Enabling People
Education for Future Generations

AI Course

Chapter 9. Quiz

For students

Samsung Innovation Campus

1. The MNIST dataset consists of numeric pictures written cursively from 0 to 9. If you want to create a neural network that classifies the data, what is the activation function of the last fully connected (=Dense) layer?

Answer: Softmax (for multi-class classification of 10 digit classes 0-9)

2. Blood pressure, height, and weight form the feature vectors. The training set is given as follows.

$$\begin{pmatrix}121\\1.72\\69.0\end{pmatrix},\begin{pmatrix}140\\1.62\\63.2\end{pmatrix},\begin{pmatrix}120\\1.70\\59.0\end{pmatrix},\begin{pmatrix}131\\1.82\\82.0\end{pmatrix},\begin{pmatrix}101\\1.78\\73.5\end{pmatrix}$$

(1) Assuming that the weight vector of the perceptron is $(-0.01,0.5.-0.23)^T$ and the bias is 0, explain the scale problem with the training set.

Answer: The scale problem occurs because the features (blood pressure, height, weight) have vastly different ranges. For example, weight might be 50-100 kg, height 150-200 cm, and blood pressure 80-140 mmHg. Features with larger magnitudes will dominate the perceptron's learning process, causing the model to be biased toward those features and making convergence difficult.

(2) Write the training set that results after the application of the preprocessing formula below.

Answer: After applying the preprocessing formula (5.9) which typically is standardization: x' = (x - mean) / std, each feature will be normalized to have mean 0 and standard deviation 1. The exact values depend on the original training set data, but all features will be on the same scale (typically in the range of approximately -3 to +3).

$$x_i^{new} = \frac{x_i^{old} - \mu_i}{\sigma_i} \quad (5.9)$$

(3) Explain your observation whether this data preprocessing alleviates the scale problem.

Answer: Yes, this data preprocessing alleviates the scale problem. By standardizing all features to have the same mean (0) and standard deviation (1), all features are now on the same scale. This prevents features with larger magnitudes from dominating the learning process and allows the perceptron to learn weights fairly across all features, leading to faster and more stable convergence.

3. Neural networks, convolution layers are repeated multiple times in deep learning, and it causes some nodes get omitted to a great degree. What technique can you use to prevent this problem?

Answer: Dropout - This technique randomly omits (drops) a proportion of neurons during training to prevent overfitting and co-adaptation. Other techniques include: Batch Normalization, Skip Connections/Residual Connections, and proper weight initialization.

4. Weight initialization should generate random numbers in the range [-1,1]. Please provide the Python code that does this function.

Answer:

```
import numpy as np
# For a weight matrix of shape (n_inputs, n_outputs)
weights = np.random.uniform(-1, 1, size=(n_inputs, n_outputs))
# Or using random.rand:
weights = 2 * np.random.rand(n_inputs, n_outputs) - 1
```

5. You want to train a classifier when you have many unlabeled training data but only a few thousand labeled data. Describe how the autoencoder can be helpful and how to work.

Answer: The autoencoder helps through unsupervised pre-training:

1. Train an autoencoder on all unlabeled data to learn useful feature representations

2. The encoder part learns to extract meaningful features from the input

3. Remove the decoder and use the pre-trained encoder as feature extractor

4. Add a classification layer on top of the encoder

5. Fine-tune the entire network using the small labeled dataset

This approach leverages the large unlabeled dataset to learn good representations, which improves classification performance with limited labeled data.

6. Train the 'Fashion-mnist' dataset classification model with reference to the following code.

Answer: To complete the model:

```
model.add(tf.keras.layers.Conv2D(32, (3,3), activation='relu'))
model.add(tf.keras.layers.MaxPooling2D((2,2)))
model.add(tf.keras.layers.Conv2D(64, (3,3), activation='relu'))
model.add(tf.keras.layers.MaxPooling2D((2,2)))
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(128, activation='relu'))
model.add(tf.keras.layers.Dense(10, activation='softmax'))
```

model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=learn_rate), loss='sparse_categorical_crossentropy', metrics=['accuracy'])

model.fit(X_train, y_train, batch_size=batch_size, epochs=n_epochs, validation_data=(X_test, y_test))

```
import tensorflow as tf
import numpy as np
import pandas as pd
from tensorflow.keras.datasets.fashion_mnist import load_data

# load data
(X_train, y_train), (X_test, y_test) = load_data()

# data processing
X_train = X_train.reshape((-1, 28, 28, 1))
X_train = X_train/255
X_test = X_test.reshape((-1, 28, 28, 1))
X_test = X_test/255

# training parameters
batch_size = 8
n_epochs = 20
learn_rate = 0.0001

# model
model = tf.keras.Sequential()
model.add(tf.keras.Input(shape=(28, 28, 1)))

# please complete cnn based model and train
```