Sure, I can use the two files you provided to answer your questions. Here are the answers:

3. Illustrate Stochastic gradient descent for Optimization of neural networks with a neat diagram.

Stochastic Gradient Descent (SGD) is a popular optimization algorithm used in training deep neural networks. It is an iterative optimization algorithm that minimizes the loss function by iteratively updating the parameters of the network based on the gradients of the loss function with respect to the parameters. Here are the steps involved in SGD optimization:

1. Initialize the weights: The weights of the neural network are randomly initialized.

2. Forward pass: Input data is fed to the network, and the forward pass is performed to obtain the output.

3. Compute the loss: The loss is computed by comparing the network's output with the ground truth labels.

4. Compute the gradients: The gradients of the loss function with respect to the parameters are computed using backpropagation.

5. Update the weights: The weights are updated using the computed gradients and the learning rate. The learning rate determines the step size for the weight updates.

6. Repeat steps 2-5 for multiple iterations until convergence.

Here is a neat diagram illustrating the SGD optimization process:

![Stochastic Gradient Descent Diagram](https://i.imgur.com/5dXG1fW.png)

4. Explain in detail on Forward and Backward Propagation in Artificial Neural Network with a neat diagram.

Forward propagation is the process of computing the output of a neural network for a given input. It involves computing the weighted sum of the inputs and the biases at each neuron, and passing the result through an activation function to obtain the output of each neuron. Here are the steps involved in forward propagation:

1. Input layer: The input data is fed into the network, and the values are propagated to the first hidden layer.

2. Hidden layers: The values from the previous layer are multiplied by the weights and added to the bias to compute the weighted sum. Then, the result is passed through the activation function to obtain the output of each neuron.

3. Output layer: The output from the final hidden layer is multiplied by the weights and added to the bias to compute the final output.

Here is a neat diagram illustrating forward propagation in a neural network:

![Forward Propagation Diagram](https://i.imgur.com/5VxyDpM.png)

Backward propagation, also known as backpropagation, is the process of computing the gradients of the loss function with respect to the weights and biases of the neural network. It is used in training the network using optimization algorithms such as SGD. Here are the steps involved in backward propagation:

1. Compute the error: The difference between the network's output and the ground truth labels is computed.

2. Backpropagate the error: The error is propagated backwards through the network, and the gradients of the loss function with respect to the weights and biases are computed using the chain rule.

3. Update the weights: The weights are updated using the computed gradients and the learning rate.

Here is a neat diagram illustrating backward propagation in a neural network:

![Backward Propagation Diagram](https://i.imgur.com/4Izk0d5.png)

I hope this helps! Let me know if you have any further questions or need additional explanations.

Sure, I can use the two files you shared to provide a brief long answer to your questions. Here are the answers to your questions:

1. Principal Component Analysis (PCA) is a dimensionality reduction technique used to reduce the number of variables in a data set while retaining as much of the original information as possible. It works by finding the direction in which the data varies the most and projecting the data onto that direction. The direction with the most variation is called the first principal component, and subsequent principal components are found by finding orthogonal directions that explain as much remaining variation as possible. The working methodology of PCA can be summarized as follows:

- Normalize the data by subtracting the mean and dividing by the standard deviation.

- Compute the covariance matrix of the data.

- Find the eigenvectors and eigenvalues of the covariance matrix.

- Sort the eigenvectors in descending order of their corresponding eigenvalues.

- Select the top k eigenvectors with the largest eigenvalues, where k is the desired number of principal components.

- Project the original data onto the k eigenvectors to obtain the reduced-dimensional representation.

A suitable example of PCA is in image compression, where high-resolution images are often very large and computationally expensive to process. By using PCA to reduce the dimensionality of the images, we can compress them without losing too much information. For example, we can use PCA to transform a high-resolution image into a lower-dimensional space and then reconstruct the original image from the lower-dimensional representation with minimal loss of information.

2. Singular Value Decomposition (SVD) is a matrix factorization technique that can be used to decompose a matrix into its constituent parts. In the context of handwritten digit recognition using sci-kit-learn, SVD can be used to preprocess the input data to improve the performance of the classification algorithm. The working methodology of SVD applied to handwritten digits using sci-kit-learn can be summarized as follows:

- Load the MNIST dataset, which contains 70,000 images of handwritten digits.

- Preprocess the data by applying SVD to reduce the dimensionality of the data.

- Train a classification algorithm using the preprocessed data.

- Evaluate the performance of the classification algorithm on a test set of handwritten digits.

Some possible points to consider when describing the application of SVD to handwritten digit recognition using sci-kit-learn include:

- SVD can be used to reduce the dimensionality of the input data, which can improve the performance of the classification algorithm.

- The MNIST dataset is a common benchmark dataset for handwritten digit recognition, and has been used extensively in machine learning research.

- The preprocessed data can be visualized to see how much of the original variation is retained in the reduced-dimensional representation.

- Different values of k can be used to control the degree of dimensionality reduction and the amount of information retained in the reduced-dimensional representation.

- Other preprocessing techniques, such as data normalization, can also be used to improve the performance of the classification algorithm.

Here is a link to an image showing the MNIST dataset: https://upload.wikimedia.org/wikipedia/commons/2/27/MnistExamples.png