CSC 180 – Quiz 1 Study Guide

Lecture #1: Introduction:

- **What is AI:** AI is the study on how to program computers to solve hard problems that traditionally require human intelligence to solve.
- **AI Goal:** Computer that think and act humanly/rationally.
- **AI History:**
    - **1980:**
        - Expert systems (many rules to solve complex problems, hard to develop efficient and comprehensive rules).
        - Neural networks, backpropagation.
        - Uncertain/Probabilistic knowledge reasoning.
    - **1990:**
        - Machine learning becomes dominant.
    - **Late 2000s:**
        - Big data comes to the stage.
    - **2010s:**
        - Deep learning starts to flourish everywhere causing a new industry boom.
- **AI Applications:** Computer vision (self-driving cars, medical diagnosis, image tagging). Speech and natural language. Games (Chess, Go, etc.). Robots.
- **Agent:** A software that observes input through sensors and acts upon an environment using actuators.
- **Smart Agent:** Given input from a sensor, takes an action that will maximize its performance measure.
- **PEAS Model** (Examples of the model in Lecture #1 Slides)**:**
    - **P**erformance Measure: An objective function the agent is maximizing.
    - **E**nvironment States: A formal representation for states (a group of variables).
    - **A**ctuators: Actions that change the agent state according to a transition model.
    - **S**ensors: Observations that allow the agent to infer the world state.

Lecture #2: Data Preprocessing and Model Evaluation:

- **What is Data:** Data includes objects (an entity in the world sometimes called a record, point, sample, or instance, row) and their attributes (a property of an object, sometimes called a field or a feature, column).
- **Attribute Types:**
    - **Numerical:** Can be discreate or continues. Examples include temperature, dates, time, length, etc.
    - **Categorical:** Can be nominal (no order) or ordinal (ordered but not comparable). Examples include eye color, rankings, height (tall, medium, short), etc.
- **Label Encoding:** Assigning one unique number to each distinct value (*calabar* → 0, *uyo* → 1, etc). It is a 1to1 mapping from a string to an integer. NEVER TO BE USED ON INPUT FEATURES.
- **One Hot Encoding (OHE):** For each distinct value, create a new binary feature. (Red → 001, Green → 010, Blue → 100). Useful for encoding categorical input features.
- **Data Normalization:** Numerical input features need to be normalized before being sent to a model. This can be done the following ways.
    - **1:** Divide row values by max value, thus bringing all values between 0 and 1.
    - **2:** Divide by the difference between the max and min value, bringing all values between 0 and 1.
    - **3:** Use the Z-Score of a raw value X (most widely used)
- **Bag of Words Model:** Used for natural language processing.
    - **1.** Create a feature attribute for each unique word.
    - **2.** Define each document (vector) over all possible words with the value for the feature being the count (number of occurrences) of that particular word in that document.
- **TF-IDF:** Used for natural language processing. A measure of importance and uniqueness, scoring each word.
    - **1:** Remove puncuations, lowercase everything, clear white spaces, breach each business into the most popular words.
    - **2:** Remove stop words (not so interesting words like “the”, “to”, “get” and so forth).

    - **3:** Do TF (Term frequency analysis) multiplied by IDF (Inverse document frequency) to measure importance and uniqueness of a word.
        - $W_{x,y} = tf_{x,y} * log(\frac{N}{df_x})$
- **Machine Learning:** The study on getting a computer to finish a task without explicitly programming it.
- **Unsupervised Learning** is where the data has no labels with **Supervised Learning** being where the data has labels.
- **Classification Model Steps:**
    - Identify input features, output labels, decide what type of model to train, train model on training data and test on test data.
- **Underfitting** occurs when a model is too simple and has large errors on the training and test set. **Overfitting** is where a model fails to generalize, the details of the training set being learned, and failing on the test set. Shown by high accuracy on training data with low accuracy on test data.
- **Classification Metrics:**
    - **Precision:** How accurate? (#correction predictions/#in dataset)
    - **Recall:** How many are labeled correctly
    - **F1-Score:** Combine both precision and recall in a single summary.
    - **ROC Curve:** Plots true positive rates against false positive rates, with the area under the curve being the best model.
- **Regression Metrics:**
    - **RMSE (Root Mean Squared Error)**
    - **R2 Score**

Lecture #3: Neural Networks

- **Neural Network:** A collection of neurons
    - **Pros:**
        - Flexible and general functional approximation framework
        - Can build extremely powerful models by adding more layers.
    - **Cons:**
        - Training is prone to local optimum.
        - Huge amount of training data, and computing power required.
        - Implementation choices are huge (hyperparameter options).
- **Neurons:** A neuron is a weighted sum of all its inputs, followed by an activation function.
- **SoftMax:** Converts any arbitrary vector into a probabilistic distribution.
- **Training Multilayer Neural Networks:** The process of training a neural network involves finding the weights that minimize the error between "true" and predicted labels in the training records.
- **Gradient Descent (back propagation):** Weights are updated according to the rules of gradient descent and computed from output to input layers.
    - **How often are training weights updated:** When a model is trained, you specify how many samples you want to use per update. This is called the batch size.
    - **How are weights updated:** According to the update rules or **optimizers**
- If a model's weights are updated on each **step** (iteration), one patch is processed. If at each **epoch**, weights are updated after a pass through the entire dataset.
- A **high learning rate** runs the risk of passing over the optimum while a **low learning rate** takes forever to find the optimum.

Lecture #4: Deep Learning

- **Regular Neural Networks:**
    - Generally, have 1 to 2 hidden layers and are only used in supervised learning.
    - Dense fully connected neural network (every neuron in adjacent layers are connected).
- **Deep Neural Networks:**
    - Generally, have more hidden layers, and can be used for both supervised and unsupervised learning.
    - Includes layers such as convolutional, pooling, and dense.

- **Deep learning** is better because its greater (and more diverse) hidden layers can detect true underlying features and train to detect those features. Stacked hidden layers capture features from a lower to higher level.
- A **Deep Learning** neural network is trained layer by layer. With each of the non-output layers being trained to be an auto-encoder
  - **Auto-Encoder:** A special neural network with the output layer having the same number of neurons as the input layer. With the purpose to reconstruct the output.
- An **Auto-Encoder learns true features** by compressing input data to reveal true features. Thus, forcing the middle layer to become good feature detectors for input. This can also be referred to as **dimensionality reduction.**
- **Recurrent Neural Networks:**
  - Data with temporal or sequential structures.
- **Convolutional Neural Networks:**
  - For scenarios where "spatially closer" data is more correlated.
  - Hidden neurons connected to a local receptive field.
- **CNN Stages:**
  - **Convolution (Single or Multiple Kernels):**
    - Produces a weighted sum (compressed) image.
    - Uses a kernel of defined size, scans the image a produces a feature map.
    - Number of features maps equal to number of kernels.
    - Outputs a feature map, based on the weighted sum of the scanned image.
  - **Nonlinear Transformation:**
    - Same as an activation function in Neural Networks
  - **Polling:**
    - Max Pooling reports the maximum output within a rectangular region.
    - Average Pooling reports the average output within a rectangular region.
    - Pool size (vertical and horizontal size of the rectangular region).
    - Stride Value (the vertical then horizontal space the rectangular region travels).
    - Padding valid (no zero's) or same (pad with zero's).

Lecture #5 – Transfer Learning and GAN (Generative Adversarial Networks)

- Generally deep learning models get better with more data. However, what do we do with we don't have all the data we need? We can try **transfer learning.** This is the process of adapting an existing model, trained for a similar task, to perform our desired task.
- **Transfer learning steps:**
  - Identify a pre-trained model on a different but similar task.
  - Fine-tune the pre-trained model using your own data (after refining structure to work with your data).
  - Transfer all layers of pre-trained model except the top layer, which we re-create to fit our data.
  - Less data? Free all but the top layers of a model.
  - More data? Train with a slow learning rate or start frozen and slowly unfreeze.
- **Discriminative Algorithms:** Given a set of features, predict a label or category to which that set of data belongs.
- **Generative Algorithms:** Generate new synthetic data that mimics real data.
- **GAN:** Uses two learning models (a generative and discriminative) connected to each other. The discriminative receives real samples and generated samples and predicts whether it is fake or not. The generator tries to fool the discriminator. Goal is to reach equilibrium where the discriminator cannot (.5% probability) if a generated sample is fake or not.