Brytan M Kelly

TAILS - Training Artificial Intelligence using Linear and Square loss functions.

Forward

This code implements a neural network for regression using NumPy, with the Rectified Linear Unit (ReLU) activation function and the Mean Squared Error (MSE) loss function. It also includes functions for forward propagation, backward propagation, and training the network. The ReLU activation function is widely used in deep learning due to its simplicity and effectiveness. It has been shown to perform well on a wide range of tasks, including image classification, speech recognition, and natural language processing.

The derivative of the ReLU function is used in the backward propagation function to calculate the gradients for weight and bias updates. The MSE loss function is commonly used for regression tasks, where the goal is to predict a continuous output value. It measures the average squared difference between the predicted and true values. The loss is used to update the weights and biases during training.

The forward propagation function calculates the activations for each layer in the network given the input, weights, and biases. The backward propagation function calculates the gradients for each weight and bias using the chain rule of calculus. These gradients are used to update the weights and biases during training. The training function initializes the weights and biases randomly and iteratively updates them using the backpropagation algorithm. It also shuffles the training data and divides it into batches to improve the convergence of the algorithm. The model is saved to a file using the pickle module.

The use cases of this code in the field of medical AI are numerous. It can be used to predict the severity of a disease based on patient data, such as laboratory values, vital signs, and medical history. It can also be used to predict the efficacy of a drug or treatment based on clinical trials data. Additionally, it can be used for image analysis and diagnosis, such as detecting tumors in medical images or predicting the progression of a disease based on imaging data. The flexibility and scalability of neural networks make them a powerful tool for a wide range of medical applications.

How to use

The code has the following dependencies:

NumPy library for scientific computing with Python pickle module for serializing and de-serializing Python objects To save the trained model, the pickle module is used.

The trained model is saved in a file named "trained_model.pkl" using the "pickle.dump()" function. The trained model consists of two dictionaries - "weights" and "biases", which contain the learned parameters of the neural network.

To load the saved model, the "pickle.load()" function can be used to de-serialize the saved object. Here is an example of how to load the saved model:

with open("trained_model.pkl", "rb") as f: model = pickle.load(f) weights = model["weights"] biases = model["biases"]

This will load the saved model parameters into the "weights" and "biases" variables, respectively. These can then be used in the "forward_propagation()" function to make predictions on new data.

TAILS project Arctic.Framework Brytan M Kelly, 4/18/2023
brytankelly.com