Keras

A deep learning library

What is keras?

- Keras is a high-level neural networks API, written in Python.
- Built on top of either Theano or TensorFlow.
- Most powerful & easy to use for developing and evaluating deep learning models.

Why use Keras?

- Allows for easy and fast prototyping (through user friendliness, modularity, and extensibility).
- Supports both convolutional networks and recurrent networks, as well as combinations of the two.
- Runs seamlessly on CPU and GPU.

Creating a keras model

- **Architecture Definition**:-no of layers,no of nodes in layers,activation function to be used.
- **Compile**:-defines the loss function and some details about how optimization works.
- **Fit**:-cycle of backpropagation and optimization of model weights with your data.
- **Predict**:-to predict the model prepared.

Keras code for creating model

The **sequential** model used is a linear stack of layers.

```
### Model begins ###
model = Sequential()
model.add(Convolution2D(16, 5, 5, activation='relu', input_shape=(img_width, img_height, 3)))
model.add(MaxPooling2D(2, 2))

model.add(Convolution2D(32, 5, 5, activation='relu'))
model.add(MaxPooling2D(2, 2))

model.add(Flatten())
model.add(Dense(1000, activation='relu'))

model.add(Dense(10, activation='softmax'))
###  Model Ends ###
```

Compile model

```
# Compile model

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

- The loss function to use to evaluate a set of weights.
- The optimizer used to search through different weights for the network and any optional metrics we would like to collect and report during training.
- For classification problem you will want to set this to metrics=[‘accuracy’]

Fit the model

```
# fit model
model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=epochs,
          verbose=1,
          validation_data=(x_test, y_test))
```

- Execution of model for some data.
- Train data and iterate data in batches.

Evaluate Model

```
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

- It give us an idea of how well we have modeled the dataset.

Predict

```
classes=model.predict(x_test,batch_size=128)
```

- It generates prediction on new data

Thank you