CS303 · Artificial Intelligence Fall 2024

Project 2

THA SREYNY / 12113053

December 5, 2024

1 Subtask 1

1.1 Introduction

In this subtask, the objective is to independently train a model using the provided training set, which will then be utilized to predict labels for the test set. The predicted labels should be saved in a file named `classification_results.pkl`. The baseline model for this task is Softmax Regression.

1.2 Methodology

Algorithm/Model Design Baseline Model: Softmax Regression

The Softmax Regression model is designed as follows:

- **Constructor:**
  - `num_classes`: The number of classes in the classification task.
  - `learning_rate`: The learning rate used for gradient descent (default value = 0.01).
  - `num_iterations`: The total number of iterations for training (default value = 100).
  - `random_seed`: The seed used for random number generation, ensuring reproducibility (default value = None).
- **fit Method:**
  - `X_train`: The feature data used for training the model.
  - `y_train`: The true labels corresponding to the training data.
  - `X_val`: Optional validation feature data.
  - `y_val`: Optional validation labels.
  - `decay_rate`: The rate at which the learning rate decays (default value = 0.9995).

  In the `fit` method, the following steps are carried out:
  - The weights are initialized randomly.
  - Lists are created to store the training and validation losses and accuracies.
  - A loop runs for the specified number of iterations:
    - Logits and softmax probabilities are computed.
    - The cross-entropy loss is calculated.
    - Gradients are computed, and weights are updated using gradient descent.
    - The learning rate is decayed after a certain number of iterations.
    - Training and validation accuracies are computed and stored.
  - The method returns the lists of training and validation losses and accuracies.
- **predict Method:**
  - `X`: The feature data used for making predictions.

In the `predict` method, the logits are calculated, and the predicted class labels are returned.

1.3 Improved Model: Softmax Regression with Increased Learning Rate

In this improved model, the learning rate has been increased from the default value of 0.01 to 0.105. This change was made to accelerate the convergence of the model during training by allowing larger updates to the weights in each iteration of gradient descent. A higher learning rate may help the model converge more quickly, potentially reducing training time. However, if set too high, it might lead to instability or overshooting the optimal minimum, so the learning rate was carefully selected based on experimentation.

The updated Softmax Regression model is instantiated as follows:

```
linear_model = SoftmaxRegression(
    num_classes=num_classes,
    learning_rate=0.105,
    num_iterations=10000,
    random_seed=seed)
```

In this setup:

- `learning_rate`: The learning rate has been increased to 0.105 to accelerate convergence.
- `num_iterations`: The number of iterations has been set to 10,000 to allow enough time for the model to converge with the increased learning rate.
- `random_seed`: The random seed ensures reproducibility of the results.

This modification is intended to improve training efficiency while maintaining the stability of the model. Further adjustments or decay strategies may be considered if the higher learning rate leads to instability.

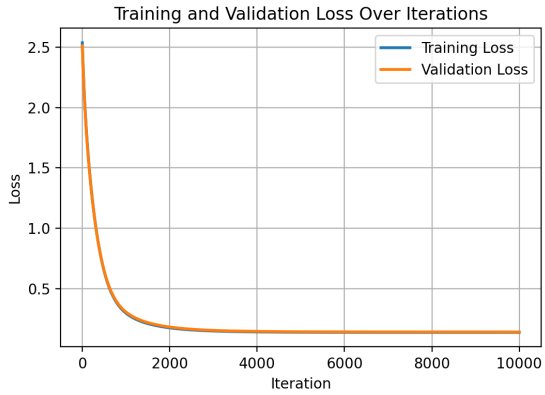

Figure 1: Training and Testing Loss

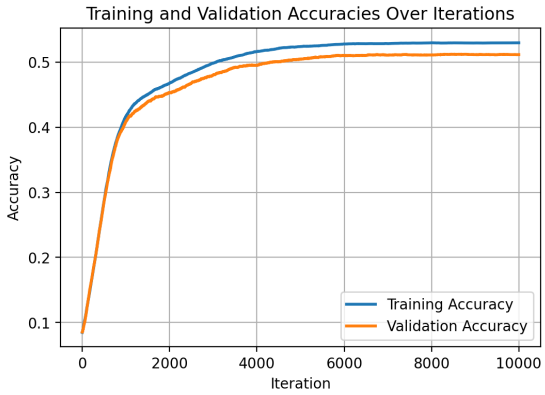

Figure 2: Training and Testing Accuracy

2 Subtask 2

2.1 Introduction

This task is to implement a nearest neighbor search (NNS) model to find the most similar images in the image repository for each image in the test set. This model should then be utilized to respond to image queries from the test set. For each image in the test set, should find 5 similar images in the image repository. The baseline model is KNN.

2.2 2. Methodology

Algorithm/Model Design

- **Baseline:** K-Nearest Neighbors (KNN) using Euclidean distance.
- **Improved Model:** K-Nearest Neighbors (KNN) using Manhattan distance.

Analysis

Manhattan distance is generally more robust against outliers compared to Euclidean distance. This is because it calculates the distance along axis-aligned directions, which reduces the influence of extreme values, making it better suited for datasets that contain noise or outliers.

Furthermore, in cases where certain features are of equal importance in determining similarity, Manhattan distance can be more effective. It gives each feature's contribution a linear weighting, ensuring all features are treated equally. This makes it a good choice when the importance of individual features is uniform.

2.3 Experiments

Metrics

For each image in the test set, 5 similar images are selected (i.e., $n = 5$), and the number of true positive similar images is determined based on the evaluation process. The accuracy is then calculated as $\frac{m}{n} \times 100\%$, where m represents the number of true positives. The overall test accuracy is the average accuracy across the entire test set.

Experimental Results

The result is: 0.05.

2.4 Further Considerations

Other distance metrics that could be explored for capturing image similarities include:

- **Cosine Similarity:** This metric measures the cosine of the angle between two vectors, disregarding their magnitudes. It is particularly effective when the focus is on the orientation of the features rather than their magnitudes, making it suitable for text-based or high-dimensional data representations.

3 Subtask 3

3.1 Introduction

The task is to develop an algorithm that selects up to 30 features from the image feature set in the classification validation set. The objective is to generate a mask that retains only the most important features.

3.2 Methodology

Algorithm/Model Design

- **Baseline:** Random selection of features using a seed value of 42.
- **Improved Model 1:** Feature selection with a seed value of 2024.

Analysis

The RandomForestClassifier approach aims to systematically select the most significant features from the validation set. In contrast to the baseline method (random selection with a fixed seed), this technique ensures that feature selection is guided by the contribution of each feature to improving the model's predictive performance.

3.3 Experiments

Metrics

Classification accuracy is measured using the selected features.

3.3.1 Experimental Results

- **Baseline:** 0.150
- **Improved Model 1:** 0.243

Further Considerations

Other feature selection methods to consider include heuristic-based functions that iteratively select features, similar to the approach used in Subtask 2 of Project 1.