EXPLORING DIMENSIONALITY REDUCTION TECHNIQUES ON MNIST HANDWRITTEN DIGIT DATASET

RHICHARD KOH

AGENDA

Objective

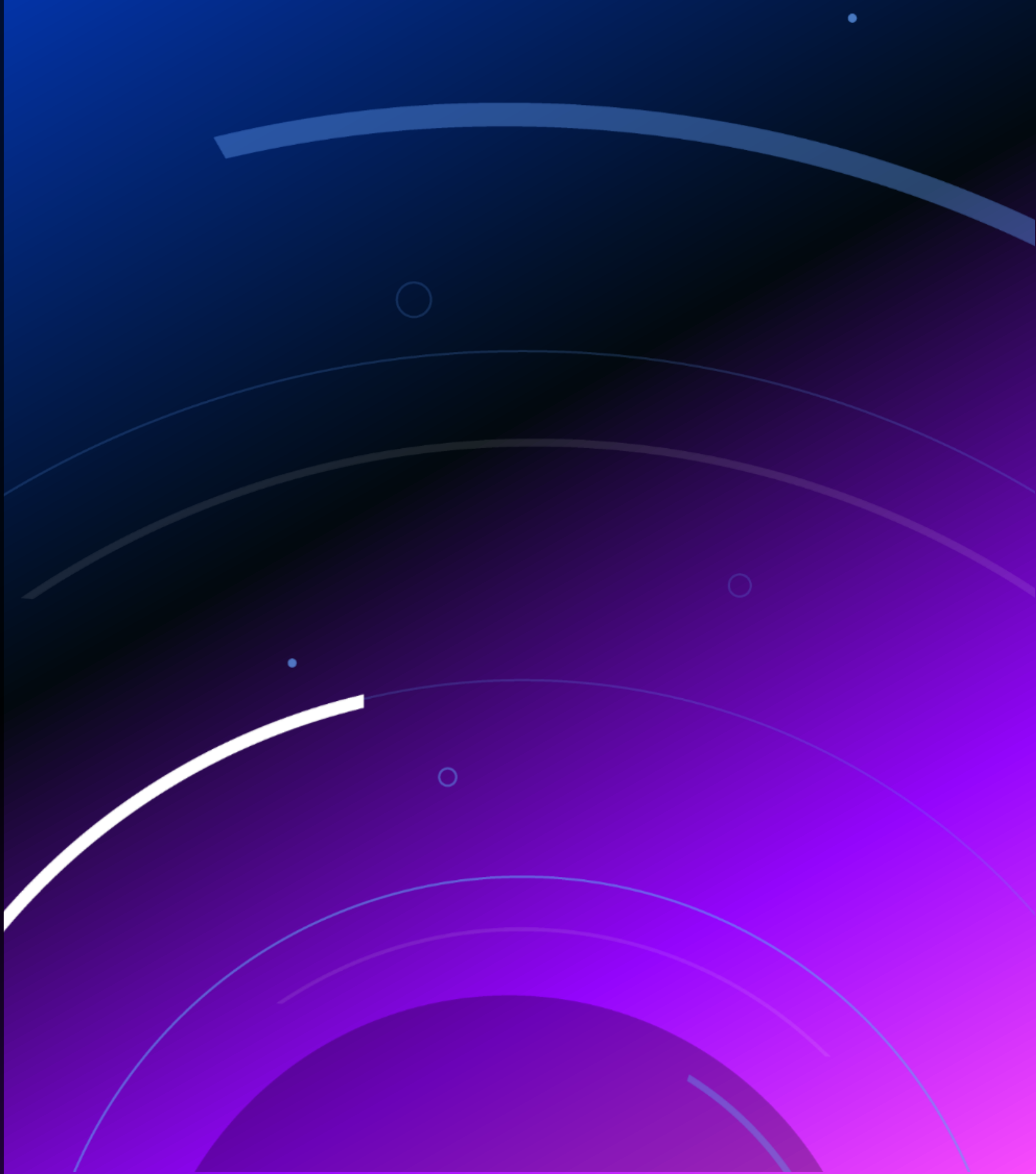
Algorithms Used

Methodology

Results

Analysis

Conclusion



OBJECTIVE

OBJECTIVE

The plan is to thoroughly analyze and compare various popular methods for reducing dimensions and extracting features, such as Principal Component Analysis, Autoencoders, Restricted Boltzmann Machines, and Deep Belief Networks to understand their strengths and weaknesses better. I will use each algorithm on the MNIST dataset, which contains 70,000 small grayscale images of handwritten digits from zero to nine. This dataset is divided into 60,000 images for training and 10,000 images for testing to see how well the models perform on new data. Each image has a resolution of 28 by 28 pixels.

ALGORITHMS USED

ALGORITHMS USED

- Principal Component Analysis (PCA)
- Autoencoders
- Restricted Boltzmann Machines (RBMs)
- Deep Belief Networks (DBNs)

PRINCIPAL COMPONENT ANALYSIS

- PCA simplifies complex data by transforming related variables into new, uncorrelated variables known as principal components.
- This transformation reduces irrelevant information (noise) and helps reveal hidden patterns in the data.

AUTOENCODERS

- Type of neural network: Used to encode unlabeled data.
- Dimension reduction: Compresses data into a smaller dimension while retaining the essence of the original data.
- Goal: Compress and then reconstruct the data as accurately as possible.
- Learning process: Helps the network learn which features of the data are most important.

RESTRICTED BOLTZMANN MACHINES

- Type of neural network: Learns probabilistic patterns of input data in an unsupervised manner.
- Purpose: Excels at recognizing underlying patterns and features in data.
- Applications: Useful for tasks such as reducing data dimensions and feature extraction.

DEEP BELIEF NETWORKS

- Structure: Consist of multiple layers of Restricted Boltzmann Machines (RBMs).
- Sequential training: Each layer is trained on the output of the previous layer.
- Function: Models complex data hierarchies effectively.
- Applications: Powerful for multi-level data structure tasks, such as feature recognition in images or dimension reduction while retaining crucial information.

METHODOLOGY

DATA PREPARATION

- Image transformation: Converted each 28x28 pixel image into a 784-dimensional vector by flattening.
- Standardization: Standardized the pixel values by dividing each by 255.
- Pixel value range: After standardization, pixel values ranged between 0 and 1.

CALCULATION OF METRICS

- Reconstruction Loss:
  - Measured using the mean squared error (MSE) between the original flattened images and the reconstructed images from the PCA-transformed space.
- Classifier Training:
  - Trained a logistic regression classifier on the PCA-reduced feature space to evaluate how well the reduced data could classify images.
  - Tested the classifier’s accuracy using a random selection of 1000 images from the test dataset.

PCA CONFIGURATION

- Configurations Tested: Used 50, 100, 150, and 200 principal components.
- Metrics Evaluated:
    - Reconstruction Loss: Assessed how well the original images could be rebuilt using different numbers of components.
    - Classifier Accuracy: Measured how accurately a classifier could identify images when trained on data reduced to different numbers of components.

AUTOENCODER CONFIGURATION

- Network Components:
  - Encoder: Compresses input images into a smaller, specified encoding dimension.
  - Decoder: Reconstructs original images from the compressed encoded representations.
- Experiments Conducted:
  - Used encoding dimensions of 50, 100, 150, and 200.
  - Aimed to analyze effects on reconstruction loss and classification accuracy.

AUTOENCODER CONFIGURATION

- Training Details:
  - Trained on the MNIST dataset for 8 epochs.
  - Used Mean Squared Error (MSE) as the loss function.
  - Employed the Adam optimizer with a learning rate of 0.001.
- Architecture:
  - Defined within the Autoencoder class, separating the encoder and decoder components.
  - Forward method efficiently handles data encoding and decoding during the forward pass.

RBN CONFIGURATION

- Key Functions:
  - sample_h: Samples the hidden layer.
  - sample_v: Samples the visible layer.
  - Calculates the free energy for training.
  - Performs a forward pass to generate the reconstructed visible layer, critical for assessing reconstruction loss.

RBN CONFIGURATION

- Initial Setup:
  - Weights (W) are initialized close to zero to prevent large activations early in training.
  - Biases for both visible and hidden layers are initially set to zero, starting the learning process without pre-existing bias.

RBN CONFIGURATION

- Training Details:
  - Trained on the MNIST dataset for 8 epochs.
  - Used Mean Squared Error (MSE) as the loss function.
  - Employed the Adam optimizer with a learning rate of 0.001.
- Architecture:
  - Defined within the Autoencoder class, separating the encoder and decoder components.
  - Forward method efficiently handles data encoding and decoding during the forward pass.

RBN CONFIGURATION

- Post-Training Evaluation:
    - Extracts hidden representations to train a SoftMax classifier.
    - Assesses the effectiveness of the RBM’s feature extraction for classification tasks.
    - Evaluates how well the RBM differentiates between classes based on the learned features.

DBN CONFIGURATION

- Network Structure:
  - Sequence of layers including dense (fully connected) and dropout layers.
  - Dense layers for data processing and dropout layers to manage overfitting.
- Data Preprocessing for MNIST Dataset:
  - Images reshaped into 784-dimensional vectors (from 28x28 pixels).
  - Pixel values normalized to a [0, 1] scale.
  - Labels encoded into one-hot vectors.

DBN CONFIGURATION

- Model Configuration:
  - Dynamically built with tunable hyperparameters: units in each dense layer and dropout rate.
  - Utilized a RandomSearch tuner from Keras Tuner to explore different configurations.
  - Units tested at [50, 100, 150, 200].
  - Dropout rate tested from 0.0 to 0.5.

DBN CONFIGURATION

- Activation and Compilation:
  - ReLU activation in dense layers to introduce non-linearity and learn complex patterns.
  - Softmax activation in the final layer for multi-class probability output.
  - Compiled with the Adam optimizer and categorical crossentropy loss.
- Training and Validation:
  - Trained on the training data with separate validation set to monitor performance and prevent overtraining.
  - RandomSearch tuner ran for a maximum of 10 trials to optimize units and dropout rate based on validation accuracy.

RESULTS

PCA RESULTS

- Reconstruction loss for PCA with 50 components: 0.011795084610364161 Classification Accuracy: 0.924
- Reconstruction loss for PCA with 100 components: 0.005756633405031867 Classification Accuracy: 0.93
- Reconstruction loss for PCA with 150 components: 0.0034891502018314425 Classification Accuracy: 0.924
- Reconstruction loss for PCA with 200 components: 0.0022767954728689765 Classification Accuracy: 0.926

PCA RESULTS ANALYSIS

- Reconstruction Loss:
  - Noted a consistent decrease in reconstruction loss as the number of principal components increased, indicating improved ability to reconstruct the original data.
- Classification Accuracy:
  - Optimal minimal classification accuracy achieved with 100 principal components.
  - Utilizing more than 100 components does not significantly enhance accuracy and may result in resource wastage.
- Trade-offs in Dimensionality Reduction:
  - Adding more components generally improves data reconstruction.
  - However, beyond a certain threshold, the benefits in classification accuracy diminish, and additional components may introduce unnecessary complexity with minimal gain.

AUTOENCODER RESULTS

- 50: Reconstruction Loss: 0.9358660034152236, Classification Accuracy: 0.8038
- 100: Reconstruction Loss: 0.9164198400623509, Classification Accuracy: 0.8621
- 150: Reconstruction Loss: 0.9233703469035468, Classification Accuracy: 0.8531
- 200: Reconstruction Loss: 0.9157706576623896, Classification Accuracy: 0.8583

AUTOENCODER RESULTS ANALYSIS

- Impact of Encoding Dimension:
  - The encoding dimension significantly affects both reconstruction loss and classification accuracy.
  - Higher encoding dimensions generally increase classification accuracy as more detailed features of the images are captured.
- Trade-offs and Diminishing Returns:
  - Beyond a certain encoding size, gains in accuracy diminish, accompanied by higher reconstruction loss and efficiency loss due to potential overfitting.

AUTOENCODER RESULTS ANALYSIS

- Optimal Encoding Dimension:
  - An encoding dimension of 100 offers an effective balance, achieving a classification accuracy of 86%.
  - This setting allows the autoencoder to retain sufficient image detail for accurate classification while efficiently compressing data.
  - Suggests that a 100-dimensional encoding effectively maintains essential information necessary for tasks like image classification without excessive data loss.

RBN RESULTS

- 50: Reconstruction Loss: -19.923210117608498, Classification Accuracy: 0.8439
- 100: Reconstruction Loss: -17.291857076860442, Classification Accuracy: 0.8758
- 150: Reconstruction Loss: -15.531131760652132, Classification Accuracy: 0.8845
- 200: Reconstruction Loss: -14.382853977715792, Classification Accuracy: 0.8889

RBN RESULTS ANALYSIS

- Reconstruction Loss:
  - Generally, increasing the number of hidden units in an RBM improves its ability to model complex data patterns, which reduces reconstruction loss.
  - An observed increase in reconstruction loss with 200 hidden units suggests potential overfitting or insufficient training epochs for larger models.
- Classification Accuracy:
  - Classification accuracy improves as the number of hidden units increases, peaking at 200 hidden units.
  - The increase in accuracy from 150 to 200 hidden units is marginal, indicating diminishing returns.
  - Further increasing the number of hidden units beyond 150 shows little benefit in this context, suggesting a threshold has been reached where added complexity does not significantly enhance performance.

DBN RESULTS

- Top 1 Trial: Hyperparameters: {'units': 50, 'dropout_rate': 0.4} Objective (e.g., Loss): 0.9639000296592712 Best Validation Loss: 0.12775953114032745 Best Validation Accuracy: 0.9639000296592712

- Top 2 Trial: Hyperparameters: {'units': 50, 'dropout_rate': 0.30000000000000004} Objective (e.g., Loss): 0.9678999781608582 Best Validation Loss: 0.10606943070888519 Best Validation Accuracy: 0.9678999781608582

- Top 3 Trial: Hyperparameters: {'units': 50, 'dropout_rate': 0.2} Objective (e.g., Loss): 0.97079998254776 Best Validation Loss: 0.09899251908063889 Best Validation Accuracy: 0.97079998254776

- Top 4 Trial: Hyperparameters: {'units': 100, 'dropout_rate': 0.4} Objective (e.g., Loss): 0.9760000109672546 Best Validation Loss: 0.08561421930789948 Best Validation Accuracy: 0.9760000109672546

- Top 5 Trial: Hyperparameters: {'units': 100, 'dropout_rate': 0.30000000000000004} Objective (e.g., Loss): 0.9763000011444092 Best Validation Loss: 0.08270733803510666 Best Validation Accuracy: 0.9763000011444092

DBN RESULTS

- Top 6 Trial: Hyperparameters: {'units': 100, 'dropout_rate': 0.2} Objective (e.g., Loss): 0.9778000116348267 Best Validation Loss: 0.07927803695201874 Best Validation Accuracy: 0.9778000116348267
- Top 7 Trial: Hyperparameters: {'units': 150, 'dropout_rate': 0.4} Objective (e.g., Loss): 0.9781000018119812 Best Validation Loss: 0.07377926260232925 Best Validation Accuracy: 0.9781000018119812
- Top 8 Trial: Hyperparameters: {'units': 200, 'dropout_rate': 0.4} Objective (e.g., Loss): 0.9786999821662903 Best Validation Loss: 0.07289712131023407 Best Validation Accuracy: 0.9786999821662903
- Top 9 Trial: Hyperparameters: {'units': 150, 'dropout_rate': 0.30000000000000004} Objective (e.g., Loss): 0.9787999987602234 Best Validation Loss: 0.07296411693096161 Best Validation Accuracy: 0.9787999987602234
- Top 10 Trial: Hyperparameters: {'units': 200, 'dropout_rate': 0.30000000000000004} Objective (e.g., Loss): 0.9815000295639038 Best Validation Loss: 0.0684111937880516 Best Validation Accuracy: 0.9815000295639038

DBN RESULTS ANALYSIS

- Reconstruction Loss:
  - Increasing the number of hidden units generally reduces validation loss, suggesting better model capacity to capture complex data patterns.
  - The optimal validation loss was observed with 200 hidden units and a dropout rate of 0.3, with a specific loss value of 0.0684111937880516.
  - The minimal differences in loss between configurations with 150 and 200 units indicate diminishing returns, potentially signaling overfitting or the network reaching its complexity limit for the task.

DBN RESULTS ANALYSIS

- Classification Accuracy:
  - Classification accuracy increases with the number of hidden units, peaking at 200 hidden units with an accuracy of 98.15%.
  - The incremental accuracy improvement from 150 to 200 units is slight, underscoring diminishing returns and suggesting that further increases may not yield significant benefits.
  - This pattern suggests that while the network gains from additional hidden units up to a point, expanding beyond 200 units might not offer substantial advantages and could lead to overfitting, impacting performance on unseen data.

CONCLUSION

CONCLUSION

- In this comprehensive evaluation, we analyzed various dimensionality reduction and feature extraction techniques on the MNIST dataset to assess their effectiveness regarding reconstruction loss and classification accuracy. The techniques included Principal Component Analysis (PCA), Autoencoders, Restricted Boltzmann Machines (RBMs), and Deep Belief Networks (DBNs).

KEY FINDINGS

- PCA proved effective with 100 principal components, striking a balance between data compression and preserving essential information, achieving a high classification accuracy of 93%.
- Autoencoders performed best with a 100-dimensional encoding, capturing a broad spectrum of features with an 86% classification accuracy.
- RBMs excelled with 150 hidden units, optimizing reconstruction quality and interpretability of digit images.
- DBNs, set with 200 hidden units, captured the most detailed features, resulting in the highest classification accuracy of 98.15% among the models.

IMPLICATIONS

- The choice of dimensionality reduction and feature extraction technique should align with the specific requirements of the task, considering factors like accuracy demands and computational resources.
- PCA is suitable for scenarios requiring high accuracy with moderate computational effort.
- Autoencoders and RBMs are ideal for applications that need detailed feature extraction.
- DBNs are best for tasks that benefit from a deep, nuanced understanding of the data, given their complex architecture.

CONCLUSION

- This analysis not only highlights the strengths and weaknesses of each method but also underscores that no single approach is universally applicable. Instead, the selection should be tailored to the particular characteristics of the data and the application's performance needs. This study serves as a foundational reference to help practitioners choose the most appropriate technique for their data-intensive tasks, aiming to enhance the efficiency and effectiveness of their models.

THANK YOU

Rhichard Koh