Exercise 3: Neural Playground

Task 1

Architecture:

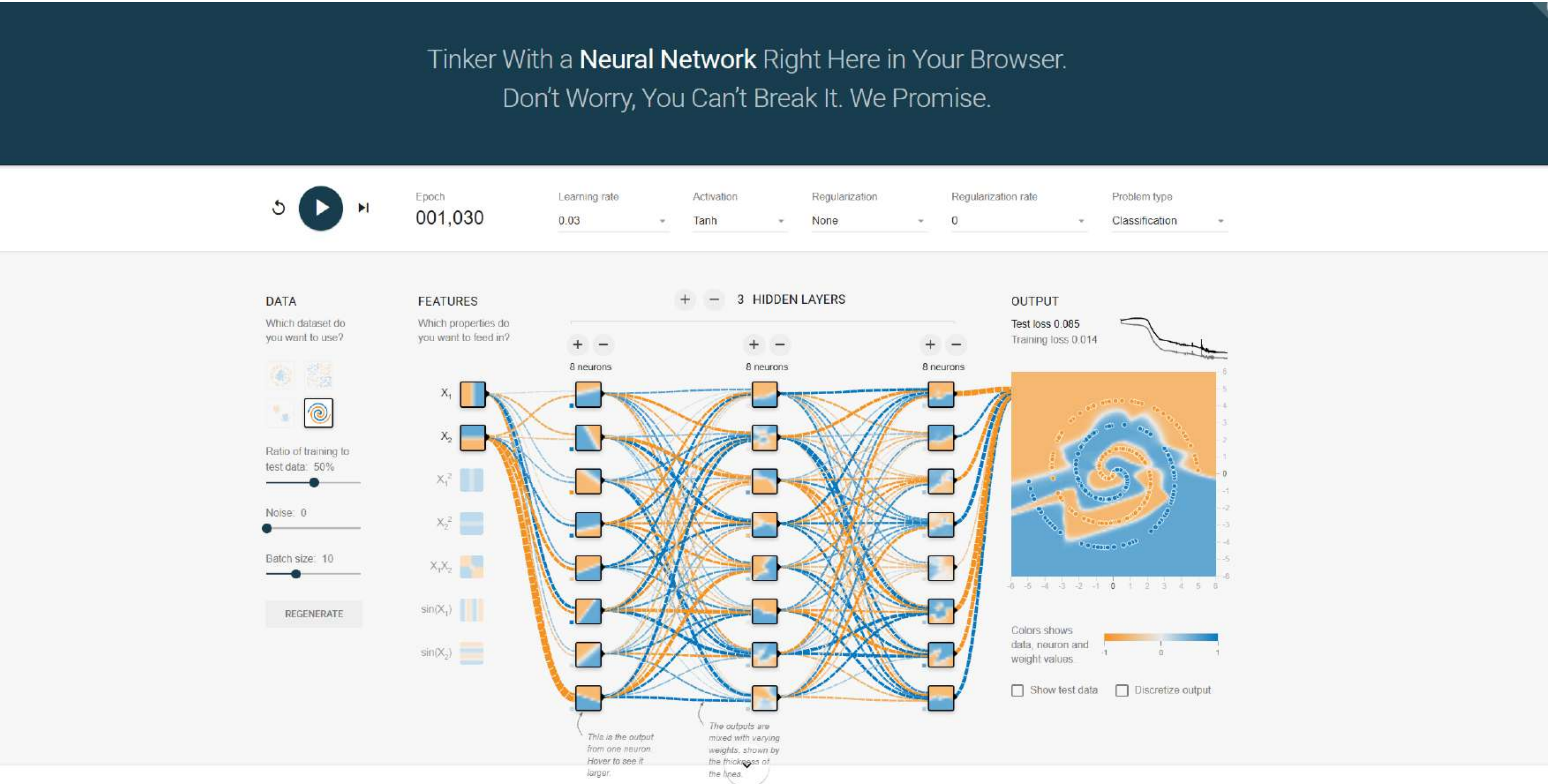
- Hidden Layers: 3
- Neuron per layer: 8 neurons in each layer
- Activation Function: $tanh$ for all layers
- Learning: 0.1
- Input features: $X1$ and $X2$
- Regularization: None

Why this architecture?

- The spiral dataset is highly nonlinear and to learn its twisting pattern might require multiple layers.
- I've tried with 2 layers, 4 and 5 layers which did not give a good loss that 3 layers did. I believe it finds a balance between underfitting and overfitting at layer 3.
- The $tanh$ activation function worked better than ReLU or sigmoid. I believe because it can handle positive and negative input ranges which the spiral pattern requires.
- I tried with different learning rate but 0.1 gave me the fastest convergence without overshooting.

Outcome:

- Achieved a test loss <= 0.1 after 500 epochs and reached a constant 0.02 test loss after 800 epochs.


Tinker With a **Neural Network** Right Here in Your Browser.
Don't Worry, You Can't Break It. We Promise.
Epoch
001,030
Learning rate
0.03
Activation
Tanh
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
REGENERATE
FEATURES
Which properties do you want to feed in?
X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$
+ − 3 HIDDEN LAYERS
8 neurons
8 neurons
8 neurons
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.
OUTPUT
Test loss 0.085
Training loss 0.014
Colors shows data, neuron and weight values.
-1
0
1
Show test data
Discretize output

Task 2

Experimental Setup:

We used the Spiral dataset with all 7 available input features:

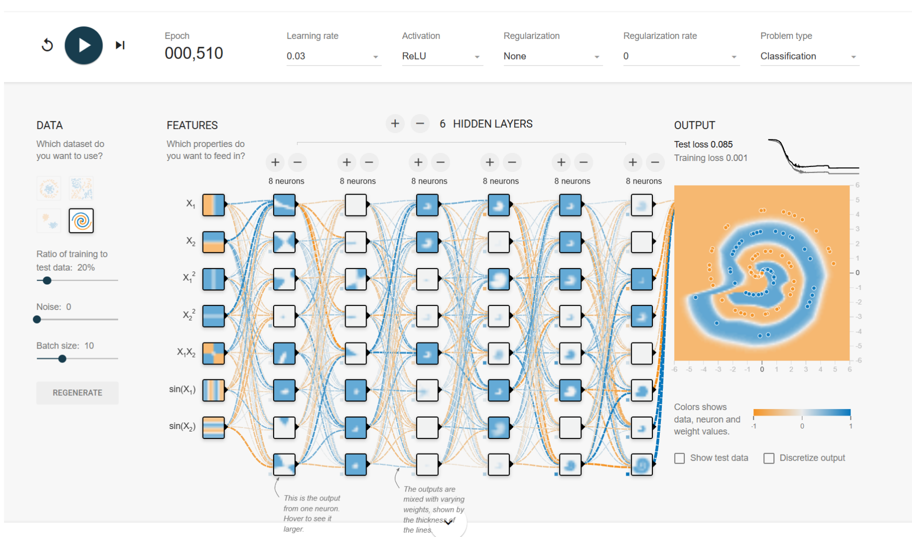
$$X_1,\ X_2,\ X_1^2,\ X_2^2,\ X_1X_2,\ \sin(X_1),\ \sin(X_2).$$

The network architecture consisted of 6 hidden layers with 8 neurons each, using the ReLU activation function. The learning rate was set to 0.03, and the training set size was restricted to 20% of the total data (80% test data). We conducted three experiments: without regularization, with L1 regularization, and with L2 regularization.

(a) No Regularization

Settings: Regularization disabled.

Observations: The model overfits quickly due to limited data availability. The training loss drops very close to zero, but the test loss remains relatively high.
(**Test loss: 0.085**).
The decision boundary follows the spiral quite well but showed signs of overfitting small fluctuations. Most neurons had strong activations (many thick connection lines).

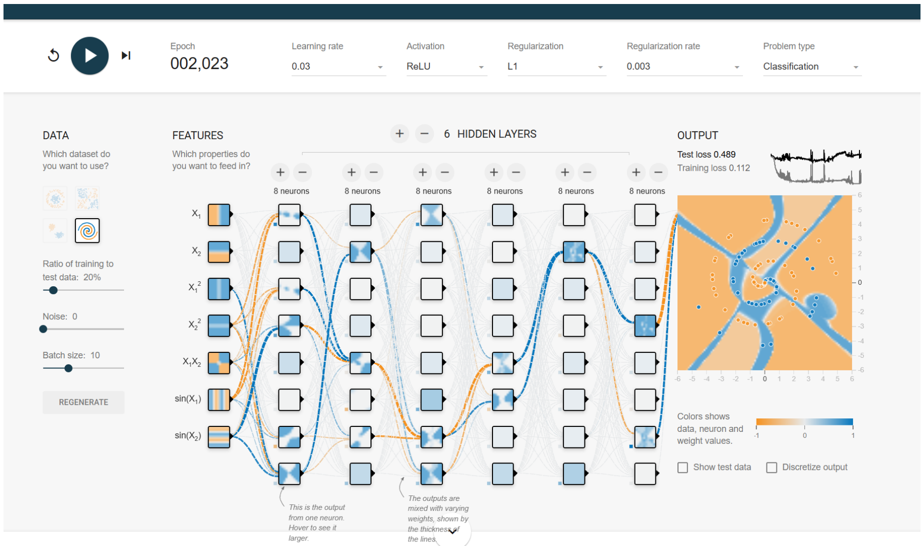


(b) L1 Regularization

Settings: L1 regularization enabled, regularization rate = 0.003.

Observations: L1 regularization results in many connections becoming inactive (thin or missing lines), which shows sparsity. The model simplifies the

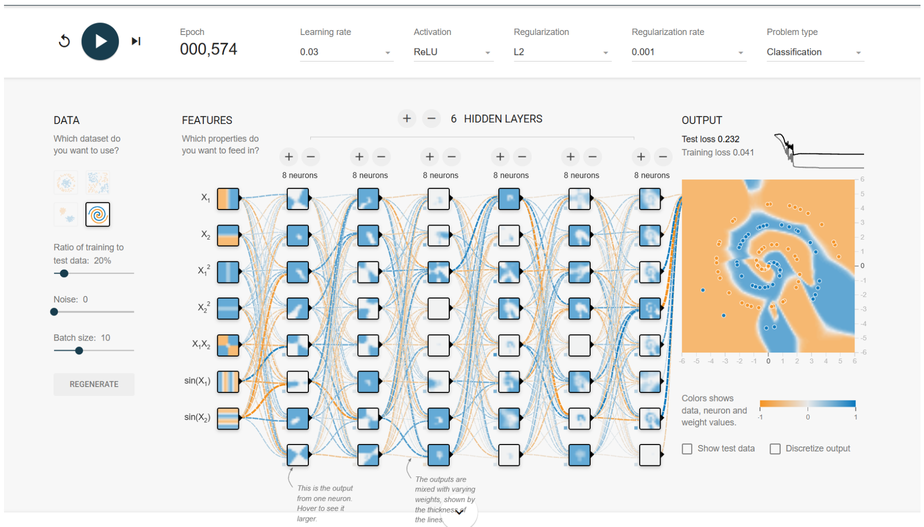
decision boundary considerably but lost its flexibility to fully model the spiral pattern, which has lead to its underfitting.
(**Test loss: 0.489**)
The network sacrifices model complexity to enforce sparse weights.



(c) L2 Regularization

Settings: L2 regularization enabled, regularization rate = 0.001.

Observations: L2 regularization seems to shrink weights more evenly across the network (visible as thinner lines, but no entirely disconnected neurons). The decision boundary still is the spiral but its in a smoother fashion compared to no regularization. Generalization has been improved, which has resulted in lower test loss compared to L1 regularization.
(**Test loss: 0.232**).



So In Summary:

- **No regularization:** Complex decision boundary, risk of overfitting with small training data.
- **L1 regularization:** Promoted sparsity, simplified network structure but underfits the highly nonlinear data.
- **L2 regularization:** Balanced complexity and generalization, which lead to the smooth decision boundary and best test performance in this setup.