EXPERIMENT-9_Report

From Scratch Code Snippets

Forward Propagation:

```
def _forward_propagation(self, X):
    A = X
    caches = []
    for i in range(len(self.layer_dims)-2):
        Z = np.dot(self.parameters[f"W{i+1}"], A) + self.parameters[f"b{i+1}"]
        A = self._activation(Z, 'relu')
        caches.append((Z, A))
          Z  =  np.dot(self.parameters[f"W{len(self.layer_dims)-1}"],  A)  +
self.parameters[f"b{len(self.layer_dims)-1}"]
    A = self._activation(Z, 'sigmoid')
    caches.append((Z, A))
    return A, caches
```

Backward Propagation:

```
def _backward_propagation(self, X, y, caches):
    grads = {}
    m = X.shape[1]
    A_final = caches[-1][1]
    dZ = A_final - y
    for i in reversed(range(len(self.layer_dims)-1)):
        A_prev = X if i == 0 else caches[i-1][1]
        grads[f"dW{i+1}"] = (1/m) * np.dot(dZ, A_prev.T)
        grads[f"db{i+1}"] = (1/m) * np.sum(dZ, axis=1, keepdims=True)
        if i > 0:
            dA_prev = np.dot(self.parameters[f"W{i+1}"].T, dZ)
            dZ = dA_prev * self._activation_derivative(caches[i-1][0], 'relu')
    return grads
```

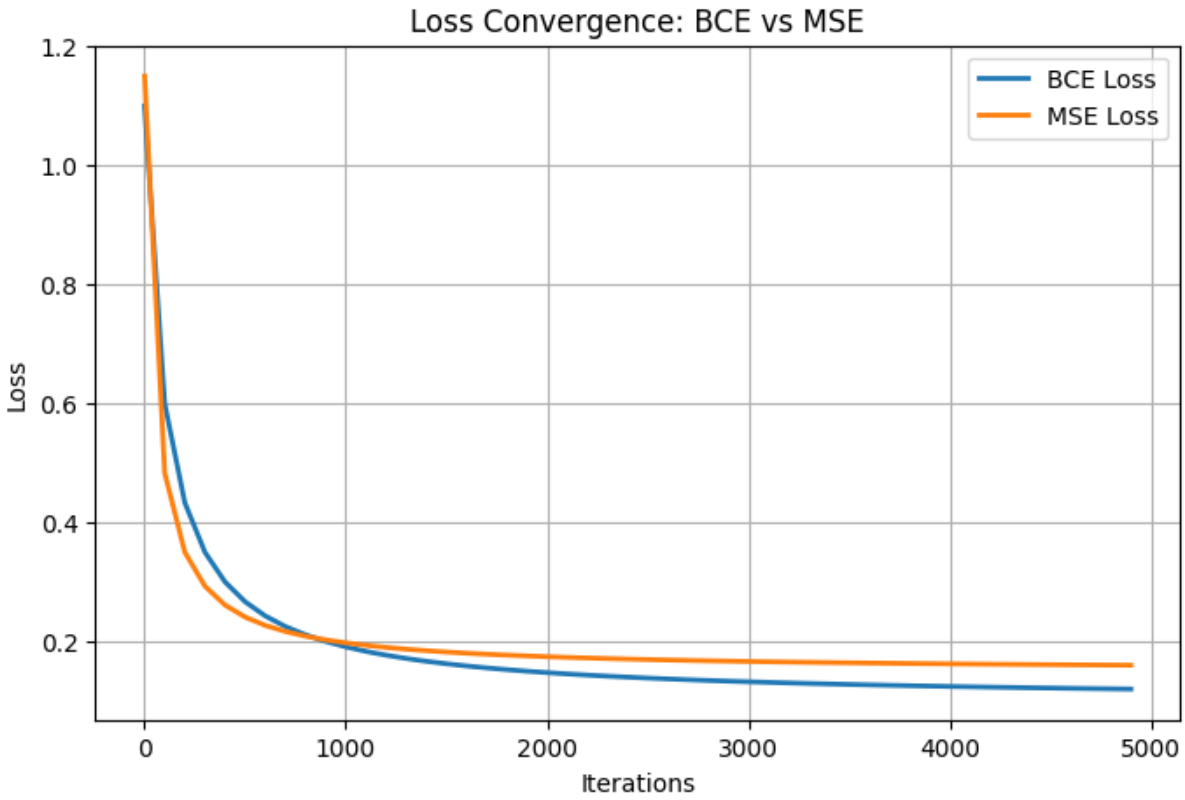
Update Parameters:

```
def _update_parameters(self, grads):
    for i in range(len(self.layer_dims)-1):
        self.parameters[f"W{i+1}"] -= self.learning_rate * grads[f"dW{i+1}"]
        self.parameters[f"b{i+1}"] -= self.learning_rate * grads[f"db{i+1}"]
```

Experiment Results (Precision, Recall, F1 for Class 1)

Model	Precision (class 1)	Recall (class 1)	F1-Score (class 1)
MyANN (BCE, 1 hidden layer)	0.63	1.00	0.77
MyANN (MSE, 1 hidden layer)	0.63	1.00	0.77
MyANN (BCE, 2 hidden layers)	0.63	1.00	0.77
sklearn.MLPClassifier	0.99	0.99	0.99

Loss Curve (BCE vs MSE)



Analysis and Conclusion

- BCE (Binary Cross Entropy) and MSE models both achieved identical results (F1 = 0.77), correctly predicting class 1 but failing to classify class 0. This suggests imbalance or poor generalization on the negative class.

- BCE is theoretically better suited for binary classification since it directly models probabilistic error, while MSE treats classification as regression.

- The deeper BCE model did not improve over the single-layer models, indicating that more layers did not add value without proper regularization or data balance.

- The sklearn MLPClassifier achieved a near-perfect F1-score (0.99) because it uses:
  - Batch-based training instead of full gradient descent
  - Better initialization and regularization defaults.

- The most challenging part of the "from scratch" implementation was ensuring correct matrix dimensions, debugging gradient flow, and stabilizing convergence across multiple layers.