Plots

May 1, 2025

1 Importing Necessary Libraries

[87]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report
import numpy as np
```

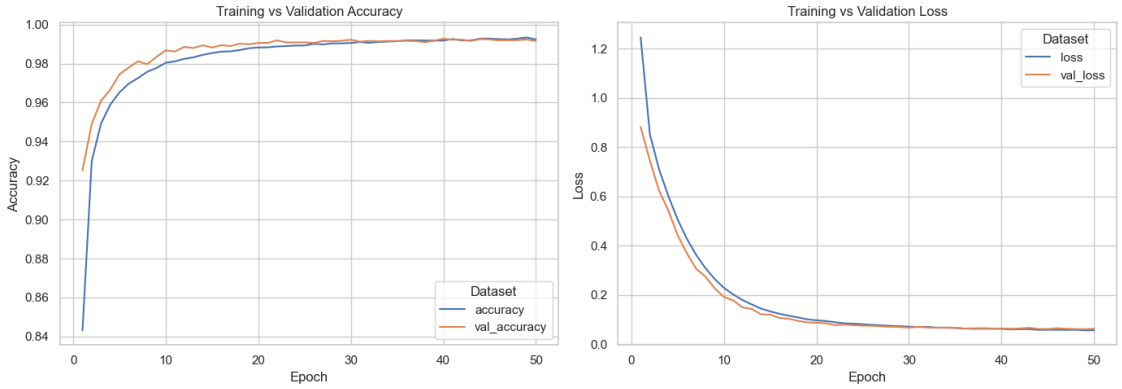
2 Loss and Accuracy Graph Plots

2.1 Loading the Model History DataFrame

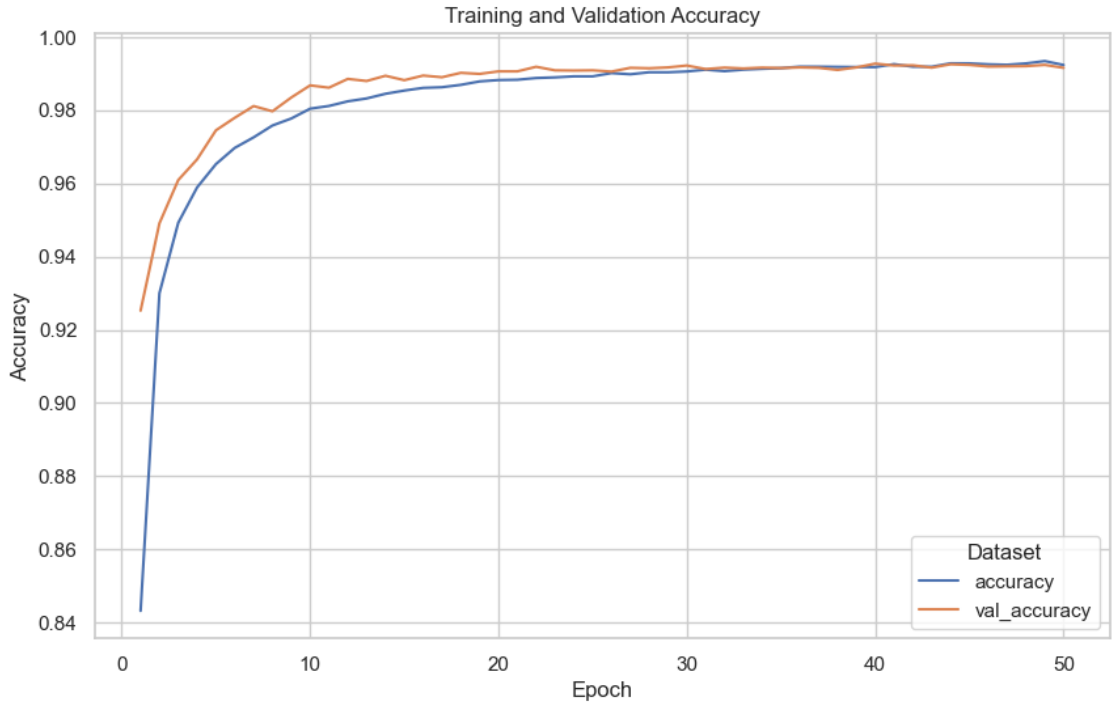
[ ]:
```
hist_df = pd.read_csv("History.csv")
hist_df["epoch"] = hist_df.index + 1

accuracy_df = hist_df.melt(id_vars = "epoch", value_vars=["accuracy",␣
 ↪"val_accuracy"], var_name="Dataset", value_name="Accuracy")
loss_df = hist_df.melt(id_vars = "epoch", value_vars = ["loss", "val_loss"],␣
 ↪var_name = "Dataset", value_name=  "Loss")
```

[ ]:
```
sns.set(style="whitegrid")

fig, axes = plt.subplots(1, 2, figsize=(14, 5))

# Accuracy plot
sns.lineplot(data=accuracy_df, x="epoch", y="Accuracy", hue="Dataset",␣
 ↪ax=axes[0])
axes[0].set_title("Training vs Validation Accuracy")
axes[0].set_xlabel("Epoch")
axes[0].set_ylabel("Accuracy")

# Loss plot
sns.lineplot(data=loss_df, x="epoch", y="Loss", hue="Dataset", ax=axes[1])
axes[1].set_title("Training vs Validation Loss")
axes[1].set_xlabel("Epoch")
```

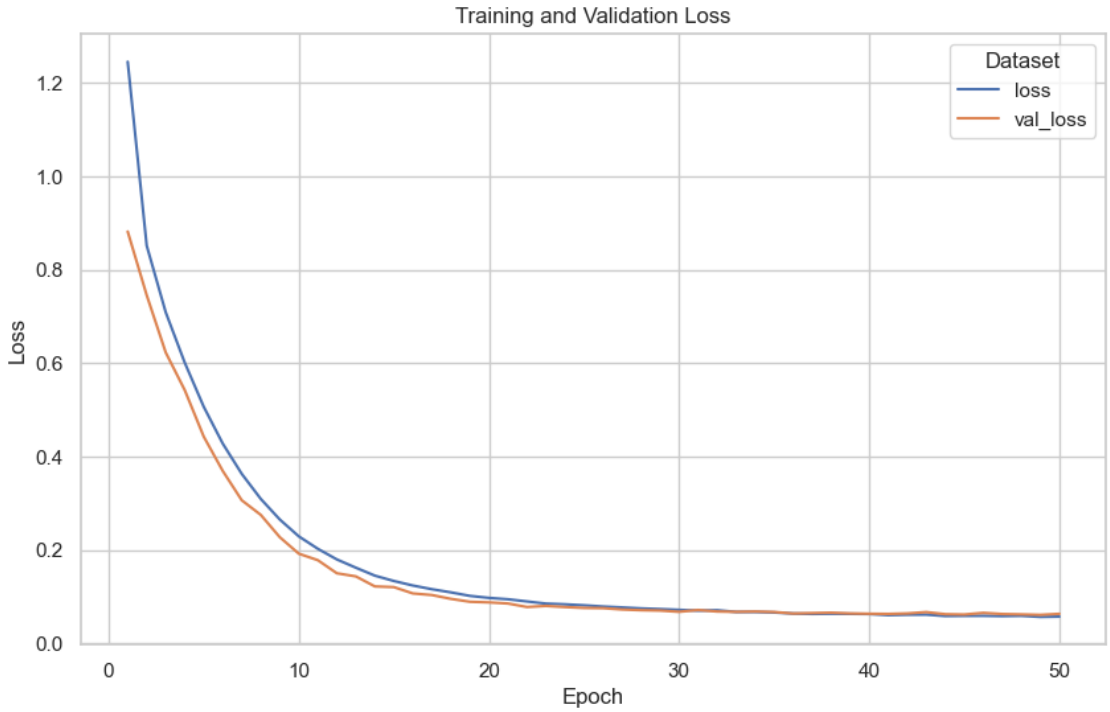
```
axes[1].set_ylabel("Loss")

plt.tight_layout()

plt.savefig("Plots/Combined Graphs.png", dpi=300)  # High-res PNG
plt.show()
```



[ ]:
```
sns.set(style="whitegrid")
plt.figure(figsize=(10, 6))
sns.lineplot(data=accuracy_df, x="epoch", y="Accuracy", hue="Dataset")
plt.title("Training and Validation Accuracy")
plt.xlabel("Epoch")
plt.ylabel("Accuracy")
plt.legend(title="Dataset")
plt.savefig("Plots/Accuracy Graphs.png", dpi=300)  # High-res PNG
plt.show()
```



[ ]:
```
plt.figure(figsize=(10, 6))
sns.lineplot(data=loss_df, x="epoch", y="Loss", hue="Dataset")
plt.title("Training and Validation Loss")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.legend(title="Dataset")
plt.savefig("Plots/Loss Graphs.png", dpi=300)
plt.show()
```



3 Confusion Matrix

3.1 Loading the Model and the Data

[8]:
```
model = tf.keras.models.load_model("../Models/Model.h5")
```

[53]:
```
X = np.load("../Features.npy")
y = np.load("../Labels.npy")
```

3.2 Splitting the Data and Performing Model Prediction

[55]:
```
_, X_test, _, y_test = train_test_split(X, y, shuffle = True, random_state =␣
↪10, test_size = 0.2)
```

[57]:
```
X_test.shape, y_test.shape
```

[57]:
```
((21884, 250, 2), (21884,))
```

[59]:
```
y_pred_probs = model.predict(X_test)
y_pred = y_pred_probs.argmax(axis = 1)
```

```
684/684 [==============================] - 3s 4ms/step
```

[69]:
```
y_true = y_test
```

3.2.1 Check for Missing Labels in y_true and y_pred

Carefully check the labels of both the True and the Predicted labels, then proceed to plotting the Confusion Matrix

[71]:
```
values, _ = np.unique(y_true, return_counts = True)
```

[73]:
```
values
```

[73]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13],
      dtype=int64)
```

[75]:
```
np.unique(y_true, return_counts = True)
```

[75]:
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13],
       dtype=int64),
 array([ 1352,   501,    21,   149,    22,  1620, 15135,  1411,     1,
         1407,    28,     6,   191,    40], dtype=int64))
```

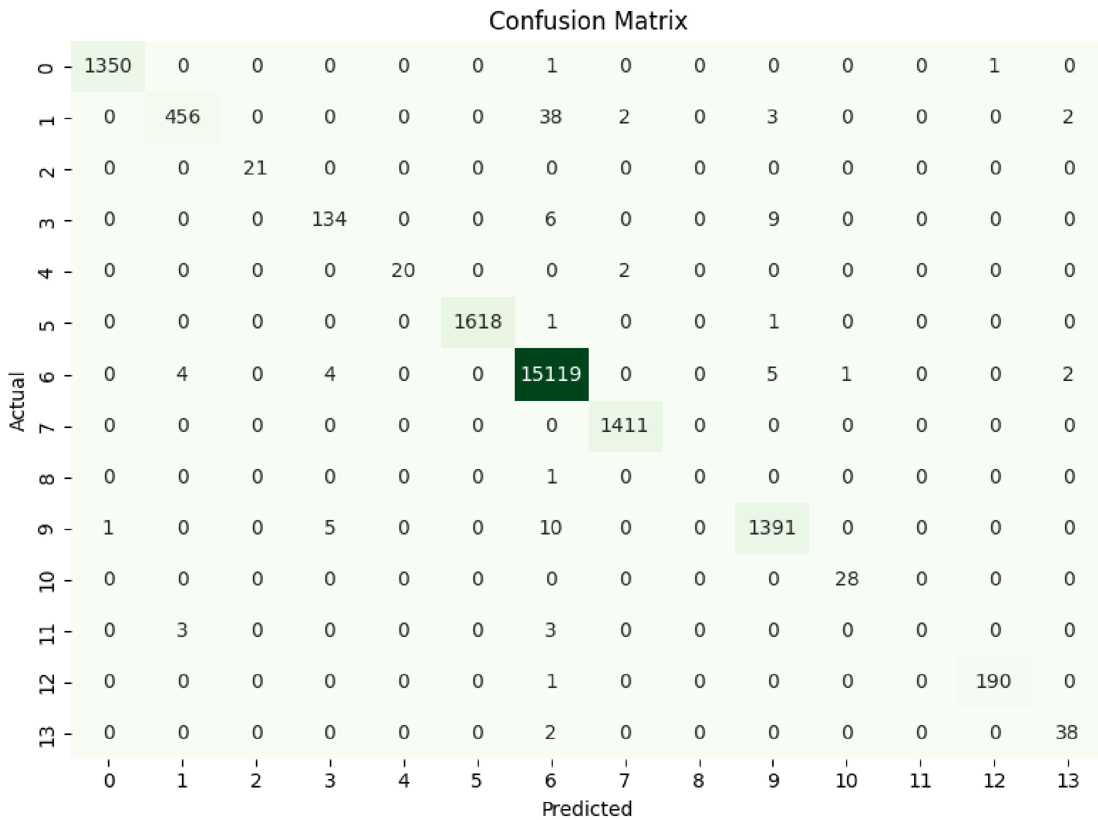
[77]:
```
np.unique(y_pred, return_counts = True)
```

[77]:
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  9, 10, 12, 13], dtype=int64),
 array([ 1351,   463,    21,   143,    20,  1618, 15182,  1415,  1409,
           29,   191,    42], dtype=int64))
```

3.3 Plotting

[80]:
```
cm = confusion_matrix(y_true, y_pred)
```

[82]:
```
plt.figure(figsize = (8, 6))
sns.heatmap(cm, annot = True, fmt = "d", cmap = "Greens", cbar = False,␣
 ↪xticklabels = values, yticklabels = values)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.tight_layout()

# Save the plot
plt.savefig("../Plots/Confusion Matrix.png", dpi=300)
plt.show()
```



4 Classification Report

[89]:
```
classification_report(y_true, y_pred)
```

```
C:\Users\neelo\anaconda3\envs\Tensorflow-GPU\lib\site-
packages\sklearn\metrics\_classification.py:1497: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\neelo\anaconda3\envs\Tensorflow-GPU\lib\site-
packages\sklearn\metrics\_classification.py:1497: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\neelo\anaconda3\envs\Tensorflow-GPU\lib\site-
packages\sklearn\metrics\_classification.py:1497: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

[89]:
```
'              precision    recall  f1-score   support\n\n           0
1.00      1.00      1.00      1352\n           1       0.98      0.91      0.95
501\n           2       1.00      1.00      1.00        21\n           3
0.94      0.90      0.92       149\n           4       1.00      0.91      0.95
22\n           5       1.00      1.00      1.00      1620\n           6
1.00      1.00      1.00     15135\n           7       1.00      1.00      1.00
1411\n           8       0.00      0.00      0.00         1\n           9
0.99      0.99      0.99      1407\n          10       0.97      1.00      0.98
28\n          11       0.00      0.00      0.00         6\n          12
0.99      0.99      0.99       191\n          13       0.90      0.95      0.93
40\n\n    accuracy                           1.00     21884\n   macro avg
0.84      0.83      0.84     21884\nweighted avg       0.99      1.00      0.99
21884\n'
```

[ ]: