```
## Load libraries
import numpy as np
import sys
import matplotlib.pyplot as plt
import matplotlib.cm as cm
plt.style.use('dark_background')
%matplotlib inline
```

```
pip install tensorflow
```

```
Collecting tensorflow
  Obtaining dependency information for tensorflow from https://files.pythonhosted.org/packages/8c/ca/c3f0114272924e690c34fd2698dcd877ab2d881fe4196cc461a98d2b4693/tensorflow-2.15.0-cp39-cp39-win_amd64.whl.metadata
  Downloading tensorflow-2.15.0-cp39-cp39-win_amd64.whl.metadata (3.6 kB)
Collecting tensorflow-intel==2.15.0 (from tensorflow)
  Obtaining dependency information for tensorflow-intel==2.15.0 from https://files.pythonhosted.org/packages/1b/86/cda55138387a291fb684691d159742e6196daad571f01de60f809e266d40/tensorflow_intel-2.15.0-cp39-cp39-win_amd64.whl.metadata
  Downloading tensorflow_intel-2.15.0-cp39-cp39-win_amd64.whl.metadata (5.1 kB)
Collecting absl-py>=1.0.0 (from tensorflow-intel==2.15.0->tensorflow)
  Obtaining dependency information for absl-py>=1.0.0 from https://files.pythonhosted.org/packages/01/e4/dc0a1dcc4e74e08d7abedab278c795eef54a224363bb18f5692f416d834f/absl_py-2.0.0-py3-none-any.whl.metadata
  Downloading absl_py-2.0.0-py3-none-any.whl.metadata (2.3 kB)
Collecting astunparse>=1.6.0 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading astunparse-1.6.3-py2.py3-none-any.whl (12 kB)
Collecting flatbuffers>=23.5.26 (from tensorflow-intel==2.15.0->tensorflow)
  Obtaining dependency information for flatbuffers>=23.5.26 from https://files.pythonhosted.org/packages/6f/12/d5c79ee252793ffe845d58a913197bfa02ae9a0b5c9bc3dc4b58d477b9e7/flatbuffers-23.5.26-py2.py3-none-any.whl.metadata
  Downloading flatbuffers-23.5.26-py2.py3-none-any.whl.metadata (850 bytes)
Collecting gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading gast-0.5.4-py3-none-any.whl (19 kB)
Collecting google-pasta>=0.1.1 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading google_pasta-0.2.0-py3-none-any.whl (57 kB)
     ---------------------------------------- 0.0/57.5 kB ? eta -:--:--
     -------------------- ------------------- 30.7/57.5 kB 1.4 MB/s eta 0:00:01
     ---------------------------------------- 57.5/57.5 kB 1.0 MB/s eta 0:00:00
Collecting h5py>=2.9.0 (from tensorflow-intel==2.15.0->tensorflow)
  Obtaining dependency information for h5py>=2.9.0 from https://files.pythonhosted.org/packages/98/e0/679ec9293f69e28406215d2209a949c7e615d459fdee90103e8bb0d75c41/h5py-3.10.0-cp39-cp39-win_amd64.whl.metadata
  Downloading h5py-3.10.0-cp39-cp39-win_amd64.whl.metadata (2.5 kB)
Collecting libclang>=13.0.0 (from tensorflow-intel==2.15.0->tensorflow)
  Obtaining dependency information for libclang>=13.0.0 from https://files.pythonhosted.org/packages/02/8c/dc970bc00867fe290e8c8a7befa1635af716a9ebdfe3fb9dce0ca4b522ce/libclang-16.0.6-py2.py3-none-win_amd64.whl.metadata
  Downloading libclang-16.0.6-py2.py3-none-win_amd64.whl.metadata (5.3 kB)
Collecting ml-dtypes~=0.2.0 (from tensorflow-intel==2.15.0->tensorflow)
  Obtaining dependency information for ml-dtypes~=0.2.0 from https://files.pythonhosted.org/packages/c7/47/54b1e5eea9ed7f8a5f701713e47ea45e798a4f3e5f476a053fd0b537e2af/ml_dtypes-0.2.0-cp39-cp39-win_amd64.whl.metadata
  Downloading ml_dtypes-0.2.0-cp39-cp39-win_amd64.whl.metadata (20 kB)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in c:\users\sa ravi\anaconda3\envs\aimlsem1\lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (1.23.5)
Collecting opt-einsum>=2.3.2 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading opt_einsum-3.3.0-py3-none-any.whl (65 kB)
     ---------------------------------------- 0.0/65.5 kB ? eta -:--:--
     ---------------------------------------- 65.5/65.5 kB 3.5 MB/s eta 0:00:00
Requirement already satisfied: packaging in c:\users\sa ravi\anaconda3\envs\aimlsem1\lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (23.1)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in c:\users\sa ravi\anaconda3\envs\aimlsem1\lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (4.24.0)
Requirement already satisfied: setuptools in c:\users\sa ravi\anaconda3\envs\aimlsem1\lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (68.0.0)
Requirement already satisfied: six>=1.12.0 in c:\users\sa ravi\anaconda3\envs\aimlsem1\lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (1.16.0)
Collecting termcolor>=1.1.0 (from tensorflow-intel==2.15.0->tensorflow)
  Obtaining dependency information for termcolor>=1.1.0 from https://files.pythonhosted.org/packages/d9/5f/8c716e47b3a50cbd7c146f45881e11d9414def768b7cd9c5e6650ec2a80a/termcolor-2.4.0-py3-none-any.whl.metadata
  Downloading termcolor-2.4.0-py3-none-any.whl.metadata (6.1 kB)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\sa ravi\anaconda3\envs\aimlsem1\lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (4.7.1)
Collecting wrapt<1.15,>=1.11.0 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading wrapt-1.14.1-cp39-cp39-win_amd64.whl (35 kB)
Collecting tensorflow-io-gcs-filesystem>=0.23.1 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading tensorflow_io_gcs_filesystem-0.31.0-cp39-cp39-win_amd64.whl (1.5 MB)
     ---------------------------------------- 0.0/1.5 MB ? eta -:--:--
     --- ------------------------------------ 0.1/1.5 MB 7.0 MB/s eta 0:00:01
     --- ------------------------------------ 0.1/1.5 MB 2.1 MB/s eta 0:00:01
     ------ --------------------------------- 0.2/1.5 MB 2.0 MB/s eta 0:00:01
     -------- ------------------------------- 0.3/1.5 MB 2.2 MB/s eta 0:00:01
     ---------- ----------------------------- 0.4/1.5 MB 2.1 MB/s eta 0:00:01
     ------------ --------------------------- 0.5/1.5 MB 2.2 MB/s eta 0:00:01
     --------------- ------------------------ 0.6/1.5 MB 2.1 MB/s eta 0:00:01
     ----------------- ---------------------- 0.6/1.5 MB 1.9 MB/s eta 0:00:01
     ------------------- -------------------- 0.7/1.5 MB 1.9 MB/s eta 0:00:01
```

```
import tensorflow as tf
```

```
WARNING:tensorflow:From c:\Users\SA RAVI\anaconda3\envs\aimlsem1\lib\site-packages\keras\src\losses.py:2976: The name tf.losses.sparse_softmax_cross_entropy is deprecated. Please use tf.compat.v1.losses.sparse_softmax_cross_entropy instead.
```

```
tf.__version__
```

```
'2.15.0'
```

```
# Generate artificial data with 5 samples, 4 features per sample
# and 3 output classes
num_samples = 5 # number of samples
num_features = 4 # number of features (a.k.a. dimensionality)
num_labels = 3 # number of output labels
# Data matrix (each column = single sample)
X = np.random.choice(np.arange(3, 10), size = (num_features, num_samples), replace = True)
# Class labels
y = np.random.choice([0, 1, 2], size = num_samples, replace = True)
print(X)
print('------')
print(y)
print('------')
# One-hot encode class labels
y = tf.keras.utils.to_categorical(y)
print(y)
```

```
[[6 4 7 7 7]
 [4 3 7 8 5]
 [5 8 4 6 5]
 [9 6 9 5 8]]
------
[1 2 0 2 1]
------
[[0. 1. 0.]
 [0. 0. 1.]
 [1. 0. 0.]
 [0. 0. 1.]
 [0. 1. 0.]]
```

A generic layer class with forward and backward methods

```
class Layer:
  def __init__(self):
    self.input = None
    self.output = None

  def forward(self, input):
    pass

  def backward(self, output_gradient, learning_rate):
    pass
```

The softmax classifier steps for a generic sample $\mathbf{x}$ with (one-hot encoded) true label $\mathbf{y}$ (3 possible categories) using a randomly initialized weights matrix (with bias abosrbed as its last last column):

1. Calculate raw scores vector for a generic sample $\mathbf{x}$ (bias feature added):
$$\mathbf{z} = \mathbf{W}\mathbf{x}.$$
2. Calculate softmax probabilities (that is, softmax-activate the raw scores)
$$\mathbf{a} = \text{softmax}(\mathbf{z}) \Rightarrow \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \text{softmax}\left(\begin{bmatrix} z_0 \\ z_1 \\ z_2 \end{bmatrix}\right) = \begin{bmatrix} \frac{e^{z_0}}{e^{z_0}+e^{z_1}+e^{z_2}} \\ \frac{e^{z_1}}{e^{z_0}+e^{z_1}+e^{z_2}} \\ \frac{e^{z_2}}{e^{z_0}+e^{z_1}+e^{z_2}} \end{bmatrix}$$
3. Softmax loss for this sample is (where output label y is not yet one-hot encoded)
$$\begin{aligned} L &= -\log([a]_y) \\ &= -\log\Big([\text{softmax}(\mathbf{z})]_y\Big) \\ &= -\log\Big([\text{softmax}(\mathbf{W}\mathbf{x})]_y\Big). \end{aligned}$$
4. Predicted probability vector that the sample belongs to each one of the output categories is given a new name
$$\hat{\mathbf{y}} = \mathbf{a}.$$
5. One-hot encoding the output label
$$\underbrace{y \rightarrow \mathbf{y}}_{\text{e.g. } 2 \rightarrow \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}}$$
results in the following representation for the softmax loss for the sample which is also referred to as the categorical crossentropy (CCE) loss:
$$L = L\left(\mathbf{y}, \hat{\mathbf{y}}\right) = \sum_{k=0}^{2} -y_k \log(\hat{y}_k)\,.$$

6. Calculate the gradient of the loss for the sample w.r.t. weights by following the computation graph from top to bottom (that is, backward):

$$
\begin{array}{c}
L \\
\downarrow \\
\hat{\mathbf{y}} = \mathbf{a} \\
\downarrow \\
\mathbf{z} \\
\downarrow \\
\mathbf{W}
\end{array}
$$

$$
\begin{aligned}
\Rightarrow \nabla_{\mathbf{W}}(L) &= \nabla_{\mathbf{W}}(\mathbf{z}) \times \nabla_{\mathbf{z}}(\mathbf{a}) \times \nabla_{\mathbf{a}}(L) \\
&= \underbrace{\nabla_{\mathbf{W}}(\mathbf{z})}_{\text{first term}} \times \underbrace{\nabla_{\mathbf{z}}(\mathbf{a})}_{\text{second to last term}} \times \underbrace{\nabla_{\hat{\mathbf{y}}}(L)}_{\text{last term}}.
\end{aligned}
$$

7. Now focus on the last term $\nabla_{\hat{\mathbf{y}}}(L)$:

$$
\nabla_{\hat{\mathbf{y}}}(L) = \begin{bmatrix} \nabla_{\hat{y}_0}(L) \\ \nabla_{\hat{y}_1}(L) \\ \nabla_{\hat{y}_2}(L) \end{bmatrix} = \begin{bmatrix} -y_0/\hat{y}_0 \\ -y_1/\hat{y}_2 \\ -y_0/\hat{y}_2. \end{bmatrix}
$$

8. Now focus on the second to last term $\nabla_{\mathbf{z}}(\mathbf{a})$:

$$
\begin{aligned}
\nabla_{\mathbf{z}}(\mathbf{a}) &= \nabla_{\mathbf{z}} \left(\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} \right) \\
&= [\, \nabla_{\mathbf{z}}(a_0) \quad \nabla_{\mathbf{z}}(a_1) \quad \nabla_{\mathbf{z}}(a_2) \,] \\
&= \begin{bmatrix} \nabla_{z_0}(a_0) & \nabla_{z_0}(a_1) & \nabla_{z_0}(a_2) \\ \nabla_{z_1}(a_0) & \nabla_{z_1}(a_1) & \nabla_{z_1}(a_2) \\ \nabla_{z_2}(a_0) & \nabla_{z_2}(a_1) & \nabla_{z_2}(a_2) \end{bmatrix} \\
&= \begin{bmatrix} a_0(1-a_0) & -a_1 a_0 & -a_2 a_0 \\ -a_0 a_1 & a_1(1-a_1) & -a_1 a_1 \\ -a_0 a_2 & -a_1 a_2 & a_2(1-a_2) \end{bmatrix}.
\end{aligned}
$$

9. On Monday, we will focus on the first term to complete the gradient calculation using the computation graph.

```
## Softmax activation class
class Softmax(Layer):
  def forward(self, input):
    self.output = np.array(tf.nn.softmax(input))

  def backward(self, output_gradient, learning_rate):
    return(np.dot((np.identity(np.size(self.output))-self.output.T) * self.output, output_gradient))


## Define the loss function and its gradient
def cce(y, yhat):
  return(-np.sum(y*np.log(yhat)))

def cce_gradient(y, yhat):
  return(-y/yhat)

# TensorFlow in-built function for categorical crossentropy loss
#cce = tf.keras.losses.CategoricalCrossentropy()


# Step-1: add the bias feature to all the samples
X_bias = np.vstack((X, np.ones(X.shape[1])))
X_bias

     array([[6., 4., 7., 7., 7.],
            [4., 3., 7., 8., 5.],
            [5., 8., 4., 6., 5.],
            [9., 6., 9., 5., 8.],
            [1., 1., 1., 1., 1.]])


# Step-2: initialize the entries of the weights matrix randomly
W = np.random.rand(num_labels, num_features + 1)



# Step-3: create softmax layer object softmax
softmax = Softmax()
```

```
# Step-4: run over each sample
for i in range(X.shape[1]):
  z = np.dot(W, X_bias[:, i])
  softmax.forward(z)
  cce_loss = cce(y[i, :], softmax.output)
  print(f"Sample {i + 1}, Cross-Entropy Loss: {cce_loss}  and Gradient: {cce_gradient(y[i, :], softmax.output)}")
```

```
Sample 1, Cross-Entropy Loss: 7.033108378769083  and Gradient: [    -0.         -1133.54863819    -0.        ]
Sample 2, Cross-Entropy Loss: 10.28275450582269  and Gradient: [    -0.             -0.         -29224.26118621]
Sample 3, Cross-Entropy Loss: 0.0028763987446593026  and Gradient: [-1.00288054 -0.         -0.         ]
Sample 4, Cross-Entropy Loss: 10.3623275346211  and Gradient: [    -0.             -0.         -31644.74992358]
Sample 5, Cross-Entropy Loss: 5.549089202804743  and Gradient: [  -0.         -257.00337132   -0.        ]
```

```
# (d) Print gradient
gradient = cce_gradient(y[i, :], softmax.output)
print("Gradient:", gradient)
```

```
Gradient: [  -0.         -257.00337132   -0.        ]
```