Porting a Neural Network from NumPy to TensorFlow

Context

These notes document the issues encountered while rewriting a fully connected neural network from **NumPy** to **TensorFlow (eager execution)**.
Although the mathematical model is unchanged, TensorFlow's execution model, tensor semantics, and API constraints introduce several non-trivial differences.

This file serves as a **personal reference** and learning record.

1. Core Conceptual Difference

NumPy vs TensorFlow

- NumPy operates on mutable arrays and executes eagerly.
- TensorFlow operates on tensors and variables with stricter mutation rules.

Implication:
Code that is mathematically correct in NumPy can fail in TensorFlow due to shape, mutability, or type constraints.

2. Data Representation Errors

X and Y Meaning

- X: input data (features), shape (n_x, m)
- Y: ground-truth labels, shape (1, m)

Common mistake:

```
X = [0, 1, 0, 1]
Y = [0.2, 0.8, 0.25, 0.75]
```

Correct:

```
X = tf.constant([[0], [1], [0], [1]], dtype=tf.float32)
Y = tf.constant([[0.2]], dtype=tf.float32)
```

TensorFlow is strict about **rank and dimensions**.

3. Lists vs Arrays vs Tensors

NumPy Error

AttributeError: 'list' object has no attribute ‘shape'

Cause: Python lists were passed instead of NumPy arrays.

Fix:

```
X = np.array(X)
Y = np.array(Y)
```

TensorFlow equivalent:

```
X = tf.constant(X, dtype=tf.float32)
Y = tf.constant(Y, dtype=tf.float32)
```

4. Tensor Mutability Rules

Illegal Item Assignment

NumPy:

```
p[0, i] = 1
```

TensorFlow:

TypeError: ResourceVariable object does not support item assignment

Reason: Tensors are immutable.

Correct TensorFlow approach:

```
p = tf.cast(probas > 0.5, tf.float32)
```

Use vectorized operations, not loops with assignments.

5. ReLU Backward Masking

NumPy

```
dZ = np.array(dA, copy=True)
dZ[Z <= 0] = 0
```

TensorFlow

```
dZ = dA * tf.cast(Z > 0, dA.dtype)
```

Boolean masking must be expressed arithmetically.

6. Variables vs Tensors in Updates

Error

AttributeError: 'EagerTensor' object has no attribute ‘assign_sub'

Cause: Biases were created as tensors.

Correct:

```
parameters['b' + str(l)] = tf.Variable(
    tf.zeros((layer_dims[l], 1), dtype=tf.float32)
)
```

Only tf.Variable supports parameter updates.

7. Numerical Stability in Cost Function

Log(0) Problem

```
tf.math.log(AL)
```

Fails when AL ≈ 0 or 1.

Fix:

```
epsilon = 1e-7
AL = tf.clip_by_value(AL, epsilon, 1 - epsilon)
```

8. Layer Indexing Consistency

A recurring source of bugs was **off-by-one indexing**.

Correct pattern:

- Parameters: W1 ... WL
- Loop:

for l in range(1, L + 1):

Indexing must be consistent across:

- initialization
- forward pass
- backward pass
- updates

9. Why Loss Converges to ~0.5

With:

- very small datasets
- near-constant labels

Binary cross-entropy converges to predicting the **mean label value**.

This indicates:

- correct implementation
- limited data, not model failure

Summary of Key Lessons

- TensorFlow requires stricter shape discipline than NumPy.
- Mutation must be expressed via TensorFlow ops.
- Boolean logic must be vectorized.
- Mathematical correctness ≠ framework correctness.

Purpose of These Notes

These notes are intended as:

- a personal learning log
- a reference for future medical AI work
- a reminder of framework-level pitfalls when moving from theory to implementation