TensorFlow Workshop

Agenda

- **Introduction**
  - What is TensorFlow and how is it used in Machine Learning?
  - Applications of TensorFlow
  - Google Colab
  - Neural Networks
- **Linear Regression**
  - Dense Layers
  - Training and Testing
  - Loss
  - Hands-On(Celsius to Fahrenheit)
- **Classification**
  - Convolutional Neural Network(CNN)
  - Hands-On(Text Classification with IMDB or Translation)

Introduction

Tensorflow

"TensorFlow is an open-source software library for numerical computation using data flow graphs".

But actually, TensorFlow is a suite of software, an ecosystem for developing deep learning models. It contains all the tools right from building to deployment. TensorFlow has 3 main components

- TensorFlow (API)
- TensorBoard
- TensorFlow Serving.

TensorFlow (API)

This component of TensorFlow contains the API's to define the models and train the models with the data. The actual computation was written in C++ though it is accessed with python API's. The advantages are of 2-fold, first, we get the more user-friendly python interface to develop the models and second, we can run the models on fast and efficient compiled C++ code.

TensorBoard

The third component of the ecosystem and the boon for engineers is the TensorBoard. It helps to analyze, visualize, and debug TensorFlow graphs. Using TensorFlow to its fullest means using all three components in conjunction with one other.

TensorFlow Serving

This component of TensorFlow helps to deploy the pre-trained models. TensorFlow Serving is capable of switching from old models to new models without any downtime. This is the stand out feature in the ecosystem. This is also written in C++ and can be accessible with python interfaces.

Features of TensorFlow

- Opensource: Licensed under Apache 2.0
- Heterogeneity: It works on computers of all shapes and sizes right from servers -desktop-mobile in real-time.
- Distributed.
- Support: It is supported by the internet giant Google itself.

Why TensorFlow?

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML-powered applications.

Applications of Tensorflow

- Text-Based Applications
- Image Recognition
- Voice/Sound Recognition
- Time Series
- Video Detection

Types of ML Problems

There are several subclasses of ML problems based on what the prediction task looks like. These are the common supervised and unsupervised ML problems.

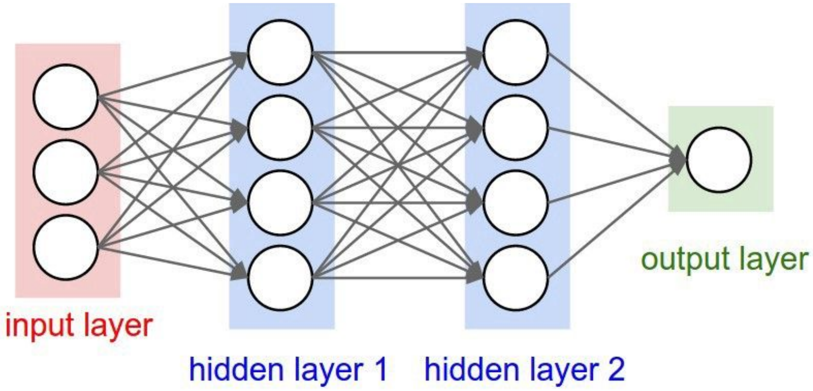
- Classification
- Regression
- Clustering
- Association rule learning
- Structured output
- Ranking

Artificial Neural Networks

A neural network is exactly what it says in the name. It is a network of neurons that are used to process information. To create these, scientists looked at the most advanced data processing machine at the time — the brain. Our brains process information using networks of neurons. They receive input, process it, and accordingly output electric signals to the neurons it is connected to. Using biomimicry, we were able to apply the architecture of our brains to further the field of artificial intelligence. Artificial neural networks recreate the structure of human neurons to process information resulting in much more accurate results than previously used regression models.

Layers

A neural network is made up of 3 main parts:



Input layer

This is the layer that inputs information for the neural network to process. Each circle represents 1 feature (a piece of information). This can be anything. It could be the square footage of your house for a house price prediction program, or the value of a pixel on a screen for a computer vision program

Hidden layers

These layers do all the processing for neural networks. You can have as many of these as you want. Generally speaking, the more hidden layers you have, the more accurate the neural network will be. Each layer consists of nodes that mimic our brains' neurons. These nodes receive information from the previous layer's nodes, multiply it by weight and then add a bias to it. Each line in the diagram represents a weight. That may sound confusing so here's an analogy:

Let's say I want to predict my SAT score and I have gathered two pieces of data: the number of hours I prepared, and the breakfast I had on the day of the test. These would go into the input layer. It's quite obvious that the hours I spend preparing will have a much greater impact on my mark than my breakfast (or everyone would ace the SAT). Because of this, the hours I spend studying receive a significantly larger weight value to be multiplied by. Now, granted, there's a chance that your breakfast may vary slightly impact your score, so it will receive a much lower weight value.

Output Layer

This layer simply brings together the information from the last hidden layer of the network to output all the information you need from the program.

To sum up, neural networks take information from the input layer, process it in the hidden layers, and output the desired information in the output layer. This whole process of running a neural network is called forward propagation.

We'll be looking at two most commonly used ML problems

- Linear Regression
- Classification

Linear Regression

Linear regression is used for finding the linear relationship between the target and one or more predictors. There are two types of linear regression - Simple and Multiple.

Linear regression is a basic and commonly used type of predictive analysis. The overall idea of regression is to examine two things:

1. Does a set of predictor variables do a good job in predicting an outcome (dependent) variable?
2. Which variables, in particular, are significant predictors of the outcome variable, and in what way do they–indicated by the magnitude and sign of the beta estimates–impact the outcome variable?

These regression estimates are used to explain the relationship between one dependent variable and one or more independent variables. The simplest form of the regression equation with one dependent and one independent variable is defined by the formula **y = c + b*x**, where y = estimated dependent variable score, c = constant, b = regression coefficient, and x = score on the independent variable.

Training and Testing

Training

A training dataset is a dataset of examples used for learning, that is to fit the parameters (e.g., weights) of, for example, a classifier. Most approaches that search through training data for empirical relationships tend to overfit the data, meaning that they can identify and exploit apparent relationships in the training data that do not hold in general.

Testing

A test dataset is a dataset that is independent of the training dataset, but that follows the same probability distribution as the training dataset. If a model fit to the training dataset also fits the test dataset well, minimal overfitting has taken place. A better fitting of the training dataset, as opposed to the test dataset, usually points to overfitting.

Make sure that your test set meets the following two conditions:

- It is large enough to yield statistically meaningful results.
- Is representative of the data set as a whole. In other words, don't pick a test set with different characteristics than the training set.

Loss

Loss is the penalty for a bad prediction. That is, the loss is a number indicating how bad the model's prediction was on a single example. If the model's prediction is perfect, the loss is zero;

otherwise, the loss is greater. The goal of training a model is to find a set of weights and biases that have low loss, on average, across all examples.

Squared loss: a popular loss function

The linear regression models we'll examine here use a loss function called squared loss (also known as L_2 loss). The squared loss for a single example is as follows:

$$= (\text{observation - prediction}(x))^2$$

$$= (y - y')^2$$

Mean square error

Mean square error (MSE) is the average squared loss per example over the whole dataset. To calculate MSE, sum up all the squared losses for individual examples and then divide by the number of examples.

Hands-On(Celsius to Fahrenheit)

In this Hands-On the problem we will be converting Celsius to Fahrenheit using the concept of Linear Regression, where the approximate formula is:

$$\mathbf{f=c\times1.8+32}$$

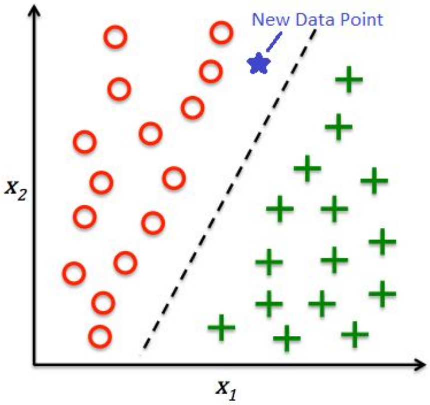
Of course, it would be simple enough to create a conventional Python function that directly performs this calculation, but that wouldn't be machine learning.

Instead, we will give TensorFlow some sample Celsius values and their corresponding Fahrenheit values. Then, we will train a model that figures out the above formula through the training process.

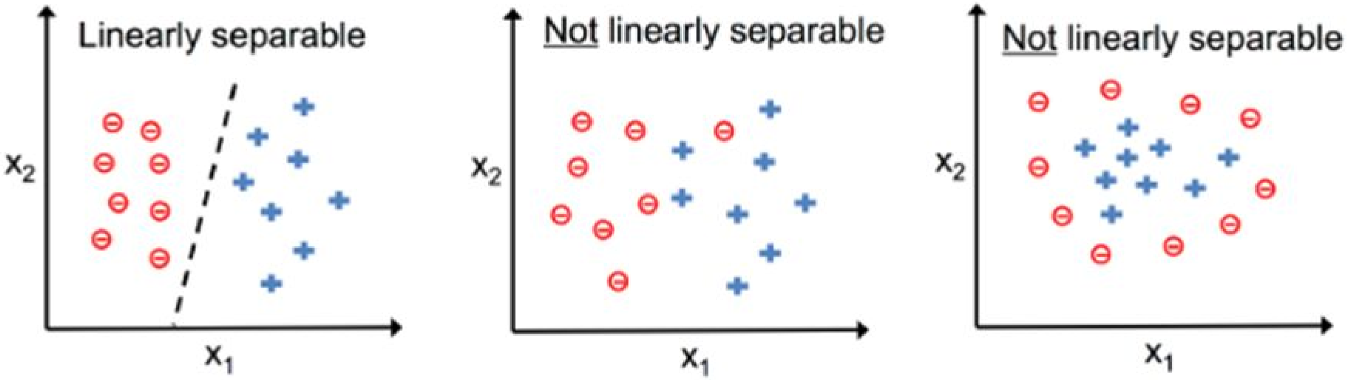
Classification

Classification is used to categorize different objects. It is a supervised problem in machine learning (just like regression) where we have labelled dataset. A classification model attempts to draw some conclusions from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes.
There are two main types of classification problems:

- **Binary classification:** The typical example is email spam detection, which each email is spam → 1 spam; or isn't → 0, or a medical test, where if the results are positive → 1, else → 0



- **Multi-class classification:** Like handwritten character recognition (where classes go from 0 to 9).



Convolutional Neural Network(CNN)

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analyzing visual imagery. CNNs are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "fully-connectedness" of these networks makes them prone to overfitting data. Typical ways of regularization include adding some form of magnitude measurement of weights to the loss function. However, CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns. Therefore, on the scale of connectedness and complexity, CNNs are on the lower extremity.

Hands-On(Text Classification with IMDB or Translation)

In this Hands-On we'll be writing a program that classifies movie reviews as *positive* or *negative* using the text of the review. This is an example of binary or two-class classification, an important and widely applicable kind of machine learning problem.

We'll be using the IMDB dataset that contains the text of 50,000 movie reviews from the Internet Movie Database. These are split into 25,000 reviews for training and 25,000 reviews for testing. The training and testing sets are *balanced*, meaning they contain an equal number of positive and negative reviews.

Sources

Tensorflow - https://www.tensorflow.org
What is ML? -
https://medium.com/datadriveninvestor/what-is-machine-learning-55028d8bdd53
Reinforcement Learning - https://www.geeksforgeeks.org/what-is-reinforcement-learning/
Tensorflow Usecases -
https://www.exastax.com/deep-learning/top-five-use-cases-of-tensorflow/
Case Studies - https://www.tensorflow.org/about/case-studies
Towards Data Science - https://towardsdatascience.com

Courses

Machine Learning Crash Course | Google
TensorFlow
TensorFlow | Udacity

--