Report on Neural Networks Using TensorFlow Playground

Aaron David, Angel Candelas, Monica Joya, Saif UR Rehman, and Varit Kobutra

Houston Community College

Author Note

This report is being submitted on June 30, 2024, for Professor Patricia McManus for Computer Vision ITAI 1378: 12461 at Houston Community College by Aaron David, Angel Candelas, Monica Joya, Saif UR Rehman, and Varit Kobutra.

Introduction to Neural Networks

Neural networks are computational models inspired by the human brain, consisting of interconnected nodes (neurons) organized in layers. These networks are capable of learning complex patterns through training on data. The key components of a neural network include:

- **Neurons**: Basic units that receive input, process it, and pass the output to the next layer.
- **Layers**: Organized groups of neurons, including input, hidden, and output layers.
- **Weights and Biases**: Parameters that are adjusted during training to minimize error.
- **Activation Functions**: Functions applied to the output of neurons to introduce non-linearity, enabling the network to learn complex patterns.

Neural networks are significant in various applications, including image recognition, natural language processing, and predictive analytics (NCBI, 2022).

Exploration Phase Using TensorFlow Playground

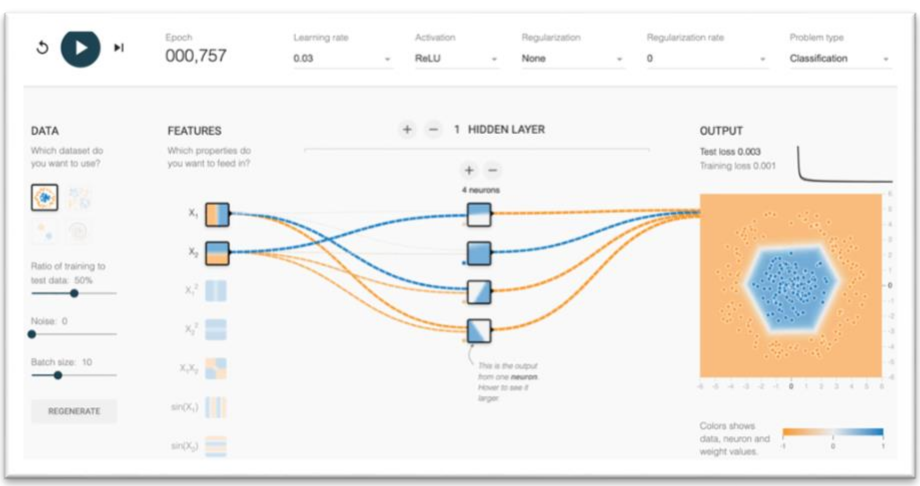
Access the TensorFlow Playground at playground.tensorflow.org.

Task 1: Activation Functions

Experiment: Create a neural network with one hidden layer and experiment with different activation functions (ReLU, sigmoid).

Figure 1

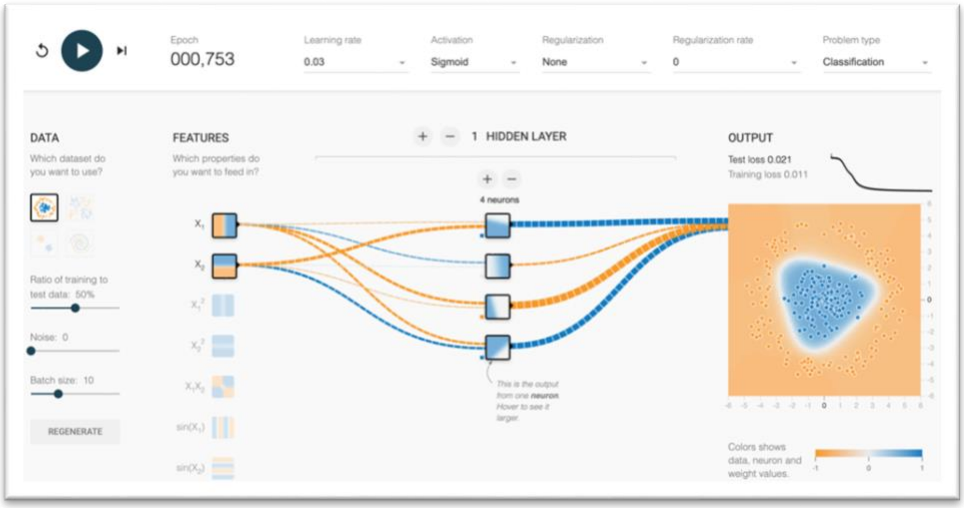
Neural Network with ReLU Activation Function



Note. Screenshot from TensorFlow Playground (https://playground.tensorflow.org). The network uses one hidden layer with 4 neurons and the ReLU activation function.

Figure 2

Neural Network with Sigmoid Activation Function



Note. Screenshot from TensorFlow Playground (https://playground.tensorflow.org). The network uses one hidden layer with 4 neurons and the Sigmoid activation function.

Explanation

- **Activation Functions**: These functions determine whether a neuron should be activated by transforming the weighted sum of inputs. Common activation functions include:
  - **ReLU (Rectified Linear Unit)**: Outputs the input directly if positive, otherwise zero. It helps mitigate the vanishing gradient problem and speeds up training.
  - **Sigmoid**: Maps input to a value between 0 and 1, useful for binary classification but prone to vanishing gradients (Jeong et al., 2020).

Observations

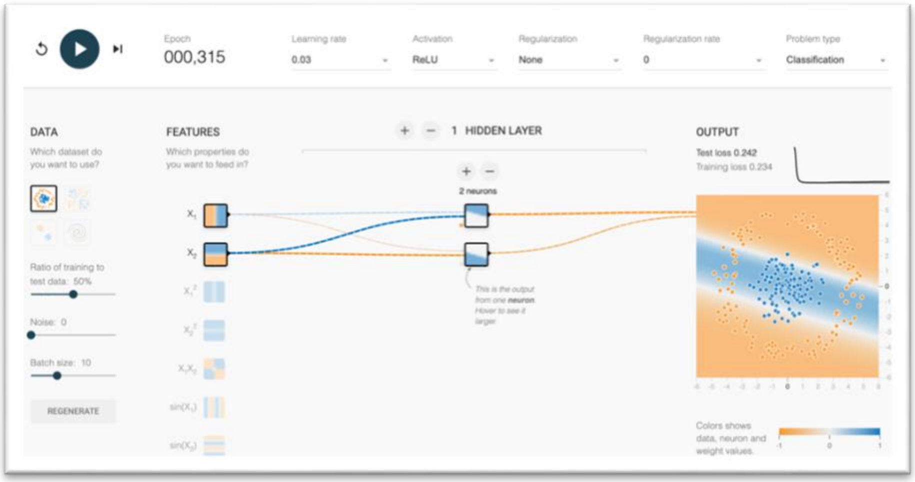
- **ReLU**: Faster convergence and better performance on complex patterns due to non-saturating gradients.
- **Sigmoid**: Slower convergence and potential issues with vanishing gradients, making it less effective for deep networks.

Task 2: Hidden Layer Neurons

Experiment: Change the number of neurons in the hidden layer and the number of hidden layers.

Figure 3

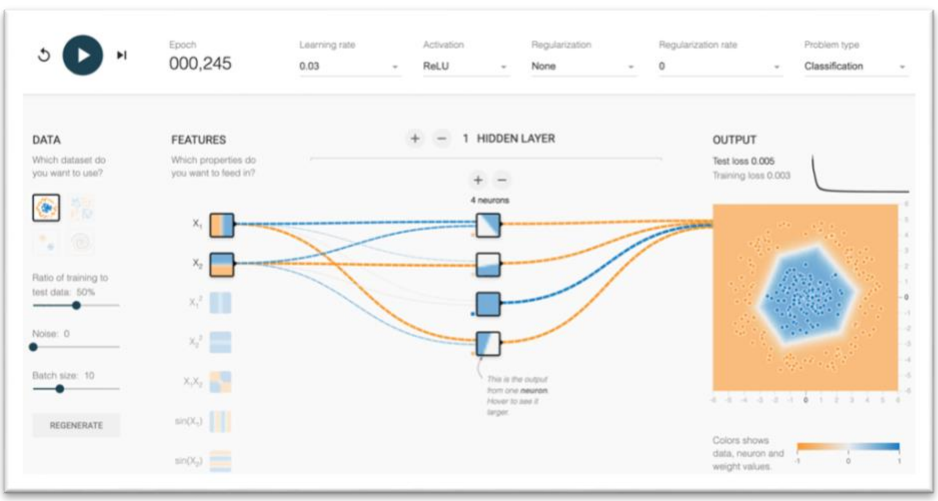
Neural Network with 1 Hidden Layer and 2 Neurons



Note. Screenshot from TensorFlow Playground (https://playground.tensorflow.org). The network uses one hidden layer with 2 neurons.

Figure 4

Neural Network with 1 Hidden Layer and 4 Neurons



Note. Screenshot from TensorFlow Playground (https://playground.tensorflow.org). The network uses one hidden layer with 4 neurons.

Explanation

- **Neurons and Hidden Layers**: Neurons in hidden layers perform computations on inputs, and more neurons/layers can model more complex functions. However, too many neurons/layers can lead to overfitting (Jeong et al., 2020).

Observations

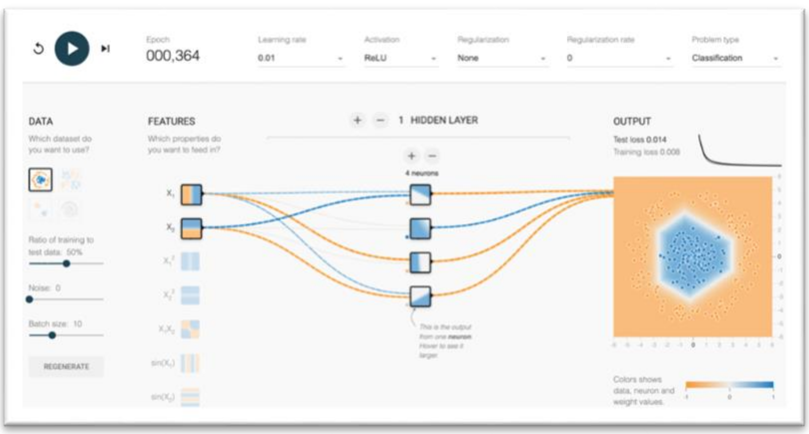
- Increasing neurons/layers improved performance up to a point, after which overfitting occurred.
- Optimal configuration balanced complexity and generalization.

Task 3: Learning Rate

Experiment: Adjust the learning rate and observe its impact on convergence speed and accuracy.

Figure 5

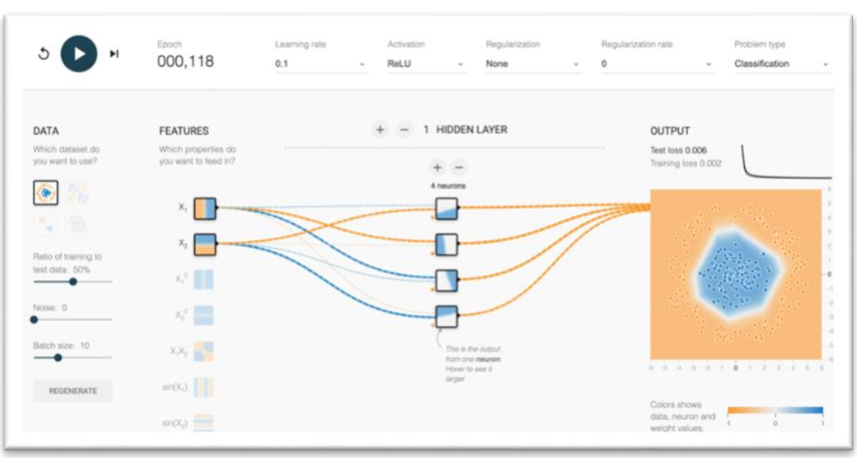
Neural Network with Learning Rate 0.01



Note. Screenshot from TensorFlow Playground (https://playground.tensorflow.org). The network uses one hidden layer with 4 neurons and a learning rate of 0.01.

Figure 6

Neural Network with Learning Rate 0.1



Note. Screenshot from TensorFlow Playground (https://playground.tensorflow.org). The network uses one hidden layer with 4 neurons and a learning rate of 0.1.

Explanation

- **Learning Rate**: Controls the step size during weight updates. A high learning rate can lead to overshooting, while a low rate can slow convergence (Jeong et al., 2020).

Observations

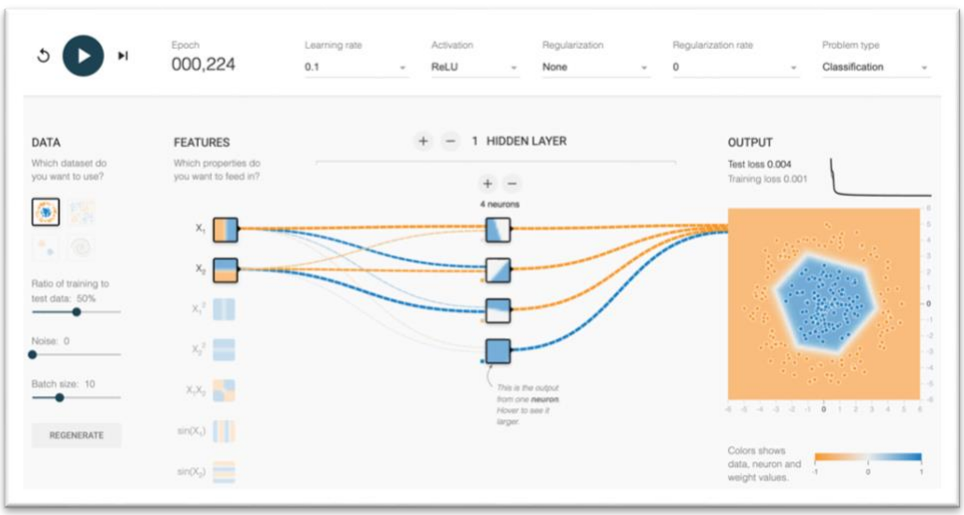
- Moderate learning rates achieved the best balance between speed and accuracy.
- Extremely high or low rates resulted in poor performance.

Task 4: Data Noise

Experiment: Introduce noise in the data and observe its effect on the network's generalization ability.

Figure 7

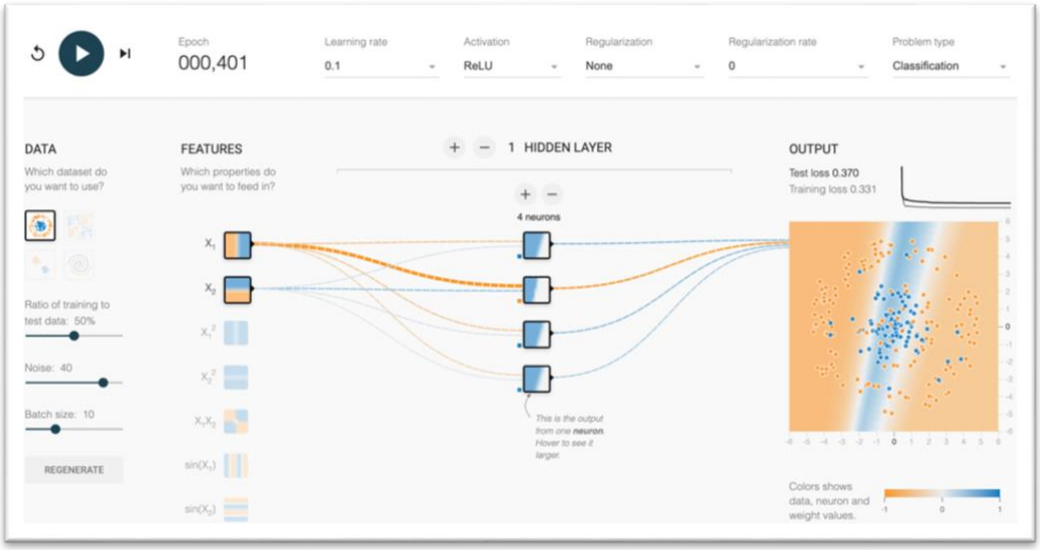
Neural Network with Noise Level 0



Note. Screenshot from TensorFlow Playground (https://playground.tensorflow.org). The network uses one hidden layer with 4 neurons and a noise level of 0.

Figure 8

Neural Network with Noise Level 40



Note. Screenshot from TensorFlow Playground (https://playground.tensorflow.org). The network uses one hidden layer with 4 neurons and a noise level of 40.

Explanation

- **Data Noise**: Random variations in data that can affect model training. Noise can help prevent overfitting, but too much noise can degrade performance (NCBI, 2022).

Observations

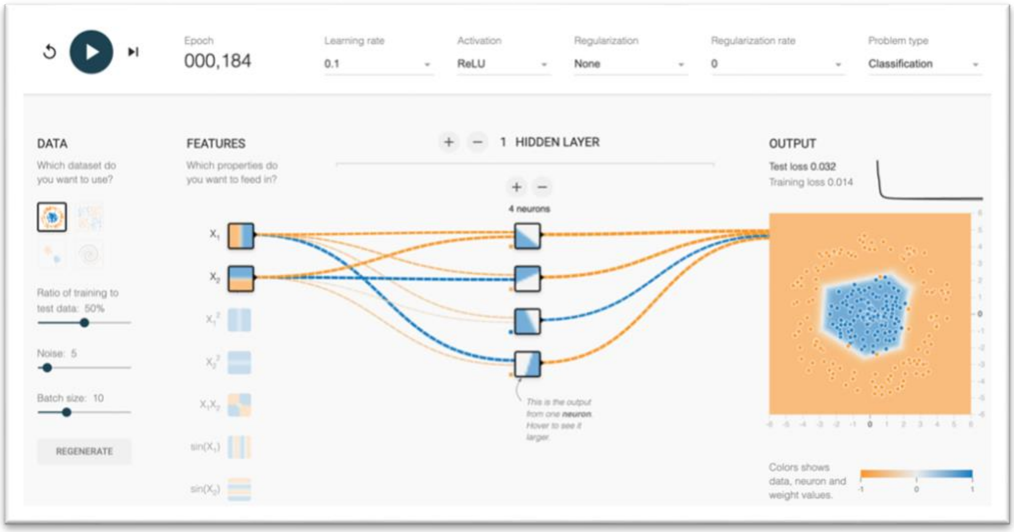
- Moderate noise improved generalization by preventing overfitting.
- Excessive noise led to poor model performance.

Task 5: Dataset Exploration

Experiment: Explore different datasets available in TensorFlow Playground and analyze network performance.

Figure 9

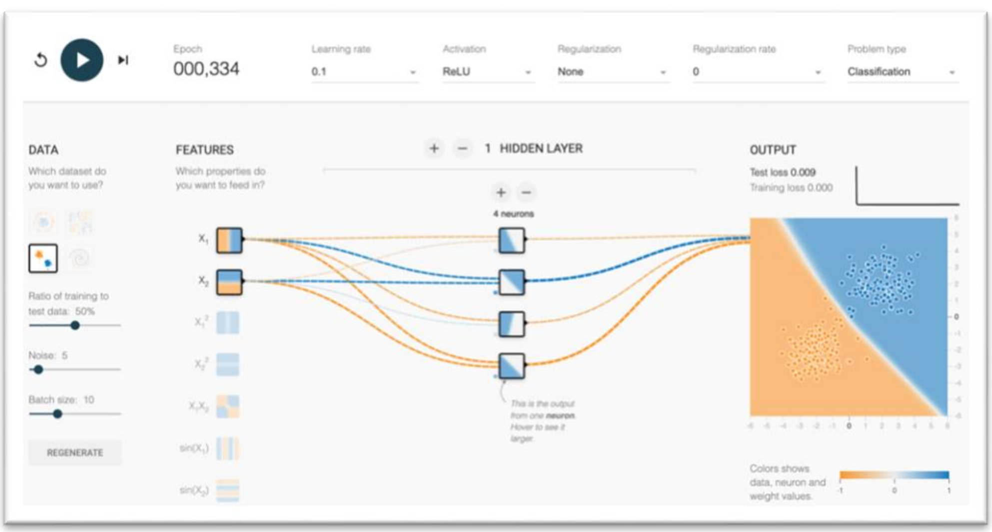
Neural Network with Circle Dataset



Note. Screenshot from TensorFlow Playground (https://playground.tensorflow.org). The network uses one hidden layer with 4 neurons and the Circle dataset.

Figure 10

Neural Network with XOR Dataset



Note. Screenshot from TensorFlow Playground (https://playground.tensorflow.org). The network uses one hidden layer with 4 neurons and the XOR dataset.

Explanation

- **Dataset Selection**: The choice of dataset affects the training and performance of neural networks. Different datasets can highlight various strengths and weaknesses of the model (NCBI, 2022).

Observations

- Performance varied significantly across datasets.
- Some configurations worked better for specific datasets, emphasizing the importance of dataset selection.

Report Summary

Introduction to Neural Networks

Neural networks are powerful tools for modeling complex patterns. Their key components include neurons, layers, weights, biases, and activation functions.

Task Descriptions and Observations

- **Activation Functions**: ReLU outperformed sigmoid due to faster convergence and better handling of gradients.
- **Hidden Layers**: More neurons/layers improved performance up to a point beyond which overfitting occurred.
- **Learning Rate**: Moderate rates balanced speed and accuracy.
- **Data Noise**: Moderate noise improved generalization, while excessive noise degraded performance.
- **Dataset Exploration**: Performance varied across datasets, highlighting the importance of dataset selection.

Practical Implications

Understanding these parameters is crucial for developing effective neural networks. Proper tuning can significantly enhance model performance in real-world applications.

Conclusion

This hands-on experience with TensorFlow Playground provided valuable insights into neural network behavior. Challenges included balancing complexity and generalization, which were addressed through careful parameter tuning.

References

1. Jeong, R., Barton, W., & Ramirez, M. (2020). Final Report: Deep Neural Networks. ICERM.
2. Fundamentals of Artificial Neural Networks and Deep Learning. (2022). NCBI.
3. Understanding neural networks with TensorFlow Playground. (2016). Google Cloud.
4. Neural networks memorise personal information from one sample. (2023). Nature.