Documenting a Hybrid ML Workflow using CNN, SVM and Weights & Biases

Kavya Reddy

April 16, 2025

1 Introduction

This project demonstrates the use of synthetic data for classification tasks by generating artificial feature datasets using `make_classification()` and applying CNN-based image classification on the real-world Fashion-MNIST dataset. It further enhances model performance through data augmentation and tracks experiments using Weights & Biases.

2 Installation and Imports

Listing 1: Installing and importing required libraries

```
!pip install --upgrade wandb
!pip install scikit-learn pydbgen

import wandb
import keras
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt

from keras import Sequential
from keras.layers import Conv2D, Flatten, Dense
from sklearn import svm
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from wandb.integration.keras import WandbCallback

wandb.login()
wandb.init(project="synth-data-normal", config={"hyper": "parameter"})
```

3 SVM Classification on Synthetic Data

```
X, y = make_classification(n_features=2, n_redundant=0, n_classes=2)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

plt.figure()
plt.scatter(X[:,0], X[:,1], c=y)
wandb.log({"chart": plt})

clf = svm.SVC(probability=True)
clf.fit(X_train, y_train)
```

```
y_pred = clf.predict(X_test)
y_probas = clf.predict_proba(X_test)

wandb.sklearn.plot_classifier(clf, X_train, X_test, y_train, y_test,
                              y_pred=y_pred, y_probas=y_probas,
                              labels=['1','2'], model_name='SVC')
```

4 CNN Classification on Fashion-MNIST

```
(x_train, y_train), (x_test, y_test) = tf.keras.datasets.fashion_mnist.load_data()

x_train = x_train.reshape(-1, 28, 28, 1).astype('float32') / 255.0
x_test = x_test.reshape(-1, 28, 28, 1).astype('float32') / 255.0
y_train = keras.utils.to_categorical(y_train)
y_test = keras.utils.to_categorical(y_test)

fig, axs = plt.subplots(2, 2)
for i, ax in enumerate(axs.flat):
    ax.imshow(np.squeeze(x_train[i]), cmap='gray')
    ax.axis('off')
wandb.log({"example_ims": fig})
```

Sample Fashion-MNIST Images

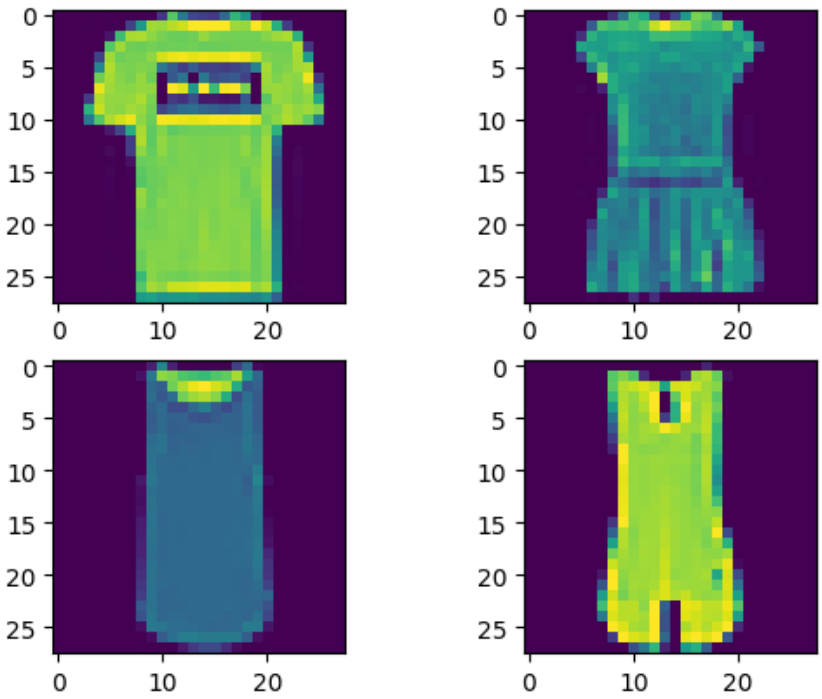

Figure 1: Sample images from the Fashion-MNIST dataset

5 CNN Architecture and Training

```
model = Sequential()
model.add(Conv2D(input_shape=(28, 28, 1), filters=4, kernel_size=4, strides=2, padding
    ='same'))
model.add(Conv2D(filters=4, kernel_size=4, strides=2, padding='same'))
model.add(Flatten())
model.add(Dense(10, activation='softmax'))

opt = tf.keras.optimizers.RMSprop(learning_rate=0.0001, decay=1e-6)
model.compile(opt, loss='categorical_crossentropy', metrics=['acc'])

model.fit(x_train, y_train, validation_data=(x_test, y_test), epochs=10)
```

Model Summary

Insert screenshot or 'model.summary()' output.

6 Prediction and Visualization

```
example = np.rot90(x_train[0].reshape(28, 28))
print(np.argmax(model.predict(example[np.newaxis, :, :, np.newaxis])))

plt.figure()
plt.imshow(np.squeeze(example))
wandb.log({"example_rot": fig})
```

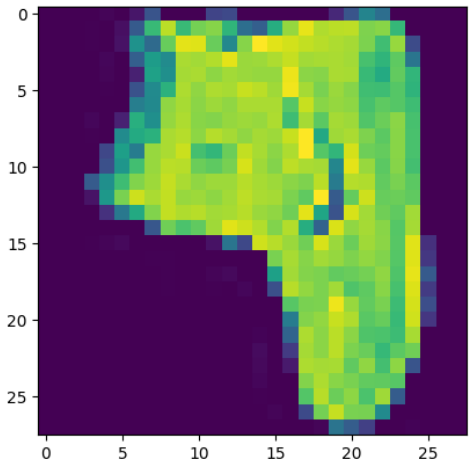

Figure 2: Prediction visualization of rotated example

7 Data Augmentation using ImageDataGenerator

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

datagen = ImageDataGenerator(rotation_range=180)
datagen.fit(x_train)

model.fit(datagen.flow(x_train, y_train, batch_size=32),
          epochs=10,
          validation_data=(x_test, y_test))
```

8 Conclusion

This project showcases the effectiveness of combining synthetic data generation and real-world datasets to build robust machine learning models. By utilizing tools like scikit-learn for synthetic data creation and Fashion-MNIST for real image classification, alongside data augmentation and experiment tracking via Weights & Biases, we establish a complete workflow that emphasizes both performance and reproducibility. This hybrid approach not only improves model generalization but also supports deeper experimentation in a controlled, flexible environment.