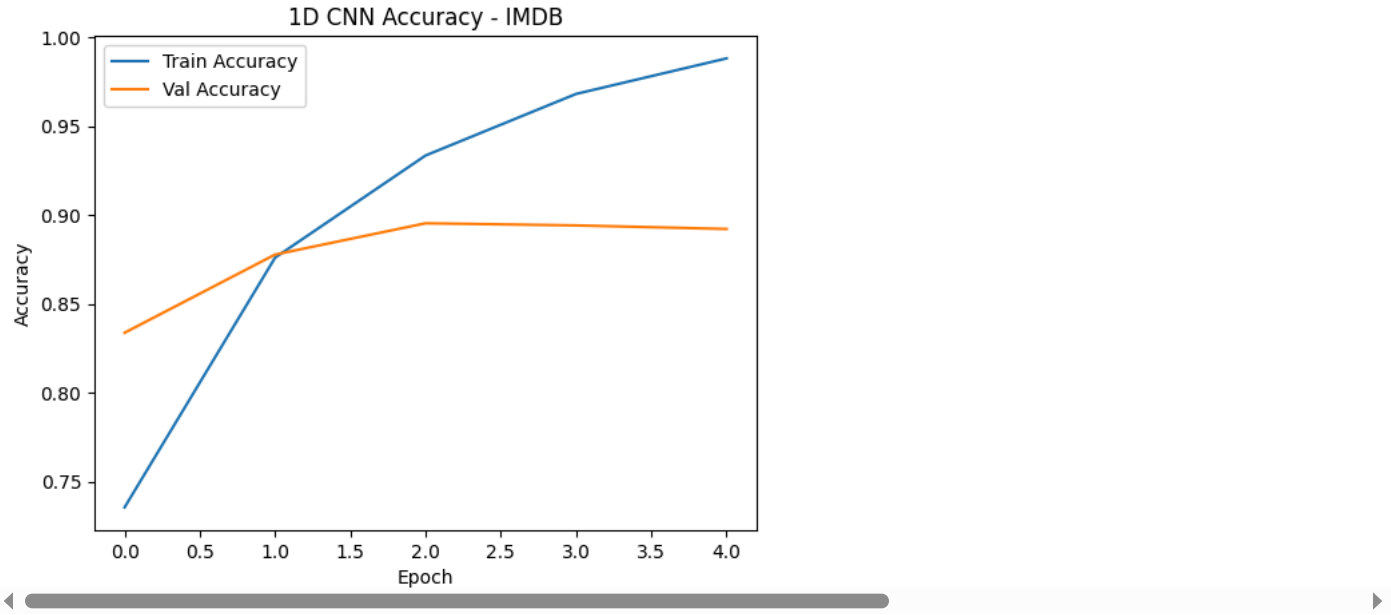
```
# Task 1: 1D CNN for IMDB Dataset
import numpy as np
import matplotlib.pyplot as plt
from keras.datasets import imdb
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Embedding, Conv1D, GlobalMaxPooling1D, Dense

# Load IMDB dataset
max_features = 10000  # Top 10,000 words
maxlen = 500          # Max review length

(x_train, y_train), (x_test, y_test) = imdb.load_data(num_words=max_features)

# Padding
x_train = pad_sequences(x_train, maxlen=maxlen)
x_test = pad_sequences(x_test, maxlen=maxlen)

# Build 1D CNN model
model_1d = Sequential([
    Embedding(max_features, 128, input_length=maxlen),
    Conv1D(64, 5, activation='relu'),
    GlobalMaxPooling1D(),
    Dense(1, activation='sigmoid')
])

model_1d.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Train the model
history_1d = model_1d.fit(x_train, y_train, epochs=5, batch_size=128, validation_split=0.2)

# Evaluate
score_1d = model_1d.evaluate(x_test, y_test)
print("Test Accuracy (1D CNN):", score_1d[1])

# Plot
plt.plot(history_1d.history['accuracy'], label='Train Accuracy')
plt.plot(history_1d.history['val_accuracy'], label='Val Accuracy')
plt.title('1D CNN Accuracy - IMDB')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

```
Epoch 1/5
157/157 ━━━━━━━━━━━━━━━━━━━━ 82s 512ms/step - accuracy: 0.6533 - loss: 0.6330 - val_accuracy: 0.8338 - val_loss: 0.3942
Epoch 2/5
157/157 ━━━━━━━━━━━━━━━━━━━━ 81s 506ms/step - accuracy: 0.8651 - loss: 0.3286 - val_accuracy: 0.8778 - val_loss: 0.2924
Epoch 3/5
157/157 ━━━━━━━━━━━━━━━━━━━━ 79s 503ms/step - accuracy: 0.9320 - loss: 0.1985 - val_accuracy: 0.8954 - val_loss: 0.2650
Epoch 4/5
157/157 ━━━━━━━━━━━━━━━━━━━━ 80s 509ms/step - accuracy: 0.9722 - loss: 0.1056 - val_accuracy: 0.8942 - val_loss: 0.2728
Epoch 5/5
157/157 ━━━━━━━━━━━━━━━━━━━━ 77s 493ms/step - accuracy: 0.9903 - loss: 0.0569 - val_accuracy: 0.8922 - val_loss: 0.2979
782/782 ━━━━━━━━━━━━━━━━━━━━ 26s 33ms/step - accuracy: 0.8767 - loss: 0.3211
Test Accuracy (1D CNN): 0.8799200057983398
```




```
# Task 2: 2D CNN for MNIST
import tensorflow as tf
from tensorflow.keras.datasets import mnist
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

# Load MNIST data
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Normalize and reshape
x_train = x_train.astype('float32') / 255.0
x_test = x_test.astype('float32') / 255.0
x_train = x_train[..., np.newaxis]
x_test = x_test[..., np.newaxis]

# One-hot encoding
y_train = to_categorical(y_train, 10)
y_test = to_categorical(y_test, 10)

# Build 2D CNN model
model_2d = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(28,28,1)),
    MaxPooling2D((2,2)),
    Flatten(),
    Dense(128, activation='relu'),
    Dense(10, activation='softmax')
])

model_2d.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Train model
history_2d = model_2d.fit(x_train, y_train, epochs=5, batch_size=128, validation_split=0.2)

# Evaluate
score_2d = model_2d.evaluate(x_test, y_test)
print("Test Accuracy (2D CNN):", score_2d[1])

# Plot
plt.plot(history_2d.history['accuracy'], label='Train Accuracy')
```

```
plt.plot(history_2d.history['val_accuracy'], label='Val Accuracy')
plt.title('2D CNN Accuracy - MNIST')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
/usr/local/lib/python3.11/dist-packages/keras/src/layers/convolutional/base_conv.py:107: UserWarning: Do not pass an `input_shape`/`inpu
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/5
375/375 ━━━━━━━━━━━━━━━━━━━━ 28s 70ms/step - accuracy: 0.8567 - loss: 0.5123 - val_accuracy: 0.9737 - val_loss: 0.0948
Epoch 2/5
375/375 ━━━━━━━━━━━━━━━━━━━━ 26s 69ms/step - accuracy: 0.9768 - loss: 0.0814 - val_accuracy: 0.9783 - val_loss: 0.0720
Epoch 3/5
375/375 ━━━━━━━━━━━━━━━━━━━━ 40s 66ms/step - accuracy: 0.9852 - loss: 0.0500 - val_accuracy: 0.9826 - val_loss: 0.0598
Epoch 4/5
375/375 ━━━━━━━━━━━━━━━━━━━━ 25s 67ms/step - accuracy: 0.9884 - loss: 0.0383 - val_accuracy: 0.9842 - val_loss: 0.0544
Epoch 5/5
375/375 ━━━━━━━━━━━━━━━━━━━━ 40s 66ms/step - accuracy: 0.9915 - loss: 0.0272 - val_accuracy: 0.9852 - val_loss: 0.0532
313/313 ━━━━━━━━━━━━━━━━━━━━ 2s 6ms/step - accuracy: 0.9806 - loss: 0.0586
Test Accuracy (2D CNN): 0.9847999811172485
```