

AI Development Report: Theoretical Understanding and Ethical Reflection

This report presents theoretical insights, comparative analyses, and ethical reflections in Artificial Intelligence (AI) development, with a focus on TensorFlow, PyTorch, and related tools. The report also highlights practical debugging and optimization approaches in TensorFlow model implementation.

Part 1: Theoretical Understanding

1. Short Answer Questions

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

TensorFlow	PyTorch
Developed by Google, released in 2015	Developed by Facebook, released in 2016
It was deployed on Theano, which is a Python library	It was made using the Torch library.
Has a larger community	Has a smaller community
It believes in a static graph concept	It works on a dynamic graph concept
It requires the use of a debugger tool.	It has a dynamic computational process.
It is comparatively hard to learn	It is easy to learn and understand.
It has a higher level of functionality and provides a broad spectrum of choices to work on.	PyTorch has fewer features compared to TensorFlow.
It has a major benefit that the whole graph could be saved as a protocol buffer.	Pytorch uses a simple API that saves the entire weight of the model.
Default settings are well-defined in TensorFlow.	It requires the user to store everything in the device.
Its features or libraries include Sonnet, Ludwig, and Magenta, among others.	Its features or libraries include PYRO, Horizon, and CheXNet, among others.
It is used to automate things faster and make artificial intelligence-related products	Used by developers who are more research-oriented.

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

Used for rapid prototyping and experimentation, i.e., data scientists use Jupyter Notebook to design and test AI solutions, such as developing recommendation systems, image classification models, or fine-tuning language models on domain-specific data within a single, executable document

The ability to integrate generative AI within the notebooks allows users to generate high-quality code, perform data analysis, and automate tasks using simple text prompts. E.g., Jupyter AI.

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

spaCy provides a structured, object-oriented approach to text processing, enhancing performance.

spaCy processes text into a **Doc** object, where each word and punctuation mark becomes a **Token** object with rich linguistic attributes, allowing advanced operations such as tokenization, part-of-speech tagging, dependency parsing, and named entity recognition.

2. Comparative Analysis

	Scikit-Learn	TensorFlow
Target applications	Designed for classical ML tasks. E.g., classification, regression, clustering, and dimensionality reduction.	Designed for deep learning and neural networks.
Ease of use for beginners	Suitable for small to medium-sized datasets to perform data analysis, preprocessing, and prototyping traditional ML models.	Suitable for complex tasks such as image recognition, natural language processing, and large-scale model training.
Community support	It benefits from its wide adoption in the machine learning community, with extensive documentation and widespread use in education and research.	It has a vibrant community of researchers, practitioners, and developers due to its deep learning capabilities. Presence of abundant tutorials, forums, and third-party resources.

Part 3: Ethics & Optimization

1. Ethical Considerations

Potential Biases in the Model

MNIST Dataset: The dataset mostly contains handwritten digits from Western demographics, which may cause poor performance on digits written by people from different regions or with unique handwriting styles.

Amazon Reviews Dataset: May contain language, gender, or product category biases. For instance, reviews from specific product types or demographics might be overrepresented.

Mitigation Tools and Strategies

TensorFlow Fairness Indicators: Enables evaluation of model performance across subgroups. It helps detect fairness issues by comparing accuracy, precision, and recall for different features such as gender or location.

spaCy's Rule-Based Systems: Allow standardization or removal of biased terms from text data before training models. This helps mitigate language or gender-related bias.

Dataset	Potential Bias	Mitigation Strategy
MNIST	Limited handwriting diversity	Data augmentation with diverse handwriting samples
Amazon Reviews	Gender, language, or product bias	TensorFlow Fairness Indicators; spaCy rule-based preprocess

2. Troubleshooting Challenge – Debugging TensorFlow Script

Common Issues Identified:

- Input shape mismatch: Dense layers require flattened vectors.
- Incorrect loss function: 'mse' used instead of 'sparse_categorical_crossentropy'.
- Missing output activation and dropout for regularization.

Buggy Code

```
import tensorflow as tf

# Load MNIST dataset
(x_train, y_train), (x_test, y_test) = tf.keras.datasets.mnist.load_data()

# Normalization
x_train = x_train / 255.0
x_test = x_test / 255.0

# Incorrect input shape for Dense layer
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(128, activation='relu', input_shape=(28, 28)),
    tf.keras.layers.Dense(10)
])

# Wrong loss for classification
model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)
```

Issues in this Code

1. `input_shape=(28, 28)` is incorrect for Dense layers (should be flattened).
2. Missing a `Flatten()` layer before Dense layers.
3. The loss function 'mse' is inappropriate for classification — should use 'sparse_categorical_crossentropy'.
4. The output layer lacks an activation function (softmax) for multi-class classification.

Fixed Code

```
import tensorflow as tf

# Load and preprocess MNIST data
(x_train, y_train), (x_test, y_test) =
tf.keras.datasets.mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

# Correct model structure
model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),      # Flatten 2D images
    tf.keras.layers.Dense(128, activation='relu'),      # Hidden layer
    tf.keras.layers.Dropout(0.2),                      # Prevent
overfitting
    tf.keras.layers.Dense(10, activation='softmax')    # Output layer for
10 classes
])

# Correct loss and compile settings
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# Train model
model.fit(x_train, y_train, epochs=5, validation_data=(x_test, y_test))
```

Issue	Fix
Input shape mismatch	Added Flatten() layer before Dense layers
Wrong loss function	Changed to 'sparse_categorical_crossentropy'
Missing output activation	Added 'softmax' activation
Overfitting risk	Added Dropout(0.2) layer

Summary of the MNIST Classifier Model

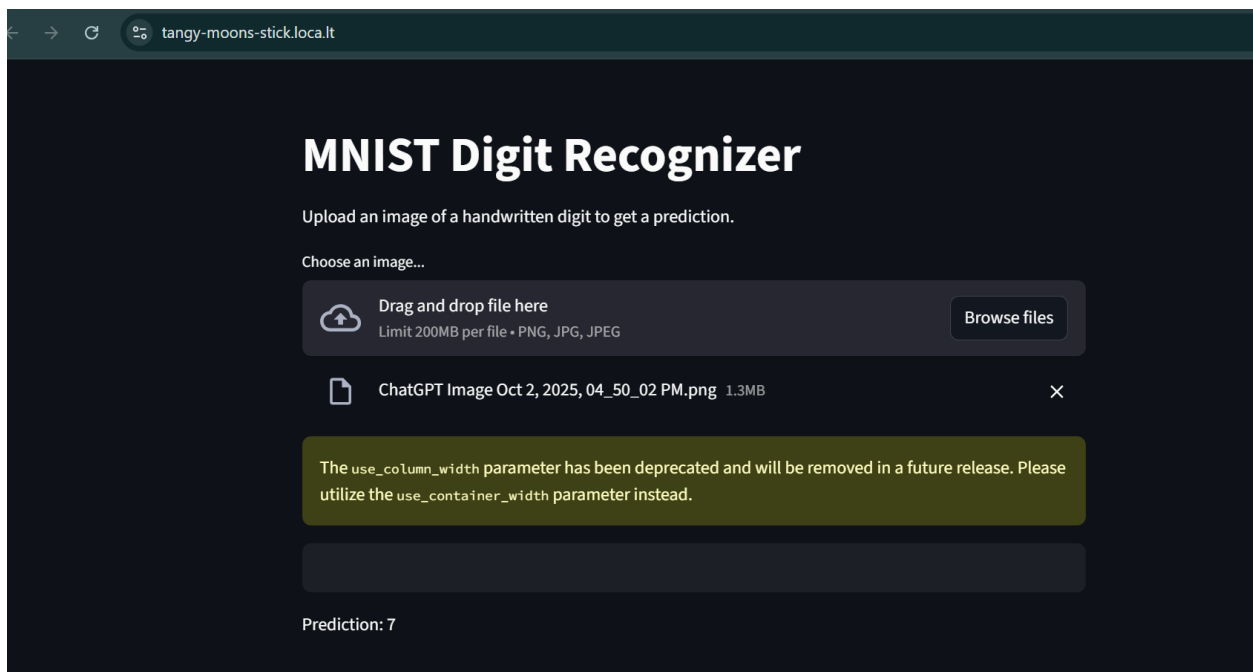
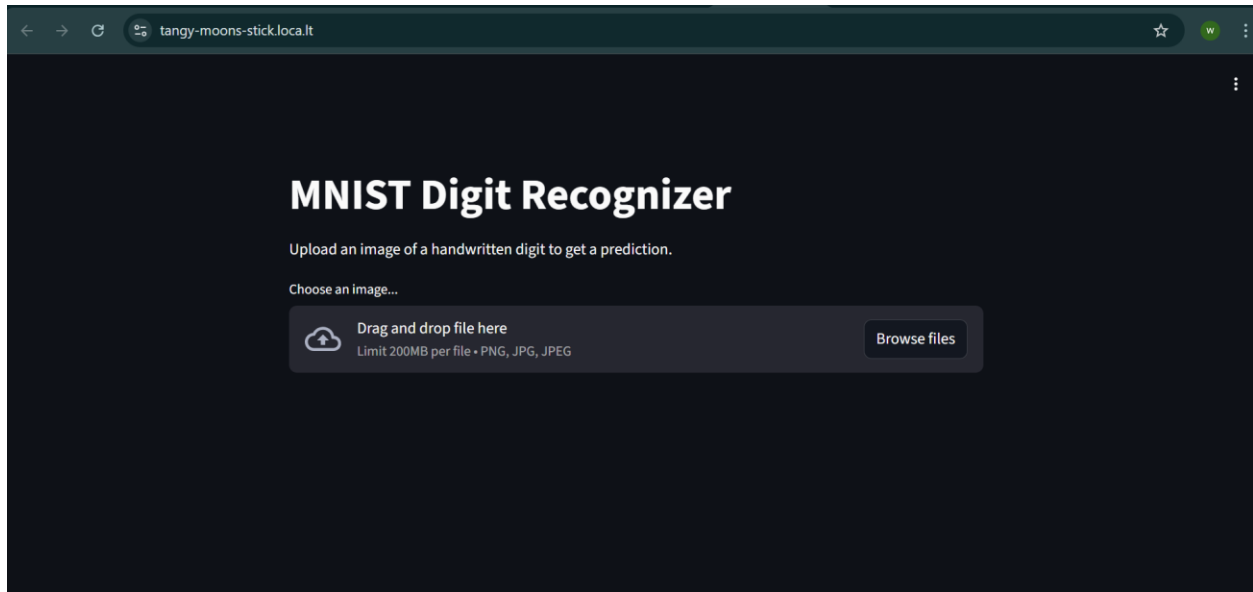
```
In [15]: # Summary of the model
         model.summary()
```

Model: "sequential_2"



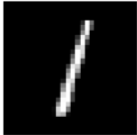


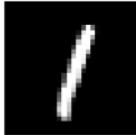























Layer (type)	Output Shape	Param #
conv2d_2 (Conv2D)	(None, 26, 26, 32)	320
max_pooling2d_2 (MaxPooling2D)	(None, 13, 13, 32)	0
conv2d_3 (Conv2D)	(None, 11, 11, 48)	13,872
max_pooling2d_3 (MaxPooling2D)	(None, 5, 5, 48)	0
dropout_1 (Dropout)	(None, 5, 5, 48)	0
flatten_1 (Flatten)	(None, 1200)	0
dense_2 (Dense)	(None, 500)	600,500
dense_3 (Dense)	(None, 10)	5,010

Total params: 619,702 (2.36 MB)
 Trainable params: 619,702 (2.36 MB)
 Non-trainable params: 0 (0.00 B)

Web interface for the MNIST classifier.



Visualize Predictions

<div>Pred: 7, True: 7</div> 	<div>Pred: 2, True: 2</div> 	<div>Pred: 1, True: 1</div> 	<div>Pred: 0, True: 0</div> 	<div>Pred: 4, True: 4</div> 	<div>Pred: 1, True: 1</div> 
<div>Pred: 4, True: 4</div> 	<div>Pred: 9, True: 9</div> 	<div>Pred: 5, True: 5</div> 	<div>Pred: 9, True: 9</div> 	<div>Pred: 0, True: 0</div> 	<div>Pred: 6, True: 6</div> 
<div>Pred: 9, True: 9</div> 	<div>Pred: 0, True: 0</div> 	<div>Pred: 1, True: 1</div> 	<div>Pred: 5, True: 5</div> 	<div>Pred: 9, True: 9</div> 	<div>Pred: 7, True: 7</div> 
<div>Pred: 3, True: 3</div> 	<div>Pred: 4, True: 4</div> 	<div>Pred: 9, True: 9</div> 	<div>Pred: 6, True: 6</div> 	<div>Pred: 6, True: 6</div> 	<div>Pred: 5, True: 5</div> 
<div>Pred: 4, True: 4</div> 	<div>Pred: 0, True: 0</div> 	<div>Pred: 7, True: 7</div> 	<div>Pred: 4, True: 4</div> 	<div>Pred: 0, True: 0</div> 	<div>Pred: 1, True: 1</div> 