Task 2 (10 points)

Describe the convolutional neural network. The answer should be comprehensive. (5 points)

The practical example of application in cybersecurity with data and the python code are mandatory. (5 points)

Place your answer into the task2.pdf file. The source code and the data should be included in the resulting pdf file. No additional files are required.

Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a type of deep learning architecture specifically designed for processing data with grid-like topology, such as images, videos, and signals. They are inspired by the visual cortex of the human brain and have proven to be highly effective in computer vision and pattern recognition tasks.

The key components of a CNN include:

Convolutional Layer: This layer applies a set of learnable filters (kernels) to the input data, performing a convolution operation. These filters extract local features and patterns from the input data.

Pooling Layer: This layer performs a down-sampling operation on the output of the convolutional layer, reducing the spatial dimensions while retaining the most important features. Common pooling operations include max-pooling and average-pooling.

Activation Function: Non-linear activation functions, such as ReLU (Rectified Linear Unit), are applied after each convolutional and pooling layer to introduce non-linearity and enhance the model's ability to learn complex patterns.

Fully Connected Layer: After the convolutional and pooling layers, the output is flattened and fed into one or more fully connected layers, similar to traditional feed-forward neural networks, for final classification or regression tasks.

The convolutional layers, along with the pooling layers, enable CNNs to automatically learn and extract relevant features from the input data, while maintaining translation invariance and reducing the computational complexity.

Cybersecurity Application: Network Traffic Analysis

CNNs can be applied to various cybersecurity tasks, including network traffic analysis for intrusion detection or malware classification. By treating network traffic data as image-like data, CNNs can learn patterns and features that distinguish normal traffic from malicious traffic or malware.

Sample Data:

Assume we have a dataset of network traffic data, where each traffic sample is represented as a 2D matrix (e.g., 32x32 pixels) encoding features such as source and destination IP addresses, ports, protocols, and packet payloads.

We can generate a small random sample dataset:

```
import numpy as np

from sklearn.model_selection import train_test_split

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

# Generate toy data

data = np.random.randint(0, 256, size=(1000, 32, 32, 1))

labels = np.random.randint(0, 2, size=(1000, 1))

# Split into train/test

X_train, X_test, y_train, y_test = train_test_split(data, labels, test_size=0.2)

# Build CNN model

model = Sequential()

model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 1)))

model.add(MaxPooling2D((2, 2)))

model.add(Conv2D(64, (3, 3), activation='relu'))

model.add(MaxPooling2D((2, 2)))

model.add(Flatten())

model.add(Dense(128, activation='relu'))

model.add(Dense(1, activation='sigmoid'))

# Compile and train

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

model.fit(X_train, y_train, epochs=20, batch_size=32, validation_data=(X_test, y_test))
```

This code builds a CNN with 2 convolutional layers, 2 max-pooling layers, and 2 fully connected layers to classify network traffic as normal or an attack based on the input traffic matrices. The model is trained on the generated toy data.