KUGANESH S
721221104031
Computer Science and Engineering
Karpagam Institute of Technology

PROJECT TITLE

anomaly detection with autoencoder

1.AGENDA

Introduction: Define anomalies and autoencoders.

Autoencoder Basics: Explain autoencoder architecture.

Anomaly Detection Method: Discuss using autoencoders for anomaly detection.

Training and Preprocessing: Cover data preprocessing and autoencoder training.

Evaluation: Explore evaluation metrics and case studies.

Challenges and Future: Address limitations and future directions

PROBLEM STATEMENT

"Increasingly complex datasets across various domains necessitate robust anomaly detection methods. Leveraging autoencoder neural networks, this project aims to develop a system capable of identifying anomalies within high-dimensional data, thereby enhancing data-driven decision-making processes and minimizing risks associated with anomalies."

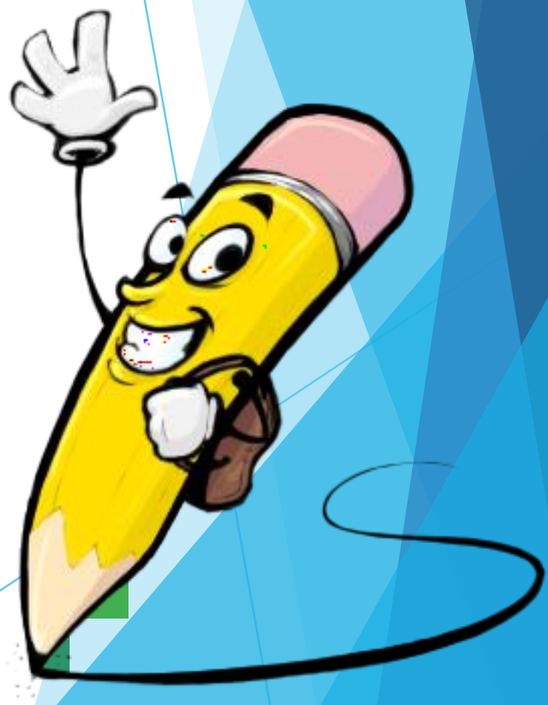
PROJECT OVERVIEW

In this project, we aim to develop an anomaly detection system using autoencoder neural networks

We'll preprocess the data, train the autoencoder model, and detect anomalies based on reconstruction errors.

Evaluation will involve comparing the system's performance with traditional methods.

Ultimately, our project aims to enhance anomaly detection accuracy and efficiency across various domains."



WHO ARE THE END USERS?

The end users for anomaly detection with autoencoders include cybersecurity analysts,
fraud detection teams, manufacturing quality control engineers,
healthcare professionals, predictive maintenance teams, and supply chain managers. These stakeholders rely on anomaly detection to identify
irregularities in data streams, ranging from network traffic to manufacturing processes, enabling proactive responses and mitigating risks in their respective domains.

YOUR SOLUTION AND ITS VALUE PROPOSITION

Our solution leverages autoencoder neural networks for anomaly detection, offering a robust and efficient method to identify irregularities in high-dimensional data. By training the autoencoder on normal data, it learns to reconstruct typical patterns, enabling it to detect deviations indicative of anomalies. The value proposition lies in its ability to autonomously identify anomalies without requiring labeled data, making it suitable for various domains where labeled anomaly data is scarce.

```
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Dense

(X_train, _), (X_test, _) = mnist.load_data()
X_train = X_train.astype('float32') / 255.0
X_test = X_test.astype('float32') / 255.0
X_train = np.reshape(X_train, (len(X_train), 28*28))
X_test = np.reshape(X_test, (len(X_test), 28*28))

noise_factor = 0.5
X_test_noisy = X_test + noise_factor * np.random.normal(loc=0.0, scale=1.0, size=X_test.shape)
X_test_noisy = np.clip(X_test_noisy, 0., 1.)

input_img = Input(shape=(784,))
encoded = Dense(128, activation='relu')(input_img)
encoded = Dense(64, activation='relu')(encoded)
encoded = Dense(32, activation='relu')(encoded)
```

```
decoded = Dense(64, activation='relu')(encoded)
decoded = Dense(128, activation='relu')(decoded)
decoded = Dense(784, activation='sigmoid')(decoded)

autoencoder = Model(input_img, decoded)
autoencoder.compile(optimizer='adam', loss='binary_crossentropy')
autoencoder.fit(X_train, X_train, epochs=10, batch_size=256, shuffle=True, validation_data=(X_test, X_test))

reconstructed_imgs = autoencoder.predict(X_test)
mse = np.mean(np.square(X_test - reconstructed_imgs), axis=1)

threshold = np.mean(mse) + 2 * np.std(mse)

anomalies = X_test[mse > threshold]
n = 10
plt.figure(figsize=(20, 4))
for i in range(n):
    ax = plt.subplot(2, n, i + 1)
    plt.imshow(X_test[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)

    ax = plt.subplot(2, n, i + 1 + n)
    plt.imshow(anomalies[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()
```

THE WOW IN YOUR SOLUTION

The "wow" factor in our solution for anomaly detection with autoencoders lies in its capability to autonomously learn and adapt to complex data patterns without the need for labeled anomaly data. By harnessing the power of unsupervised learning, our system can detect anomalies in real-time, even in highly dynamic environments, providing proactive insights into potential threats or irregularities before they escalate. Moreover, its ability to accurately identify anomalies in high-dimensional data streams sets it apart, enabling precise anomaly detection in diverse applications such as cybersecurity, finance, and manufacturing.

MODELLING

In modeling for anomaly detection with autoencoders, we first preprocess the data and select an appropriate architecture for the autoencoder. Then, we train the autoencoder on normal data instances, optimizing its parameters to minimize reconstruction error. Following training, we use the autoencoder to reconstruct both normal and anomalous data, calculating the reconstruction error for each instance

RESULTS

Demo Link