spiral dataset in the Google Tensorflow Playground. First of all we use the *ReLu* activation function as it is one of the best performing functions at the moment. We also set the learning rate to *0.03* to give us a decent total training time but at the same time a reduced chance of missing the optimum. Finally, we use only x_1 and x_2 input features as our data is 2D.

The main focus is on the number of hidden layers and the number of neurons per respective layer. We know that for a 2D space a single neuron classifies the data points into two kinds with a straight line. Consequently, we can deduce straight away that due to the geometry of the dataset a single layer will not suffice. We need more layers in order to construct more complex features that can in turn be used in the output to classify the data. We have performed the following steps:

1. **We add *3* hidden layers with *8*, *8* and *4* respective neurons** to get a first model that successfully classifies the data.


Epoch 000,954
Learning rate 0.03
Activation ReLU
Regularization None
Regularization rate 0
Problem type Classification

DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10

FEATURES
Which properties do you want to feed in?

3 HIDDEN LAYERS
8 neurons
8 neurons
4 neurons

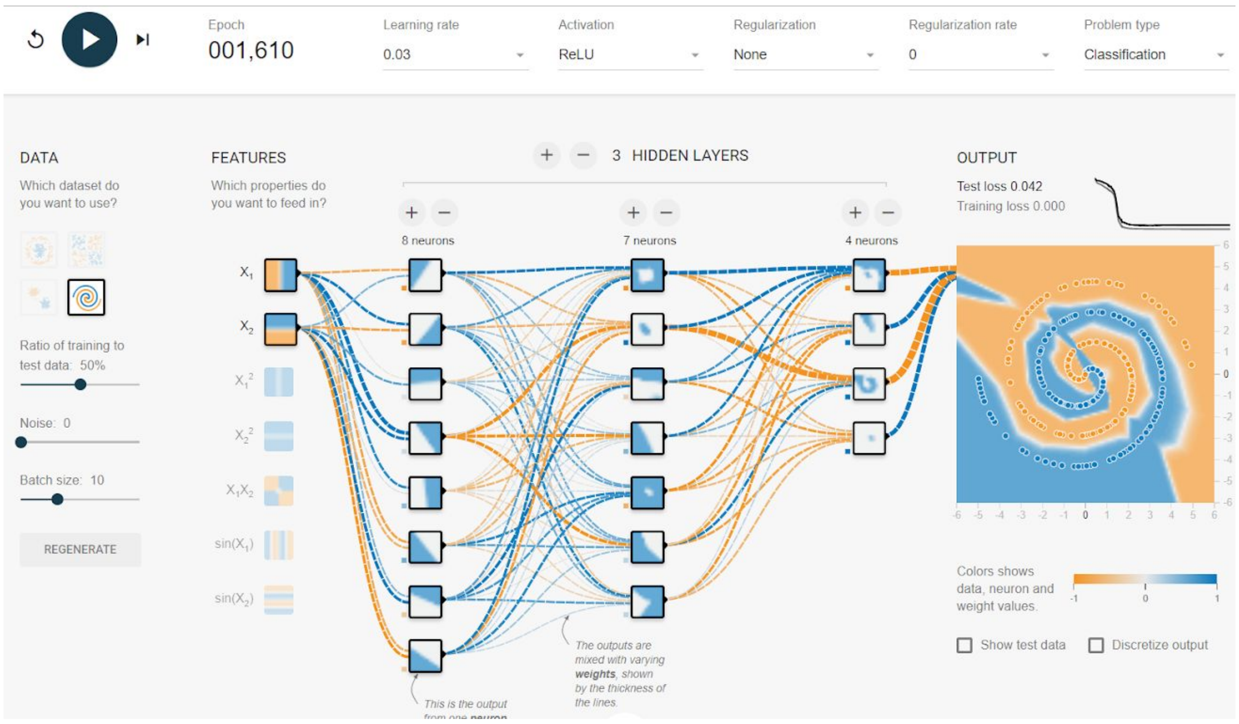
OUTPUT
Test loss 0.016
Training loss 0.000

Colors shows data, neuron and weight values.
REGENERATE
X1
X2
X1²
X2²
X1X2
sin(X1)
sin(X2)
-1
0
1
Show test data
Discretize output
This is the output from one neuron
The outputs are mixed with varying weights, shown

We notice that the first layer outputs simple features, data classified by a straight line. However, the following layers combine these features to produce more complex ones. We know that the thickness of the lines connecting the nodes represents their weight in the input of the following layer. With this in mind, we notice that the outputs of some neurons in the hidden layers are practically not used in the following layer. So we can try to reduce the model.

Authors: Burca Horia, Arias Rodrigo
Date: 19/11/2018

2. **We reduce the model to *3* hidden layers with *8*, *7* and *4* respective neurons.**



Epoch 001,610
Learning rate 0.03
Activation ReLU
Regularization None
Regularization rate 0
Problem type Classification

Ratio of training to test data: 50%
Noise: 0
Batch size: 10

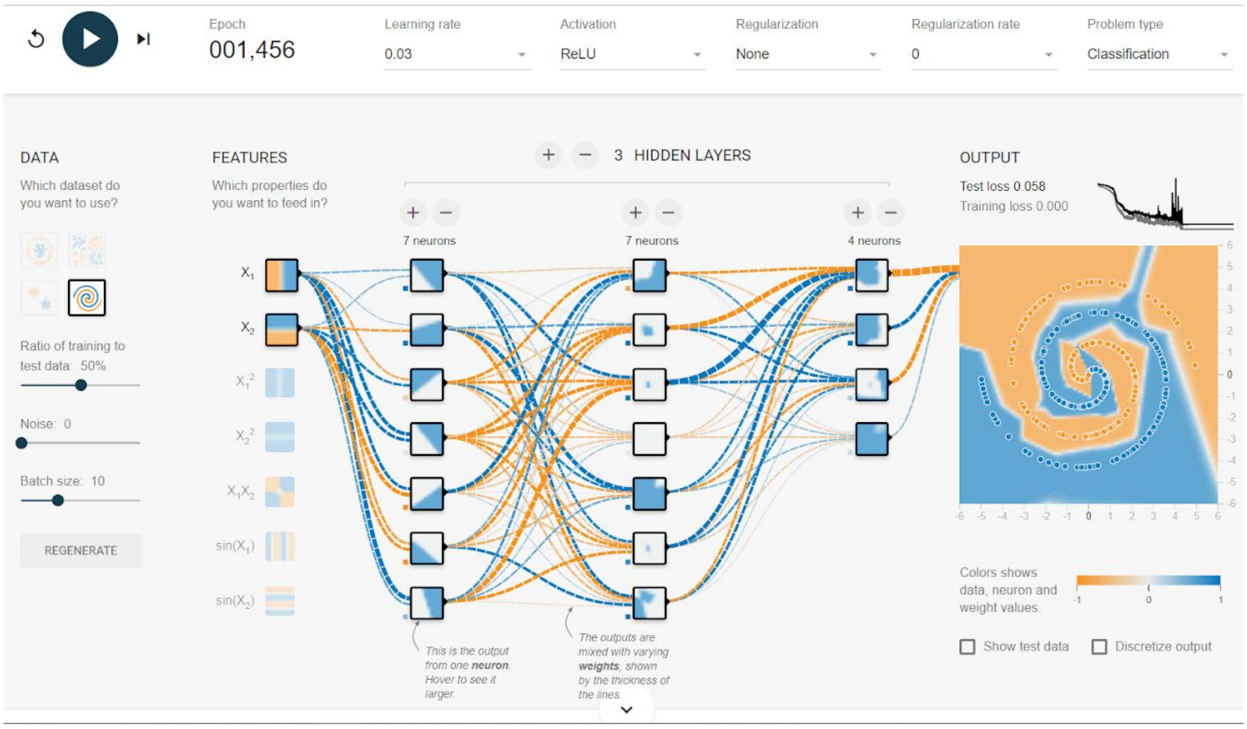
3 HIDDEN LAYERS
8 neurons
7 neurons
4 neurons

Test loss 0.042
Training loss 0.000

DATA
Which dataset do you want to use?
REGENERATE
FEATURES
Which properties do you want to feed in?
X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$
OUTPUT
Colors shows data, neuron and weight values.
-1
0
1
Show test data
Discretize output
The outputs are mixed with varying weights, shown by the thickness of the lines.
This is the output from one neuron

We still get a good test accuracy. At the same time we once again notice that some neurons are not used, so we can further reduce.

3. **We reduce the model to *3* hidden layers with *7*, *7* and *4* respective neurons.**



Epoch 001,456
Learning rate 0.03
Activation ReLU
Regularization None
Regularization rate 0
Problem type Classification

Ratio of training to test data: 50%
Noise: 0
Batch size: 10

3 HIDDEN LAYERS
7 neurons
7 neurons
4 neurons

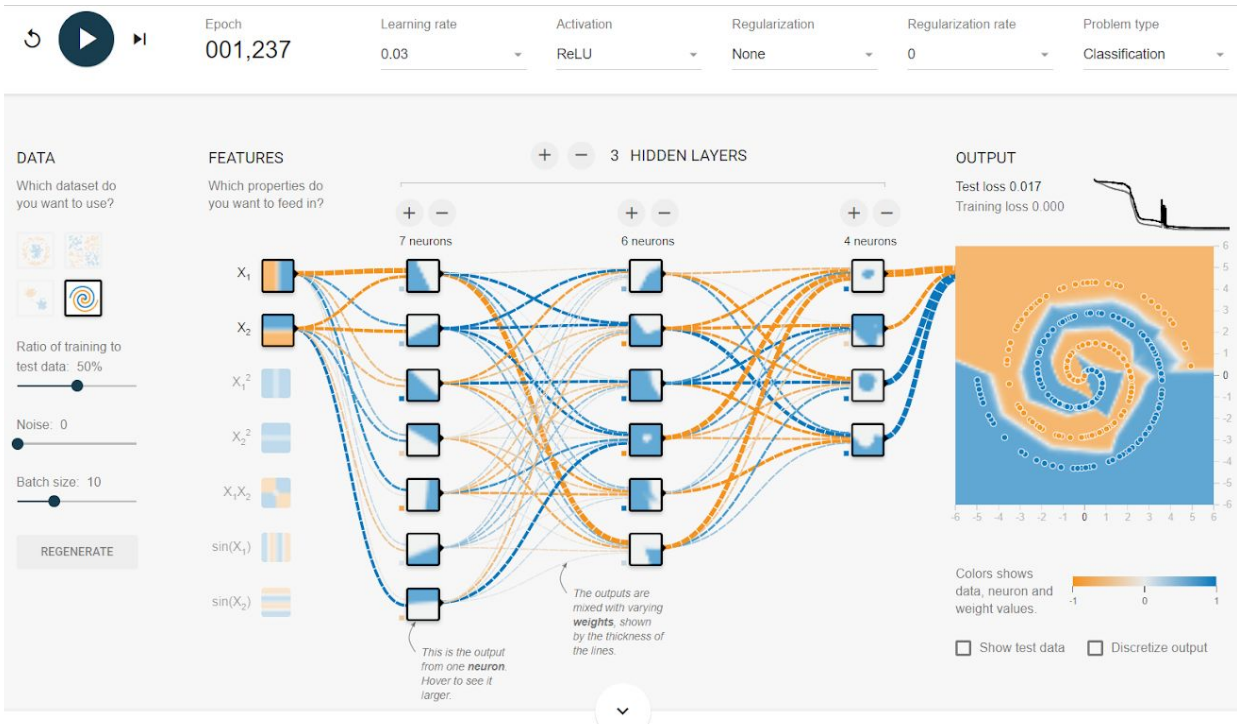
Test loss 0.058
Training loss 0.000

DATA
Which dataset do you want to use?
REGENERATE
FEATURES
Which properties do you want to feed in?
X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$
OUTPUT
Colors shows data, neuron and weight values.
-1
0
1
Show test data
Discretize output
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.

We still get a good test accuracy. At the same time we once again notice that some neurons are not used, so we can further reduce. There is in fact a totally blank neuron in the second layer which does no transformation on the data.

Authors: Burca Horia, Arias Rodrigo
Date: 19/11/2018

4. **We reduce the model to *3* hidden layers with *7*, *6* and *4* respective neurons.**



Epoch 001,237 | Learning rate 0.03 | Activation ReLU | Regularization None | Regularization rate 0 | Problem type Classification

DATA — Which dataset do you want to use? Ratio of training to test data: 50% Noise: 0 Batch size: 10

FEATURES — Which properties do you want to feed in?

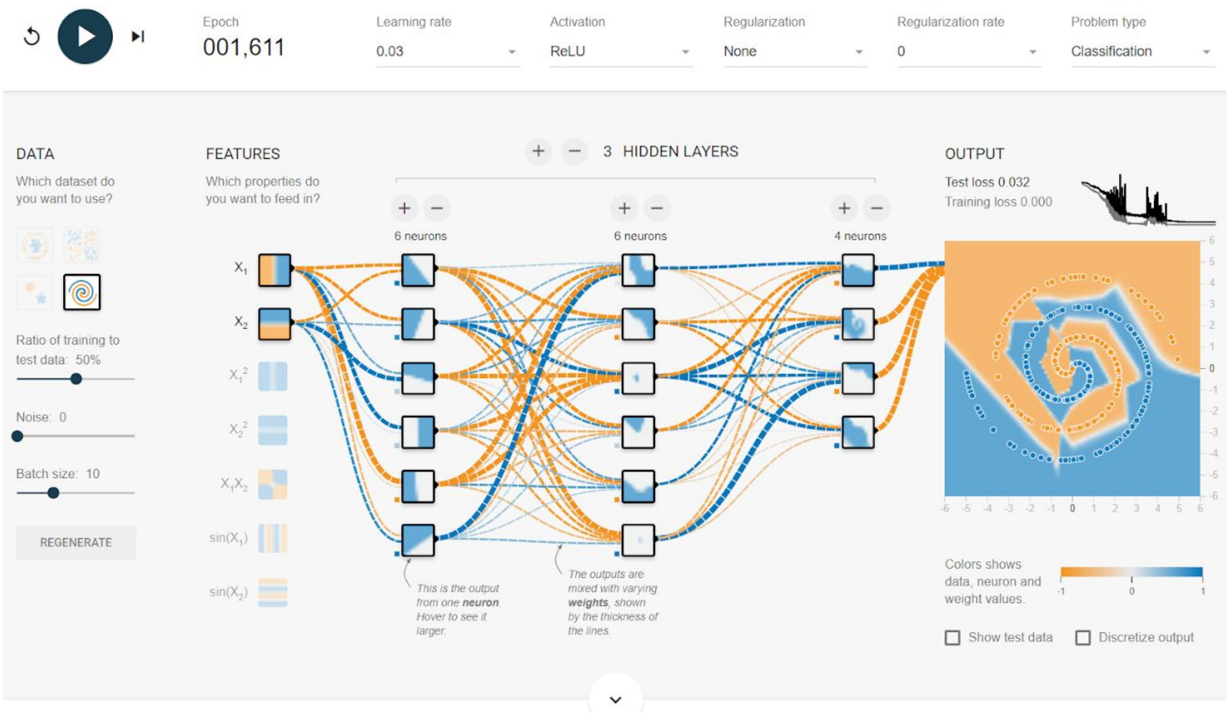
3 HIDDEN LAYERS — 7 neurons, 6 neurons, 4 neurons

OUTPUT — Test loss 0.017 Training loss 0.000

REGENERATE
X1
X2
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
Colors shows data, neuron and weight values.
Show test data
Discretize output

We still get a good test accuracy. We can see that in the second and third layers all the weights of the outputs of the neurons are substantially larger than 0. At the same time we notice that some neurons in the first layer are not used, so we can further reduce.

5. **We reduce the model to *3* hidden layers with *6*, *6* and *4* respective neurons.**



Epoch 001,611 | Learning rate 0.03 | Activation ReLU | Regularization None | Regularization rate 0 | Problem type Classification

DATA — Which dataset do you want to use? Ratio of training to test data: 50% Noise: 0 Batch size: 10

FEATURES — Which properties do you want to feed in?

3 HIDDEN LAYERS — 6 neurons, 6 neurons, 4 neurons

OUTPUT — Test loss 0.032 Training loss 0.000

REGENERATE
X1
X2
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
Colors shows data, neuron and weight values.
Show test data
Discretize output

At this point all the neurons are more or less used in the network. We obtain a decent accuracy. If we try to further reduce the model we no longer get a good enough classification of the data.

Authors: Burca Horia, Arias Rodrigo
Date: 19/11/2018

Conclusion

In this lab we saw how we can derive a Deep Neural Network that can classify complex datasets. We could experience the influence of the different hyperparameters on the performance of the model. We observed how neural networks try to extract pertinent features from the dataset in order to solve the problem and the role that the hidden layers and the neurons have in building this abstraction.