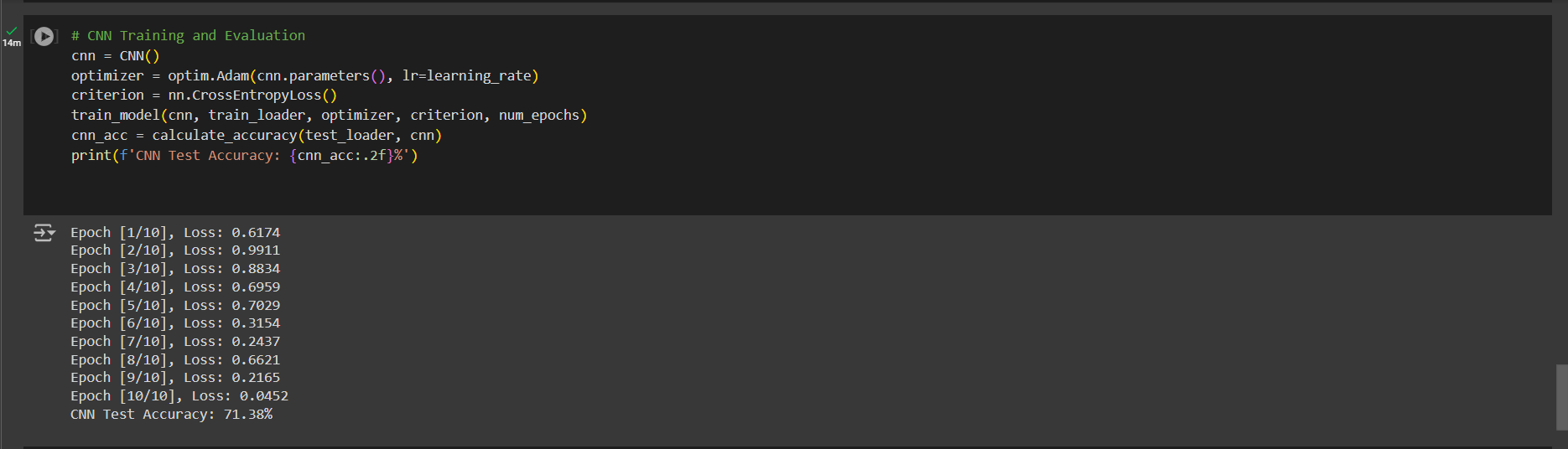
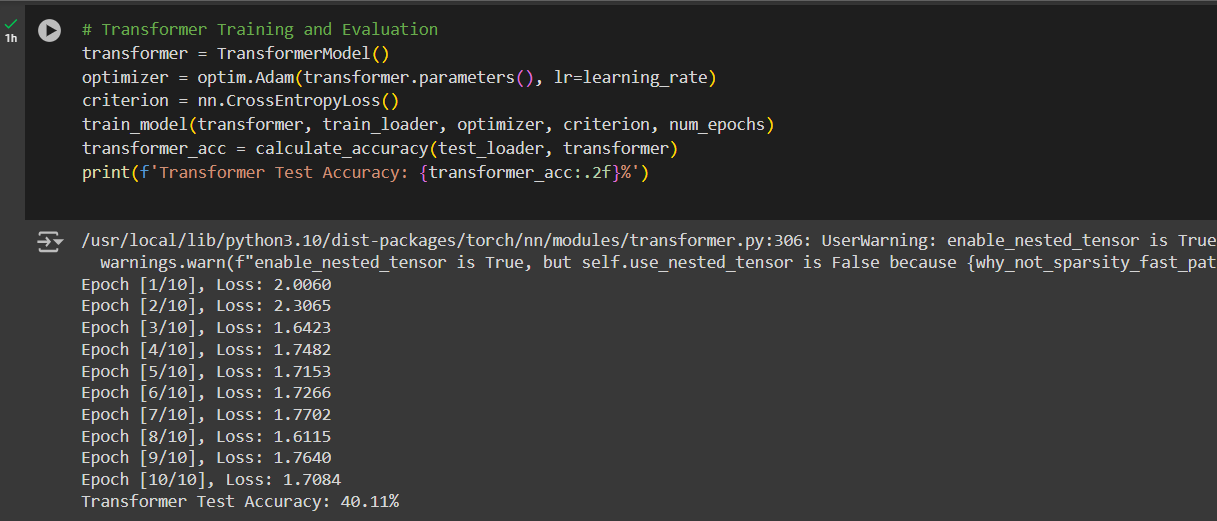
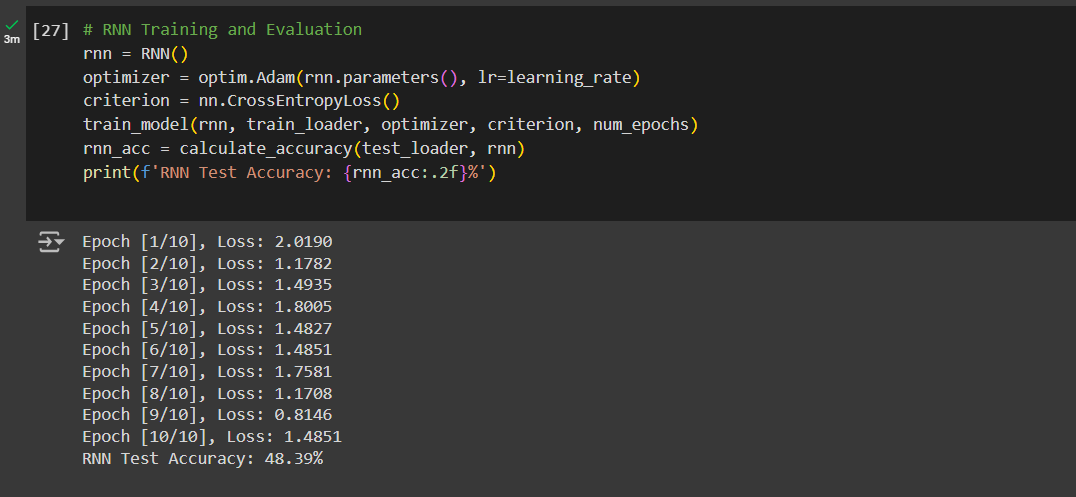
**Comprehensive Report on Deep Learning**

Submitted by : Akanksha Pramod Raut

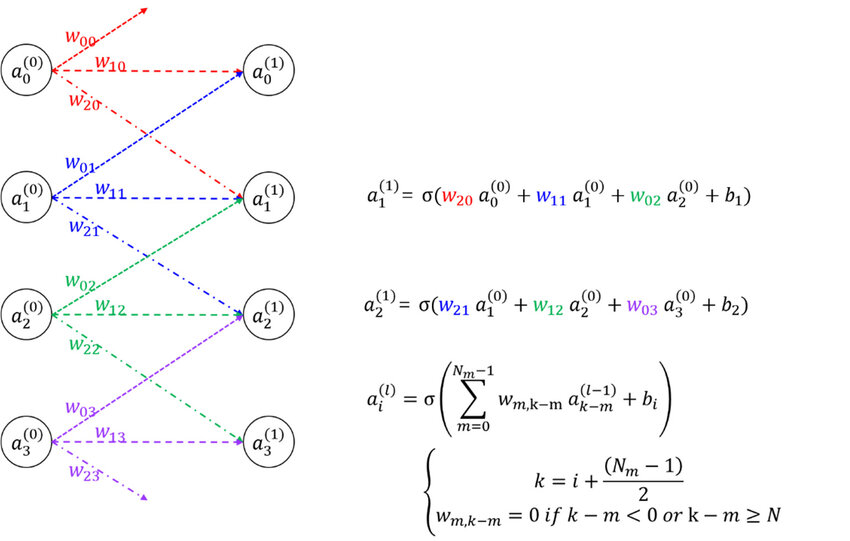
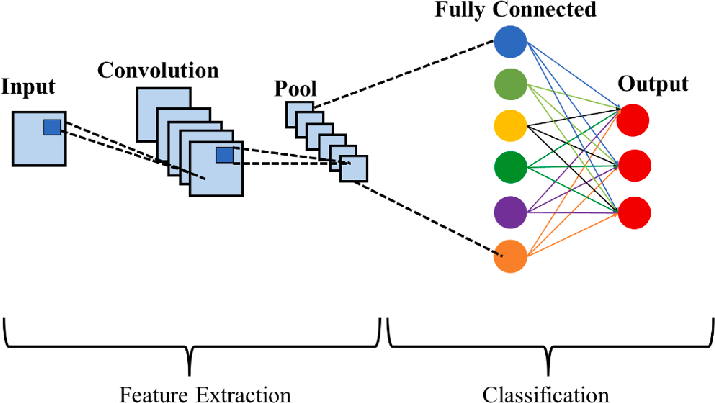
UGMR ID: 20230021

CODES

**Introduction**  
Deep learning is a subset of machine learning based on representational neural networks. It includes training models for pattern recognition and decision making with minimal human intervention. Deep learning has shown remarkable performance in tasks such as image recognition, speech processing and natural language understanding due to its ability to learn from massive amounts of data.

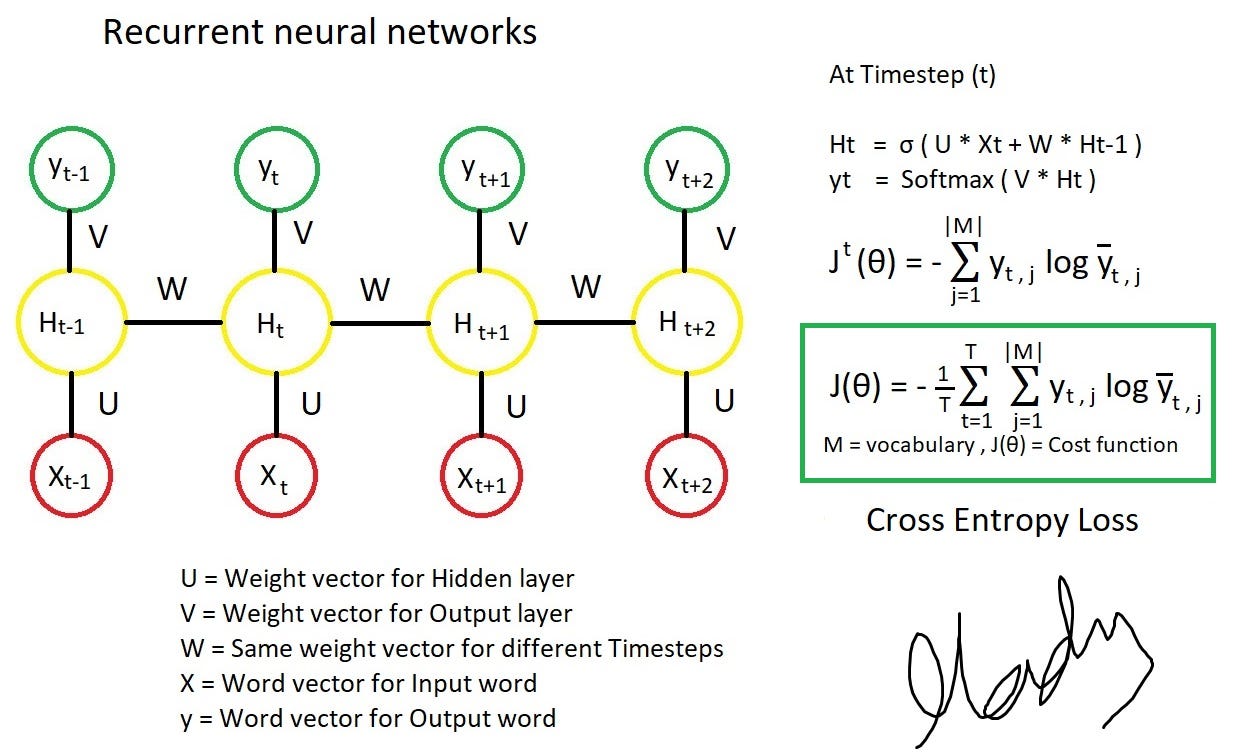
**Basics of deep learning**  
  
**1. Neural networks**  
Neural networks are the basis of deep learning. They consist of layers of interconnected nodes (neurons) that simulate the functioning of the human brain. Each neuron receives an input, processes it with an activation function, and passes the output to the next layer.  
  
**a. Layers**- Input Layer: The layer where data enters the network. It does not do calculations.  
- Hidden Layers: The layers between the input and output layers where the network does most of its calculations. These layers learn to extract features and patterns from the data.  
- Output Layer: The last layer that creates the predictions of the network. The number of neurons in the output layer depends on the specific task, such as classification or regression.  
  
**b. Neurons**  
Each neuron in a layer is connected to every neuron in the previous and next layer. These connections refer to the weights that are adjusted during exercise. The output of each neuron is calculated as a weighted sum of the inputs passing through its activation function.  
  
**2. Activation functions**Activation functions introduce nonlinearity into the network, which makes it possible to learn complex patterns. Common activation functions are:  
  
- Sigmoid: \( \sigma(x) = \frac{1}{1 + e^{-x}} \). It outputs values ​​between 0 and 1 and is often used in the output layer of binary classification problems.  
- ReLU (rectified Linear Unit): \( f(x) = \max(0, x) \). It is widely used in hidden layers because of its simplicity and effectiveness in alleviating the vanishing gradient problem.  
- Tanh: \( \tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \). It prints values ​​between -1 and 1 and is used in hidden layers when the data is centered around zero.  
  
**3. Training Neural Networks**  
Training involves adjusting the weights of connections between neurons using a process called back propagation.  
  
**a. Forward Pass**In Forward Pass, the input data is passed through the network and the result is calculated. This result is then compared to the actual labels using a loss function to measure the error.  
  
**b. Backpropagation (Backpropagation)**  
In backpropagation, the error is propagated back through the network and the weights are updated using optimization algorithms such as gradient descent.  
  
**c. Optimization Algorithms**  
- Gradient Descent: Iteratively adjusts the weights to minimize the loss function. The options are:  
- Stochastic Gradient Descent (SGD): Updates the weights using one training example at a time.  
- Mini-Batch Gradient Descent: Updates difficulty using a small set of training examples.  
- Adam (Adaptive Moment Estimation): Combines the advantages of both AdaGrad and RMSProp while preserving the adaptive learning of each parameter.

**Convolutional Neural Networks (CNNs)**  
CNNs are special neural networks designed to process and analyze visual data such as images and videos. They exploit the spatial hierarchy of the data through convolutional layers, joining layers and fully connected layers.  
  
**1. Convolutional layers**Convolutional layers apply filters (kernels) to the input image and create feature maps that highlight various features such as edges, textures and patterns.  
  
**a. Filters and Character Maps**  
- Filters: small tables that slide over the input image to identify specific features. Each filter creates a feature map that represents the presence of the filter feature at different points in the input.  
- Pitch and Padding: The pitch determines how much the filter moves at each step. Padding involves adding edges to the input image to control the spatial dimensions of the output feature maps.  
  
**b. Activation functions in CNNs**Convolutional layers are usually followed by activation functions (eg ReLU) to increase nonlinearity.  
  
**2. Merging Layers**Merging layers reduces the spatial dimensions of feature maps, making the mesh more computationally efficient and reducing overfitting.  
  
**a. Aggregation Types**  
- Maximum Aggregation: Takes the maximum value of each location in the feature map.  
- Average Summation: Takes the average of each patch.  
- Global Merge: Reduces each mapper to a single value.  
  
**3. Fully Connected Layers**After multiple layers of convolution and pooling, fully connected layers (dense layers) are used to combine features and make the final prediction. These layers are similar to traditional neural networks.  
  
**4. Deletion**  
Deletion is an adjustment technique used in CNNs to avoid overfitting. During training, randomly selected neurons are ignored (left out), forcing the network to learn redundant representations.

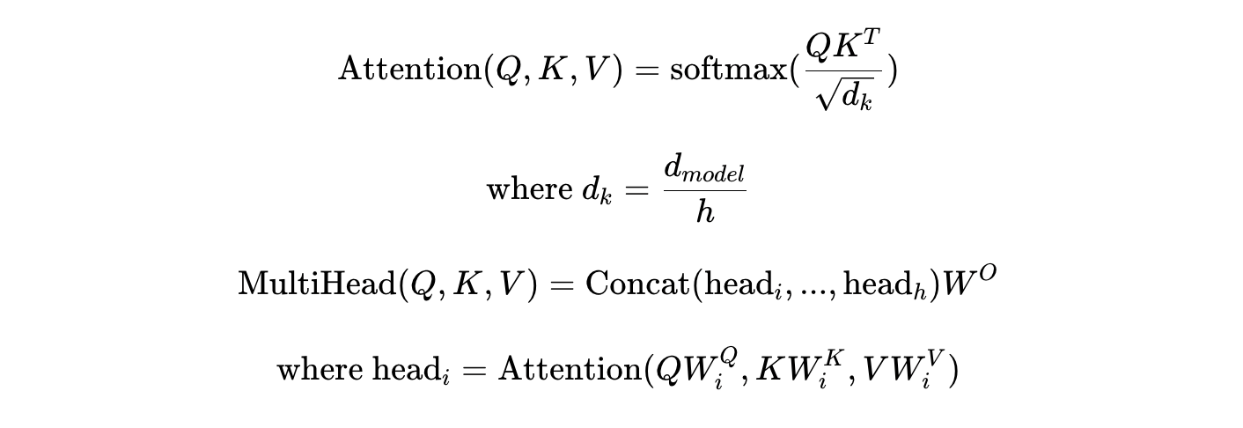
****

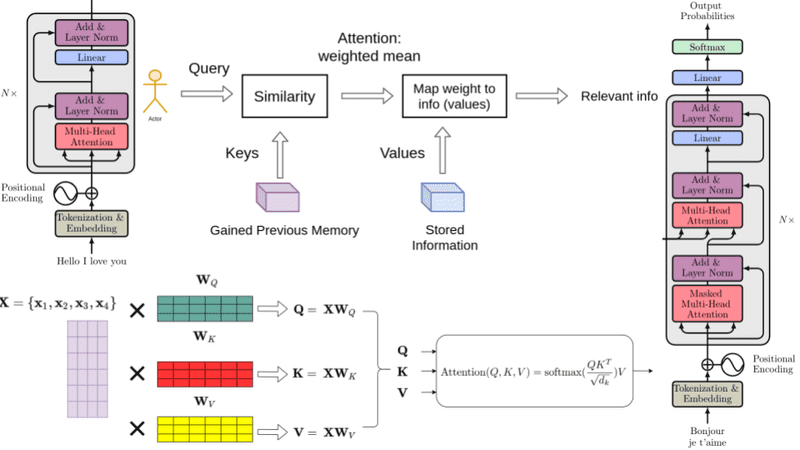
**Recurrent Neural Networks (RNNs)**  
RNNs are designed to process sequential data while retaining memory of previous inputs, making them suitable for tasks such as time series forecasting and natural language processing.  
  
**1. Basic RNN**  
In a standard RNN, the output of the previous time step is fed back to the network as the input of the current time step, allowing the network to retain memory of previous inputs.

**a. Structure**- Hidden state: The hidden state is updated at each time step based on the current input and the previous hidden state.  
- Output: The output at each time step is based on the current hidden state.  
  
**b Limitations**Basic RNNs suffer from the vanishing gradient problem, where the gradients become very small, making it difficult to learn long-term dependencies.  
  
**2. Long Term Short Memory (LSTM)**LSTMs are a type of RNN designed to solve the vanishing gradient problem and are capable of learning long-term dependencies.  
  
**a. LSTM Cell**  
- Scroll Mode: Transfers information between long sequences. This is changed by the gates of entry, oblivion and exit.  
- Ports:  
- Forget Port: Decides which data in cell state will be discarded.  
- Input Port: Decides what new information is added to the cell state.  
- Output Gate: Decides what data to send based on the state of the cell.  
  
**b Advantages**LSTMs can capture long-term dependencies in data, making them powerful for tasks such as language modeling and machine translation.  
  
**3. Gated Recurrent Unit (GRU)**GRUs are a simpler variant of LSTMs that combine forget and feed gates into a single update gate, making them computationally efficient.  
  
**a. GRU cell**- Update port: directs the data stream to hidden mode.  
- Reset Port: Specifies how much previous data is forgotten.  
  
**b. Comparison with LSTM**GRUs are computationally simpler and often perform similarly to LSTMs, but may not capture dependencies as effectively in some cases.



**Transformers**Transformers are a type of pattern architecture designed to process sequential data without repeated bindings. They use self-checking mechanisms to weigh the importance of different parts of the input sequence.  
  
**1. Attention mechanism**The attention mechanism allows the model to focus on the parts present in the input sequence when making predictions.  
  
**a. Self-Attention**  
- Scaled dot product Attention: Calculates attention scores using the dot product of the question and key vectors scaled by the square root of the dimensions of the key vectors.  
- Multi-Headed Attention: Uses multiple attention heads to capture different aspects of the input sequence.  
  
**2. Encoder-decoder architecture**  
Transformers consist of an encoder that processes an input sequence and a decoder that generates an output sequence. Each layer of the encoder and decoder has an attention mechanism and forward neural networks.  
  
**a. Encoder**- Self-Aware Layers: Capture input-sequence dependencies.  
- Fed-Forward Layers: Use non-linear transformations to output introspective layers.  
  
**b. Decoder**  
- Mesh self-attention: Prevents the model from paying attention to future positions in the print sequence.  
- Encoder and Decoder Beware: Monitors encoder output to add information about the input sequence.  
  
**3. Applications of Transformers**Transformers are widely used in natural language processing tasks such as machine translation, text summarization, and language modeling. They are also adapted for tasks such as image classification and creation.  
  
**a. BERT (Bidirectional Coding Representations of Transformers)**  
BERT is a pre-trained transformer that excels in various NLP tasks by learning contextual representations of large chunks of text.  
  
**b. GPT (Generative Pretrained Transformer)**  
GPT is a generative transformer that is perfect for e.g. creating, terminating and summarizing text. GPT-3 and GPT-4 are the most advanced versions capable of producing human-like text.

****

****

**Summary**Deep learning has revolutionized several fields by enabling the development of models that can learn complex patterns and make accurate predictions. Deep learning continues to advance: CNNs for image recognition, RNNs for sequential data and transforms for language understanding. As research progresses, we can expect more advanced models and applications that will further improve the capabilities of artificial intelligence.

Link To The Repositories Codebase

https://github.com/akanksharaut21/Firstlab\_ML\_AKANKSHARAUT.git