PAULINE ONYANGO

Part 1: Theoretical Understanding (40%)

1. Short Answer Questions

Q1: Explain the primary differences between TensorFlow and PyTorch. When would you choose one over the other?

TensorFlow and PyTorch are open-source deep learning frameworks that are also popular with the community; however, they have a number of differences:

Feature	TensorFlow	PyTorch
Execution Style	Static graph (TensorFlow 1.x), eager mode with TF 2.x	Dynamic computation graph (eager execution)
Ease of Debugging	More complex due to static graph	Easier due to Pythonic, step-by-step debugging
Deployment	Better support via TensorFlow Lite, TF Serving	Deployment options exist but require more setup
Ecosystem	Larger ecosystem (TFLite, TensorBoard, TF Hub)	Simpler ecosystem focused on research

When to choose:

- TensorFlow: Basically the same advantages as Pytorch (production-scale, model deployment (particularly mobile/edge), etc), but has better support for TensorBoard visualizations.
- PyTorch: Libraries favored in academic research and prototyping because of being flexible as well as Python-like in nature.

Q2: Describe two use cases for Jupyter Notebooks in AI development.

1. Interactive Model Prototyping: Jupyter enables developers to develop, test, experiment with architecture or parameters, cell by cell, providing quick iteration on models.
2. Data Exploration and Visualization: Its application is mainly to visualize data with libraries such as Matplotlib or Seaborn and enable practitioners to dialog with the data distributions, patterns, and outliers before training a model.

Q3: How does spaCy enhance NLP tasks compared to basic Python string operations?

The basic string operations lack sophisticated linguistic features and machine-learning-based models of text analysis which are offered by spaCy. Specifically:

- Named Entity Recognition (NER): Identifies people, brands, locations in text, which is not feasible with str.split() or re.
- Part-of-Speech (POS) Tagging: spaCy annotates tokens with grammar roles, aiding in syntactic parsing.
- Efficiency and Accuracy: spaCy is optimized in Cython for speed and is more accurate than hand-coded regex or simple Python logic.

2. Comparative Analysis

Scikit-learn vs. TensorFlow

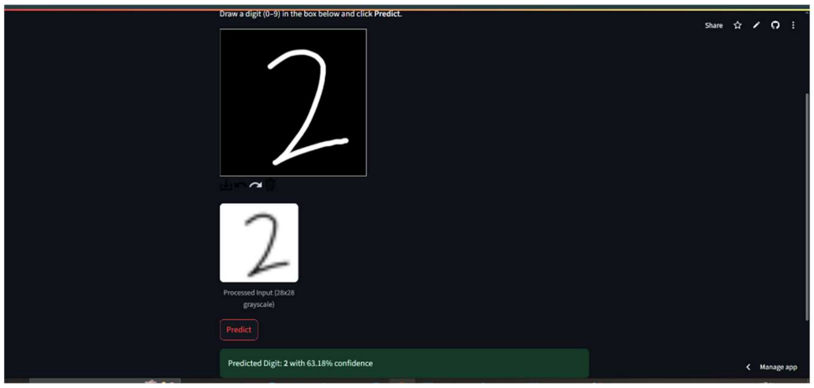
Category	**Scikit-learn**	**TensorFlow**
Target Applications	Classical ML (e.g., SVM, Random Forest, KNN)	Deep Learning (CNNs, RNNs, transformers)
Model Complexity	Shallow models, quick to train	Suitable for complex neural network models
Ease of Use	Very beginner-friendly; few lines of code needed	More complex API, requires deeper ML knowledge
Deployment	Limited built-in deployment tools	Rich deployment options (TFLite, TF Serving)
Community Support	Strong and mature community	Massive community and enterprise backing
Integration	Works well with pandas, NumPy, matplotlib	Integrates with Keras, TensorBoard, TFLite

Summary:

- Select Scikit-learn to use it with speedy experimentations and traditional ML applications.
- Select TensorFlow when you need the neural network and high production scale in deep learning applications.

MODEL OUTPUTS

Deep Learning with TensorFlow/PyTorch



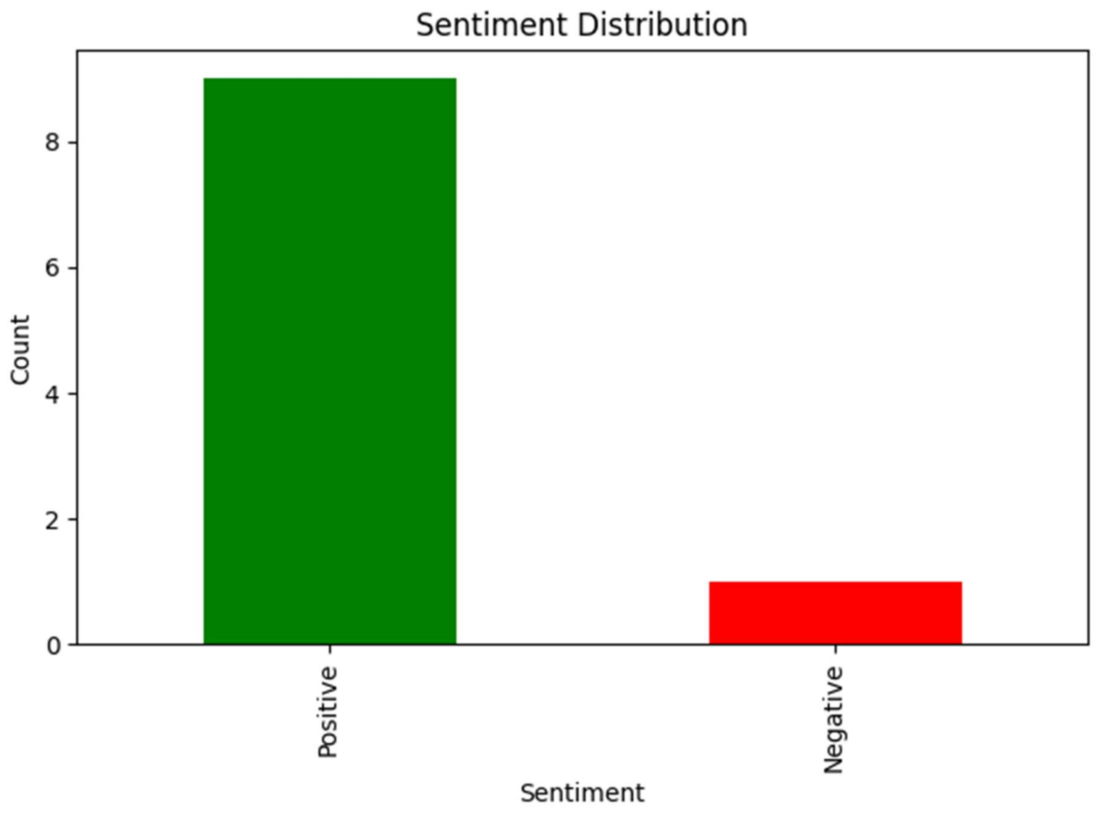
Draw a digit (0–9) in the box below and click **Predict**.

Processed Input (28x28 grayscale)

Predicted Digit: 2 with 63.18% confidence

Figure of CNN model to classify handwritten digits

NLP with spaCy using Amazon product Reviews



Sentiment distribution of Amazon Reviews

Part 3: Ethics & Optimization Report

1. Ethical Considerations

A. MNIST Model Biases

Although the MNIST has become a standard dataset and it is deemed to be non-biased, their training models my incur a number of possible biases:

- Homogeneity of Data: The dataset is on the ones written by the census workers and students, which would not represent all the forms of handwritings worldwide.
- Visual Contrast Assumptions: The model presupposes the black-on-white digits. Inverted-color images (black backgrounds, with white digits) produce poor performance unless specially addressed.
- Age of Dataset: MNIST has been over 20 years old. It would not take into consideration present day variations of inputs like stylus drawn or touchscreen input.

Mitigation Strategies

- Data Augmentation: add various and other types of styles to the digits, such as: inverted colours, thick and thin, and distortions.
- Input Normalization: bypassed normalization and automatically invert the pixel color to black-ground and white-digit.

Tools of Model Auditing:

  - TensorFlow Fairness Indicators: This is typically more applicable to demographic fairness, but it would be possible to reuse it to audit performance across digit varieties or in writer sets.
  - Custom Evaluation Metrics: Form test splits based on stylistic differences and compare model accuracy on each of the splits.

B. Amazon Reviews NLP Biases

The sentiment analysis and NER pipeline of Amazon Reviews is open to some ethical issues:

- Training Label Bias: The star-based sentiment labels do not necessarily demonstrate the proper tone or emotion.
- Polarity Bias: A word (e.g., cheap) may have different meanings in various contexts but rule-based tools such as TextBlob treat this word in the same way.
- Entity Recognition Gaps: spaCy may fail to recognize domain specific entities like not so common names of or names of brands or products which it prefers to recognize common or famous entities.

Mitigation Strategies

- Fine-Tuning: Enhance the model of spaCy by fine-tuning it with a domain-specific set of data.
- Rule-Based Layer Upgrade: Up-level the TextBlob or the spaCy sentiment with lists or patterns of keywords discussing a specialised language.
- Cross-Validation: Testers should check how accurately sentiment varies across product categories, or price to identify performance bias.
- Transparency: Reveal confidence of prediction and give explanations to the final consumer, particularly in reviews with mixed rating.

2. Troubleshooting Challenge – Buggy TensorFlow Code Fixes

Problem 1: Dimension Mismatch

Original Error:

model.add(Dense(10))

model.compile(loss='categorical_crossentropy', ...)

But labels were integers (0–9), not one-hot encoded.

Fix:
Change the loss function to:

loss='sparse_categorical_crossentropy'

This allows direct use of integer class labels.

Problem 2: Incorrect Input Shape

Original Error:

model.add(Input(shape=(28, 28)))

This caused shape errors in Conv2D, which expects 4D tensors (batch, height, width, channels).

Fix:
Explicitly reshape inputs:

x_train = x_train.reshape(-1, 28, 28, 1)

x_test = x_test.reshape(-1, 28, 28, 1)

And use:

```
model.add(Input(shape=(28, 28, 1)))
```

Problem 3: Evaluation Logic

Issue:
Model wrongly predicted a digit due to inverted image colors (white digit on black).

Fix:
Add image inversion logic before prediction:

```
if np.mean(image) > 127:
    image = 255 – image
```

Conclusion

Even such seemingly neutral activities as digit recognition or review analysis require bias and fairness. Transparency, rule-based logic, fairness tools may help unearth and reduce such biases even before they affect the outcome of diverse test sets. Similarly, the process of troubleshooting and debugging typical machine learning challenges, including the shape of the data, flawed model evaluation, among others, allows both a technically and ethically conscious model performance**.**