**Exercise 2: Cifar10**

**Shenhav Meshulam 313614273**

**Dana Eliyahu 313265332**

**Tom Mendelson 205949746**

### **Part 1: Image Classification:**

### **1st Convolutional Neural Network architecture:**

1. Number of epochs: 30
2. Number of batches: 150
3. Learning rate: 0.005
4. Network architectures:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Layer | Amount of filters | Size of filters | Strides | Activation function | Pooling | Dropout |
| Conv2D | 32 | 3,3 | 1 | relu |  |  |
| Conv2D | 32 | 3,3 | 1 | relu | MaxPooling(2,2) | 0.6 |
| Conv2D | 64 | 3,3 | 1 | relu |  |  |
| Conv2D | 64 | 3,3 | 1 | relu | MaxPooling(2,2) | 0.5 |
| Dense | 512 |  |  | relu |  | 0.4 |
| Dense | 5 |  |  | softmax |  |  |

1. Optimizer: RMSProp
2. Accuracy:
3. Loss:

### **2nd Convolutional Neural Network architecture:**

1. Number of epochs: 35
2. Number of batches: 150
3. Learning rate: 0.0002
4. Network architectures:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Layer | Amount of filters | Size of filters | Strides | Activation function | Pooling | Dropout |
| Conv2D | 32 | 3,3 | 1 | relu |  |  |
| Conv2D | 64 | 3,3 | 1 | relu | MaxPooling(2,2) | 0.4 |
| Conv2D | 64 | 3,3 | 1 | sigmoid |  |  |
| Conv2D | 128 | 3,3 | 1 | relu | MaxPooling(2,2) | 0.4 |
| Dense | 1024 |  |  | relu |  | 0.4 |
| Dense | 5 |  |  | softmax |  |  |

1. Optimizer: Adam
2. Accuracy:
3. Loss:

### **3rd Convolutional Neural Network architecture:**

1. Number of epochs: 40
2. Number of batches: 150
3. Learning rate: 0.0002
4. Network architectures:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Layer | Amount of filters | Size of filters | Strides | Activation function | Pooling | Dropout |
| Conv2D | 32 | 3,3 | 1 | relu |  |  |
| Conv2D | 32 | 3,3 | 1 | relu | AveragePooling(2,2) | 0.4 |
| Conv2D | 64 | 3,3 | 1 | relu |  |  |
| Conv2D | 64 | 3,3 | 1 | relu | AveragePooling(2,2) | 0.4 |
| Conv2D | 128 | 3,3 | 1 | relu |  |  |
| Dense | 1024 |  |  | relu |  | 0.4 |
| Dense | 5 |  |  | softmax |  |  |

1. Optimizer: RMSProp
2. Accuracy:
3. Loss:

### **Part 2: Recognizing a flower**

We used our last option of a model and removed the fully connected layers.

The last 2 layers will look like this:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Layer | Amount of filters | Size of filters | Strides | Activation function | Pooling | Dropout |
| Dense | 1024 |  |  | relu |  | 0.4 |
| Dense | 1 |  |  | Sigmoid |  |  |

In this case, the problem is a binary, so we use sigmoid instead of softmax.