

NEURAL NETWORKS AND DEEP LEARNING

ASSIGNMENT -6

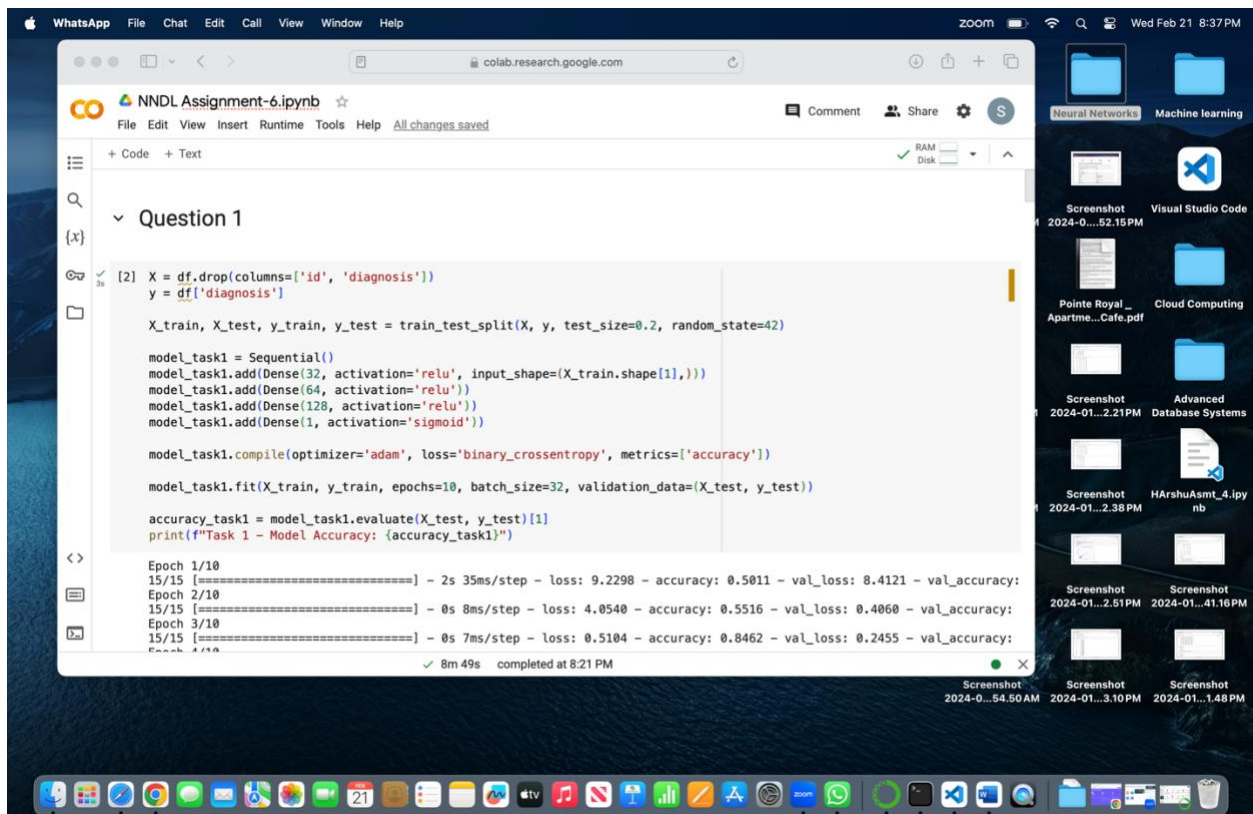
Sai Vardhan Reddy Narra

700756163

GitHub link: https://github.com/saivardhan-dev/Neural_networks_Assignment-6_700756163

Video link: https://drive.google.com/file/d/1iMdkFtxV4n-lzjw-ec9vwy-pwOVHQe7V/view?usp=drive_link

Question :1



```
colab.research.google.com

NNDL Assignment-6.ipynb
File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

Question 1

[2] X = df.drop(columns=['id', 'diagnosis'])
    y = df['diagnosis']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model_task1 = Sequential()
model_task1.add(Dense(32, activation='relu', input_shape=X_train.shape[1],))
model_task1.add(Dense(64, activation='relu'))
model_task1.add(Dense(128, activation='relu'))
model_task1.add(Dense(1, activation='sigmoid'))

model_task1.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

model_task1.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test))

accuracy_task1 = model_task1.evaluate(X_test, y_test)[1]
print(f"Task 1 - Model Accuracy: {accuracy_task1}")

Epoch 1/10
15/15 [=====] - 2s 35ms/step - loss: 9.2298 - accuracy: 0.5011 - val_loss: 8.4121 - val_accuracy:
Epoch 2/10
15/15 [=====] - 0s 8ms/step - loss: 4.0540 - accuracy: 0.5516 - val_loss: 0.4060 - val_accuracy:
Epoch 3/10
15/15 [=====] - 0s 7ms/step - loss: 0.5104 - accuracy: 0.8462 - val_loss: 0.2455 - val_accuracy:
Epoch 4/10
15/15 [=====] - 0s 7ms/step - loss: 0.5104 - accuracy: 0.8462 - val_loss: 0.2455 - val_accuracy:
Epoch 5/10
15/15 [=====] - 0s 7ms/step - loss: 0.5104 - accuracy: 0.8462 - val_loss: 0.2455 - val_accuracy:
Epoch 6/10
15/15 [=====] - 0s 7ms/step - loss: 0.5104 - accuracy: 0.8462 - val_loss: 0.2455 - val_accuracy:
Epoch 7/10
15/15 [=====] - 0s 7ms/step - loss: 0.5104 - accuracy: 0.8462 - val_loss: 0.2455 - val_accuracy:
Epoch 8/10
15/15 [=====] - 0s 7ms/step - loss: 0.5104 - accuracy: 0.8462 - val_loss: 0.2455 - val_accuracy:
Epoch 9/10
15/15 [=====] - 0s 7ms/step - loss: 0.5104 - accuracy: 0.8462 - val_loss: 0.2455 - val_accuracy:
Epoch 10/10
15/15 [=====] - 0s 7ms/step - loss: 0.5104 - accuracy: 0.8462 - val_loss: 0.2455 - val_accuracy:

8m 49s completed at 8:21 PM
```

Safari File Edit View History Bookmarks Window Help

colab.research.google.com

NNDL Assignment-6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
task 1 - Model Accuracy: 0.33039407042001
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

df = pd.read_csv('/content/Breast_Cancer.csv')
df = df.drop(columns=['Unnamed: 32'], errors='ignore')
df['diagnosis'] = df['diagnosis'].map({'M': 1, 'B': 0})

print(df.head())
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean
0	842302	1	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710
1	842517	1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017
2	84300903	1	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790
3	84348301	1	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520
4	84358402	1	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430

8m 49s completed at 8:21 PM

Finder File Edit View Go Window Help

colab.research.google.com

NNDL Assignment-6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
scaler = StandardScaler()
X_train_normalized = scaler.fit_transform(X_train)
X_test_normalized = scaler.transform(X_test)

model_task3 = Sequential()
model_task3.add(Dense(32, activation='relu', input_shape=(X_train_normalized.shape[1],)))
model_task3.add(Dense(64, activation='relu'))
model_task3.add(Dense(1, activation='sigmoid'))

model_task3.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

model_task3.fit(X_train_normalized, y_train, epochs=10, batch_size=32, validation_data=(X_test_normalized, y_test))

accuracy_task3 = model_task3.evaluate(X_test_normalized, y_test)[1]
print(f"Task 3 - Model Accuracy with Normalization: {accuracy_task3}")
```

Epoch 1/10
15/15 [=====] - 1s 30ms/step - loss: 0.5384 - accuracy: 0.7802 - val_loss: 0.3873 - val_accuracy:
Epoch 2/10
15/15 [=====] - 0s 12ms/step - loss: 0.3287 - accuracy: 0.9165 - val_loss: 0.2335 - val_accuracy:
Epoch 3/10
15/15 [=====] - 0s 9ms/step - loss: 0.2160 - accuracy: 0.9385 - val_loss: 0.1526 - val_accuracy:
Epoch 4/10
15/15 [=====] - 0s 9ms/step - loss: 0.1613 - accuracy: 0.9429 - val_loss: 0.1136 - val_accuracy:
Epoch 5/10
15/15 [=====] - 0s 10ms/step - loss: 0.1295 - accuracy: 0.9560 - val_loss: 0.0934 - val_accuracy:
Epoch 6/10
15/15 [=====] - 0s 8ms/step - loss: 0.1050 - accuracy: 0.9692 - val_loss: 0.0825 - val_accuracy:
Epoch 7/10
15/15 [=====] - 0s 9ms/step - loss: 0.0913 - accuracy: 0.9736 - val_loss: 0.0748 - val_accuracy:
Epoch 8/10

8m 49s completed at 8:21 PM

Question 2:

The screenshot shows a Google Colab notebook titled "NNDL Assignment-6.ipynb". The notebook is running a Keras model training process on the MNIST dataset. The code in the notebook is as follows:

```
[4] history = model.fit(train_images, train_labels, epochs=10, validation_data=(test_images, test_labels))

plt.figure(figsize=(12, 4))

plt.subplot(1, 2, 1)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend(['Train', 'Validation'], loc='upper right')

plt.subplot(1, 2, 2)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend(['Train', 'Validation'], loc='lower right')

plt.tight_layout()
plt.show()
```

The output of the training process is displayed in the console:

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [=====] - 0s 0us/step
Epoch 1/10
1875/1875 [=====] - 51s 27ms/step - loss: 0.1515 - accuracy: 0.9546 - val_loss: 0.0525 - val_accu
Epoch 2/10
1875/1875 [=====] - 52s 28ms/step - loss: 0.0465 - accuracy: 0.9855 - val_loss: 0.0734 - val_accu
```

The notebook interface shows the "Code" tab selected, and the output of the training process is visible in the console. The background of the Colab window shows a desktop environment with various files and folders, including "Neural Networks", "Machine learning", "Visual Studio Code", "Cloud Computing", "Advanced Database Systems", and "HarshuAsmt_4.ipynb".

Finder File Edit View Go Window Help zoom Wed Feb 21 9:03 PM

colab.research.google.com

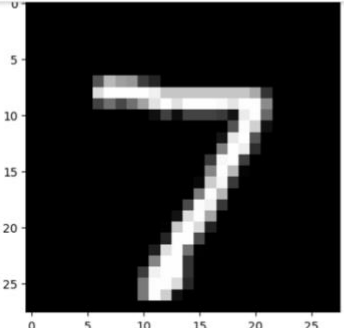
NNDL Assignment-6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

[5]



1/1 [=====] - 0s 173ms/step
Model Prediction: 7

[6]

```
modified_model = models.Sequential()  
modified_model.add(layers.Conv2D(32, (3, 3), activation='tanh', input_shape=(28, 28, 1)))  
modified_model.add(layers.MaxPooling2D((2, 2)))  
modified_model.add(layers.Conv2D(64, (3, 3), activation='tanh'))  
modified_model.add(layers.MaxPooling2D((2, 2)))  
modified_model.add(layers.Flatten())  
modified_model.add(layers.Dense(128, activation='tanh'))  
modified_model.add(layers.Dense(10, activation='softmax'))  
  
modified_model.compile(optimizer='adam',  
                        loss='sparse_categorical_crossentropy',  
                        metrics=['accuracy'])  
  
modified_history = modified_model.fit(train_images, train_labels, epochs=10, validation_data=(test_images, test_labels))  
  
plt.figure(figsize=(12, 4))  
  
plt.subplot(1, 2, 1)  
plt.plot(modified_history.history['loss'])  
plt.plot(modified_history.history['val_loss'])  
plt.title('Modified Model Loss')  
plt.xlabel('Epoch')  
plt.ylabel('Loss')  
plt.legend(['Train', 'Validation'], loc='upper right')  
  
plt.subplot(1, 2, 2)  
plt.plot(modified_history.history['accuracy'])  
plt.plot(modified_history.history['val_accuracy'])  
plt.title('Modified Model Accuracy')
```

8m 49s completed at 8:21 PM

Finder File Edit View Go Window Help zoom Wed Feb 21 9:03 PM

colab.research.google.com

NNDL Assignment-6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

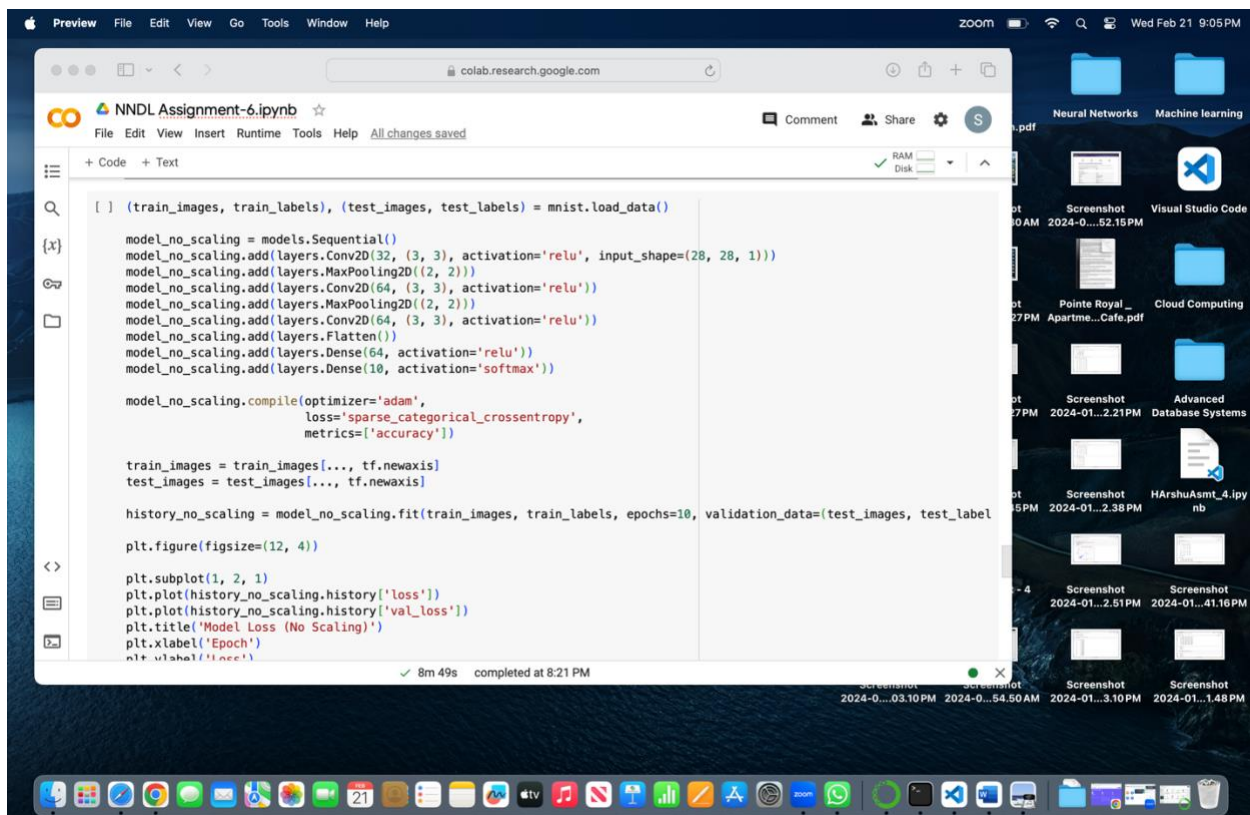
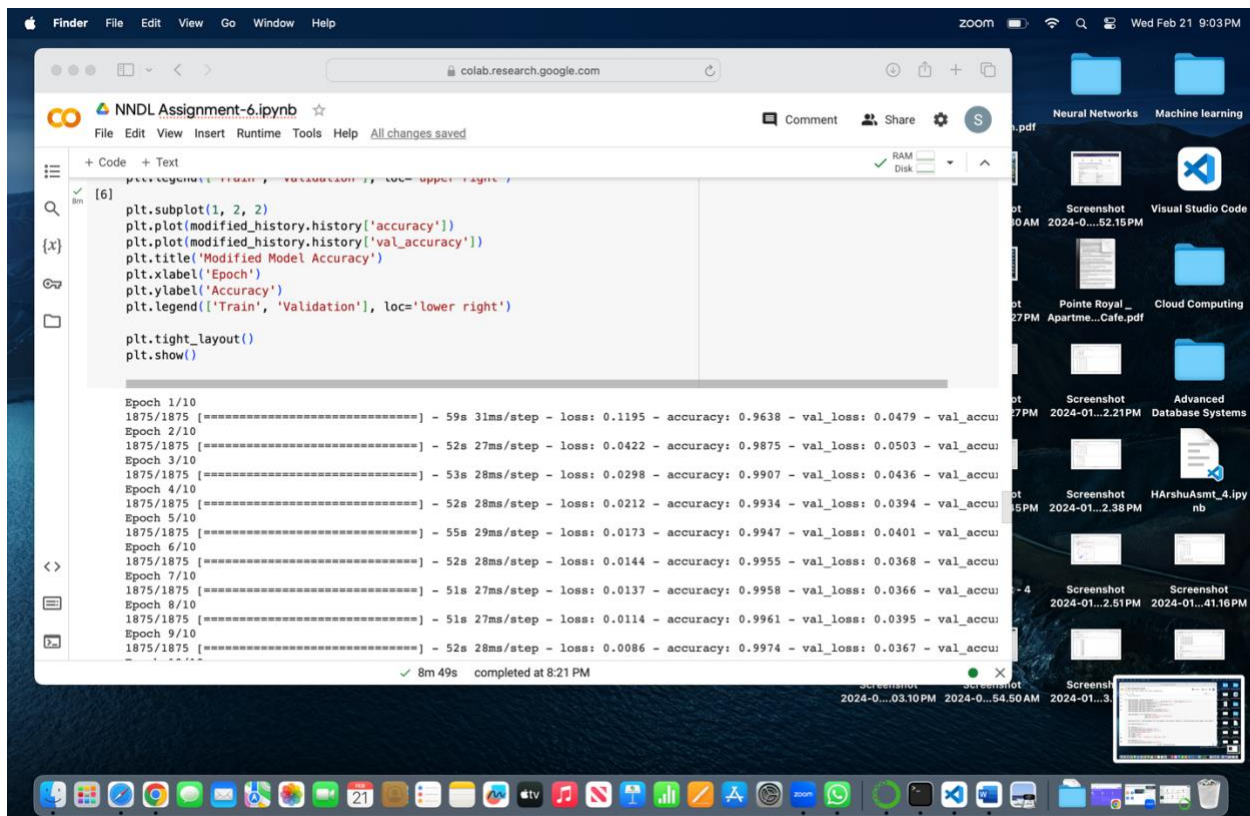
RAM Disk

Model Prediction: 7

[6]

```
modified_model = models.Sequential()  
modified_model.add(layers.Conv2D(32, (3, 3), activation='tanh', input_shape=(28, 28, 1)))  
modified_model.add(layers.MaxPooling2D((2, 2)))  
modified_model.add(layers.Conv2D(64, (3, 3), activation='tanh'))  
modified_model.add(layers.MaxPooling2D((2, 2)))  
modified_model.add(layers.Flatten())  
modified_model.add(layers.Dense(128, activation='tanh'))  
modified_model.add(layers.Dense(10, activation='softmax'))  
  
modified_model.compile(optimizer='adam',  
                        loss='sparse_categorical_crossentropy',  
                        metrics=['accuracy'])  
  
modified_history = modified_model.fit(train_images, train_labels, epochs=10, validation_data=(test_images, test_labels))  
  
plt.figure(figsize=(12, 4))  
  
plt.subplot(1, 2, 1)  
plt.plot(modified_history.history['loss'])  
plt.plot(modified_history.history['val_loss'])  
plt.title('Modified Model Loss')  
plt.xlabel('Epoch')  
plt.ylabel('Loss')  
plt.legend(['Train', 'Validation'], loc='upper right')  
  
plt.subplot(1, 2, 2)  
plt.plot(modified_history.history['accuracy'])  
plt.plot(modified_history.history['val_accuracy'])  
plt.title('Modified Model Accuracy')
```

8m 49s completed at 8:21 PM



The screenshot displays a Google Colab notebook titled "NNDL Assignment-6.ipynb". The notebook's interface includes a top menu bar with options like File, Edit, View, Insert, Runtime, Tools, and Help. Below the menu, there's a toolbar with icons for code, text, and other functions. The main area shows the notebook's content, which includes training logs and two line graphs.

The training logs show the following data for 10 epochs:

Epoch	Time	Loss	Accuracy	Val Loss	Val Accuracy
0	54s	0.2440	0.9868	0.0420	0.9868
1	29ms	0.0394	0.9879	0.0476	0.9879
2	55s	0.0394	0.9879	0.0476	0.9879
3	58s	0.0343	0.9894	0.0569	0.9894
4	54s	0.0293	0.9908	0.0425	0.9908
5	29ms	0.0260	0.9924	0.0630	0.9924
6	57s	0.0260	0.9924	0.0630	0.9924
7	54s	0.0219	0.9933	0.0482	0.9933
8	29ms	0.0219	0.9933	0.0482	0.9933
9	54s	0.0226	0.9929	0.0484	0.9929
10	29ms	0.0226	0.9929	0.0484	0.9929

Below the logs, there are two line graphs:

- Model Loss (No Scaling):** This graph plots Loss (Y-axis, 0.05 to 0.25) against Epoch (X-axis, 0 to 10). The training loss (blue line) starts at approximately 0.24 and drops sharply to about 0.03 by epoch 1, then continues to decrease slowly to around 0.02 by epoch 10. The validation loss (orange line) starts at approximately 0.04 and remains relatively stable, fluctuating between 0.04 and 0.06 throughout the 10 epochs.
- Model Accuracy (No Scaling):** This graph plots Accuracy (Y-axis, 0.94 to 0.99) against Epoch (X-axis, 0 to 10). The training accuracy (blue line) starts at approximately 0.98 and increases sharply to about 0.99 by epoch 1, then continues to increase slowly to around 0.995 by epoch 10. The validation accuracy (orange line) starts at approximately 0.98 and increases sharply to about 0.99 by epoch 1, then continues to increase slowly to around 0.99 by epoch 10.

The notebook interface also shows a file explorer on the left, a code editor in the center, and a terminal at the bottom. The status bar at the bottom indicates the notebook is completed at 8:21 PM.