# Chapter 2: Tensors in Machine Learning

Aymen Negadi

## Introduction to Tensors

A **tensor** is a mathematical function from linear algebra that maps a selection of vectors to a numerical value. The concept originated in physics and was subsequently used in mathematics. Probably the most prominent example that uses the concept of tensors is general relativity.

In the field of machine learning, tensors are used as representations for many applications, such as images or videos. They form the basis for TensorFlow's machine learning framework, which also takes its name from this.

## Tensor Rank and Dimension

If you want to describe tensors more precisely, you need the so-called **rank** and the **dimension**. These can determine the size of the object. For this, we start with a matrix whose rank we want to find out. For this purpose, we form a simple Numpy array, which represents a matrix with three rows and three columns.

Listing 1: Creating a matrix with NumPy

```
import numpy as np
matrix = np.array(
[[1, 5, 7],
 [2, 9, 4],
 [4, 4, 3]])
```

The **rank of a tensor** now provides information about how many indices are needed to reference a single number in the element. It is also often referred to as the number of dimensions.

In the case of a matrix, this means that it has rank 2 (the matrix is two-dimensional) because we need two indices to output a specific number. Suppose we want to change from the object "Matrix" to the number 5 in the first row and second column, then we need two steps to get there.

## Tensor Size and Shape

The size of a tensor gives the length of the axes for each dimension. This means that we can specify the size of our object "matrix" as follows:

Listing 2: Working with TensorFlow tensors

```
import tensorflow as tf
matrix = tf.convert_to_tensor(matrix)
tf.shape(matrix)

# Output: <tf.Tensor: shape=(2,), dtype=int32, numpy=array([3,
    3], dtype=int32)>
```

```python
# Matrix (2-D tensor) - shape: (rows, cols)
matrix = tf.constant([[1, 2], [3, 4]])  # shape: (2, 2)

# 3-D tensor - shape: (depth, rows, cols)
tensor_3d = tf.constant([[[1, 2], [3, 4]], [[5, 6], [7, 8]]])  # shape: (2, 2, 2)
```

### Useful Tensor Methods

Listing 4: Tensor properties and methods

```python
matrix = tf.constant([[1, 2, 3], [4, 5, 6]])

print("Shape:", matrix.shape)           # (2, 3)
print("Rank:", tf.rank(matrix))         # 2 (number of dimensions)
print("Size:", tf.size(matrix))         # 6 (total elements: 2*3)
print("Axis 0 size:", matrix.shape[0]) # 2 (rows)
print("Axis 1 size:", matrix.shape[1]) # 3 (columns)
```

## Types of Tensors

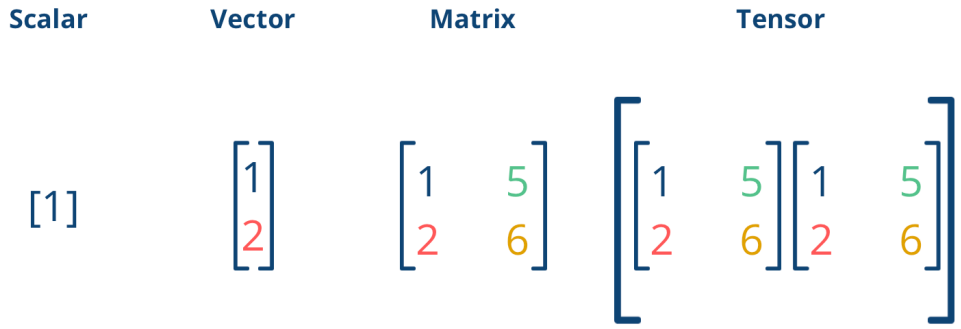


Figure 1: Different Objects in Python Programming — Source: Author

Tensors are the generalization from the objects, which are already known from linear algebra. They are used in programming mainly because they offer the possibility to represent multidimensional data structures.

## Tensor Operations

The allowed arithmetic operations are very similar to those of matrices, but may differ in naming:

1. **Addition**
2. **Subtraction**
3. **Division**

4. **Hadamard Product**: This special product is created by multiplying the objects element by element. The special name comes from the fact that there is still the "normal" way of multiplication, which is based on matrix multiplication

   Example: For tensors $A$ and $B$ of the same shape:

$$(A \odot B)_{ijk} = A_{ijk} \times B_{ijk}$$

# Tensors in Machine Learning

In order to train a machine learning model, a lot of data is needed. However, the data as we know it from the real world is not in the mathematical form in which the model can use it. Therefore, we must first convert images, videos, or text, for example, into a multidimensional data structure so that they can be interpreted mathematically.

At the same time, due to their structure, neural networks can accommodate and output a variety of dimensions that go far beyond conventional vectors and matrices. Therefore, with the proliferation of neural networks, the use of tensors has also become more common.

# Challenges in Tensor Analysis

Tensors are multi-dimensional arrays that are fundamental to many areas of science, including physics, engineering, and machine learning. While tensors provide a powerful way to represent complex data, their analysis can be challenging. Here are some of the challenges in tensor analysis:

1. **High dimensionality**: Tensors can have many dimensions, and as the number of dimensions increases, it becomes more difficult to analyze and visualize the data.
2. **Sparse data**: Tensors can be very sparse, meaning that most of the entries are zero. This makes it difficult to use traditional linear algebra techniques for analysis.
3. **Computational complexity**: Analyzing large tensors can be computationally expensive and time-consuming.
4. **Data preprocessing**: Preprocessing the data to create a tensor can be challenging, particularly when dealing with high-dimensional, sparse, or noisy data.
5. **Interpretation**: Interpreting the results of the tensor analysis can be difficult, particularly when dealing with complex, multidimensional data.

# Sources


- `https://towardsdatascience.com/what-are-tensors-in-machine-learning-5671814646ff`
- `https://e-magnetica.pl/doku.php/scalar_vector_tensor`