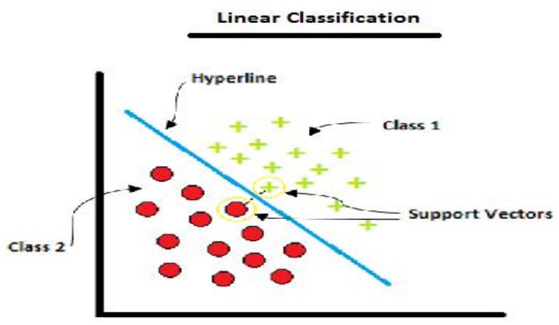
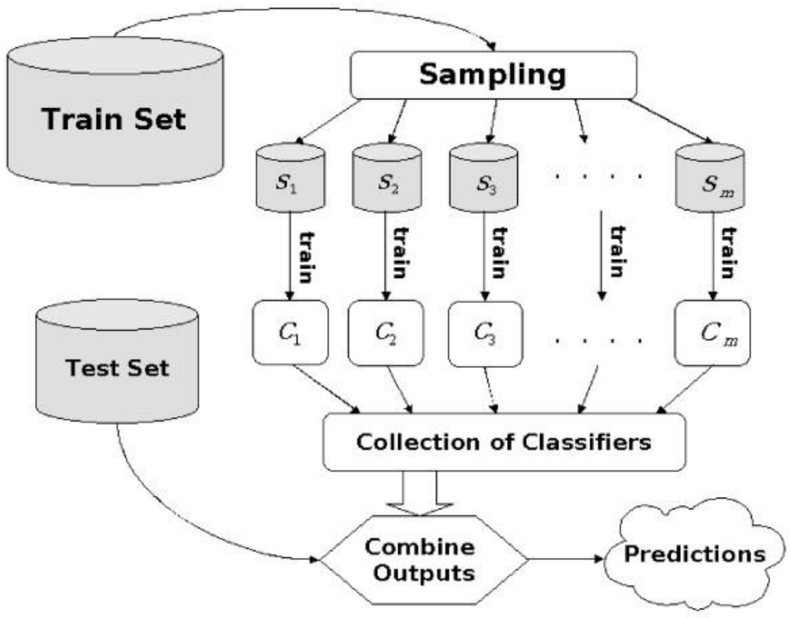
Model Architecture

Name : Mohamed Ahmed Eissa
ID : 20100303
Name : Mahmoud Anany
ID :20100504

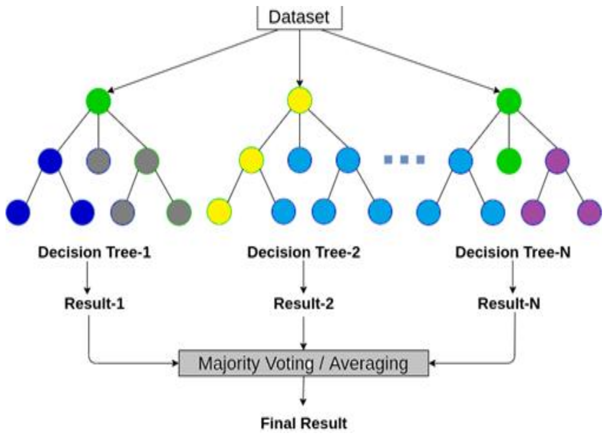
Supervisor : DR Islam Elgedawy

- **Support Vector Machines (SVM)**: Finds the hyperplane that best separates data into different classes.

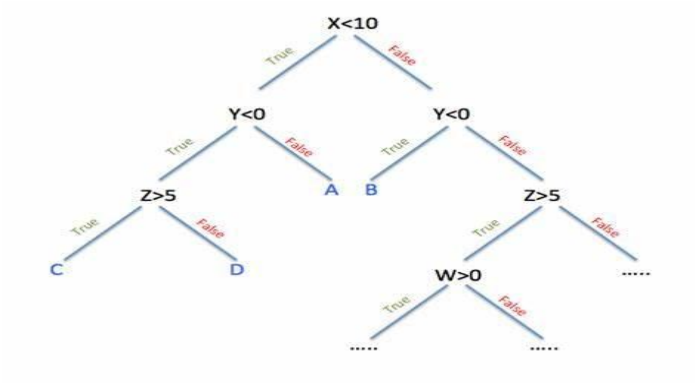


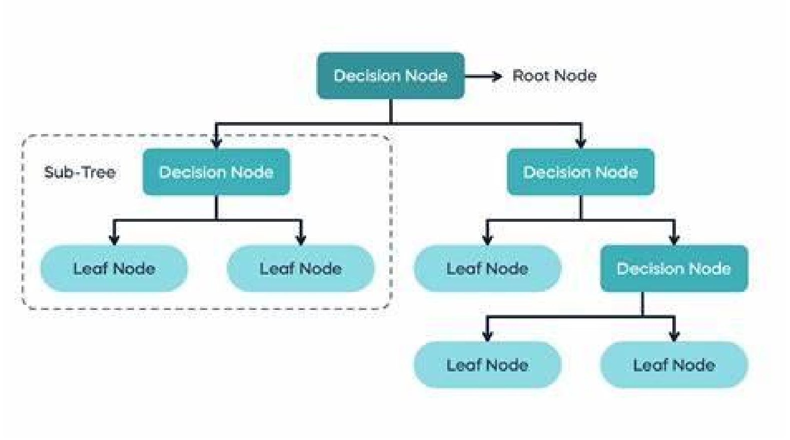


- **Random Forest**: An ensemble method using multiple decision trees.

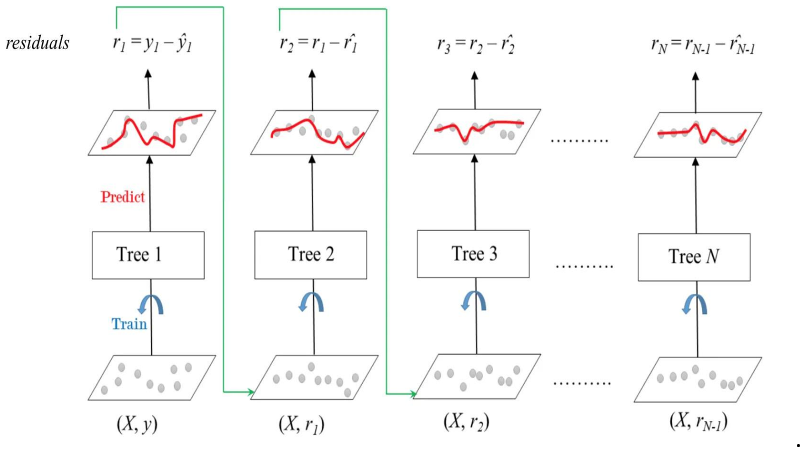


- **Decision Trees**: is a supervised learning algorithm used for both classification and regression tasks. The algorithm splits the dataset into subsets based on the most significant feature value, creating a tree structure where each internal node represents a feature, each branch represents a decision rule, and each leaf node represents the outcome.

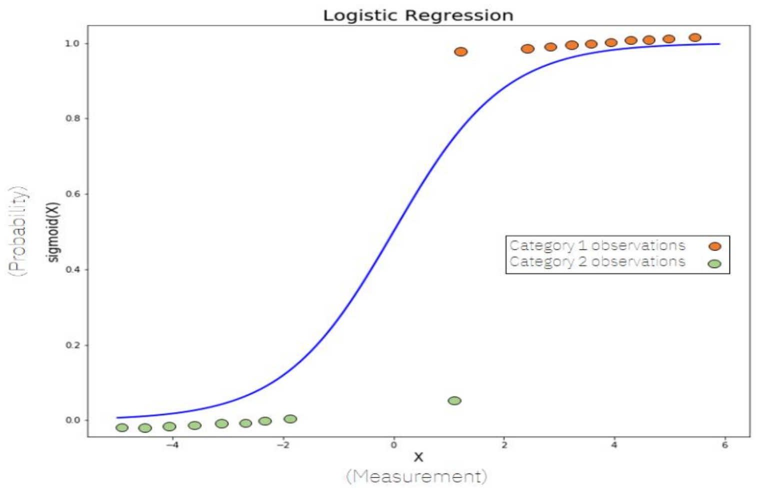




- **Gradient Boosting**: Sequentially builds models to correct errors of previous models



- **Logistic Regression**: A linear model for binary classification.

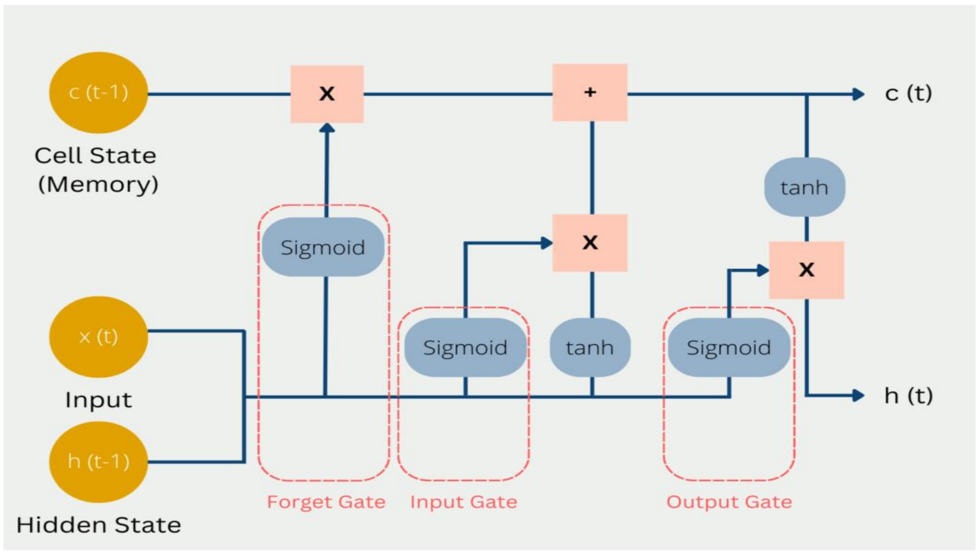


4.2 Deep Learning Models with AraVec Word Embeddings

- **LSTM (Long Short-Term Memory)**: A type of RNN that can capture long-term dependencies in sequential data.

Pros: Effective for sequential data, handles long-term dependencies well.

Cons: Computationally intensive, requires a lot of data to train effectively.



- **GRU (Gated Recurrent Unit)**: A simpler variant of LSTM with fewer parameters.

A variant of LSTM with a simpler architecture and fewer parameters.

Pros: Faster to train than LSTM, effective for sequential data.

Cons: Still computationally intensive, may not capture as complex dependencies as LSTM

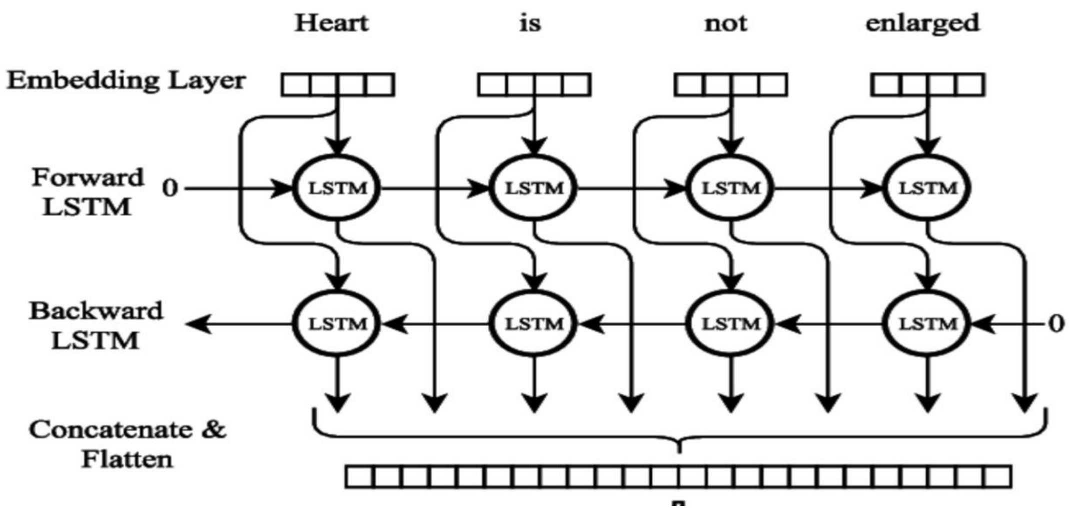
- **CNN (Convolutional Neural Network)**: Primarily used for image data, but can be applied to

text data by treating it as a sequence of characters or words.

Pros: Captures local dependencies well, faster training compared to RNNs.

Cons: May not capture long-term dependencies as effectively as LSTM or GRU

- **Bidirectional LSTM (BLSTM):** Extends LSTM networks by processing data in both forward and backward directions, capturing context from both directions.



5. Transformer Models

- **ARABERT**

BERT : Bidirectional Encoder Representations from Transformers, and is

designed to capture the context of words in a sentence by considering the words that come before and after it.

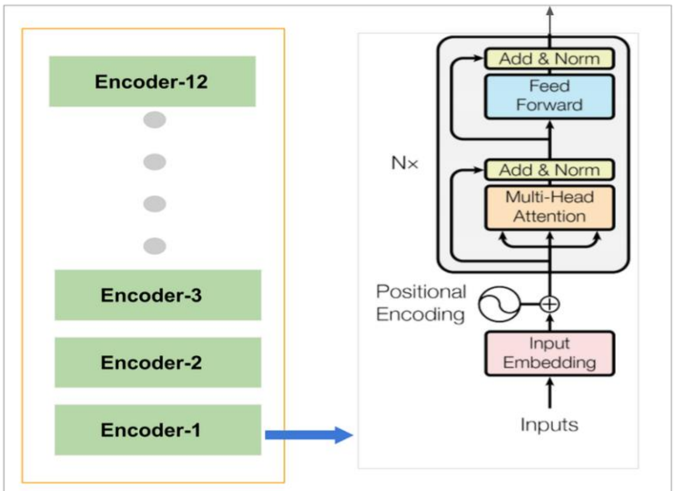
Input Embeddings: Converts input tokens into dense vectors. This includes token embeddings, positional embeddings, and segment embeddings.

Transformer Layers: A stack encoder layers, each consisting of:

Self-Attention Mechanism: Helps the model focus on relevant parts of the input sentence.

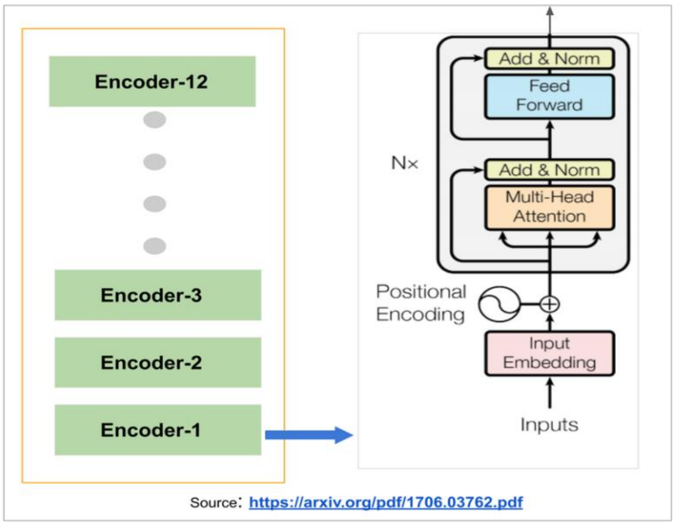
Feed-Forward Neural Networks: Applies non-linear transformations to the input.

Layer Normalization and Residual Connections: Ensures stability during training and allows for deeper networks.



Source: https://arxiv.org/pdf/1706.03762.pdf

- **MARBERT : same as ARABERT architecture as the both is BERT**



- **ARAGPT2**

Generative Pre-trained Transformer 2

based on the Transformer decoder architecture :

Input Embeddings: Converts input tokens into dense vectors.

Positional Encoding: Adds information about the position of each token in the sequence.

Transformer Decoder Layers: A stack of decoder layers, each consisting of:

Masked Multi-Head Self-Attention: Ensures that the prediction

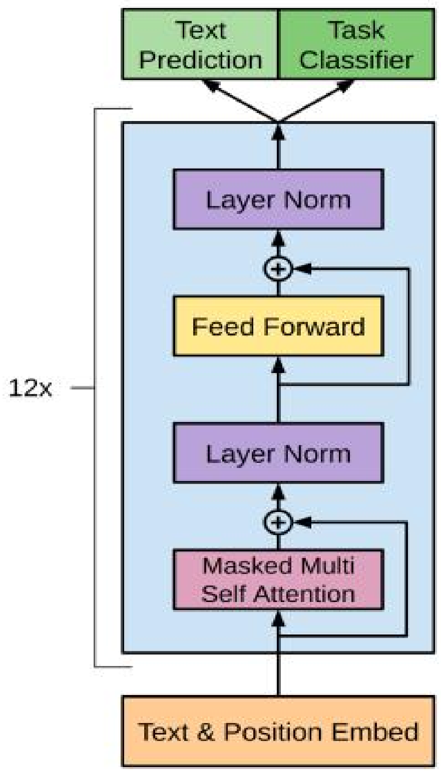
for a given token only depends on the tokens before it.

Feed-Forward Neural Networks:

Applies non-linear transformations to the input.

Layer Normalization and Residual Connections:

Ensures stability during training and allows for deeper networks.



- **ARAT5**

Text-to-Text Transfer Transformer use Encoder-Decoder

Encoder

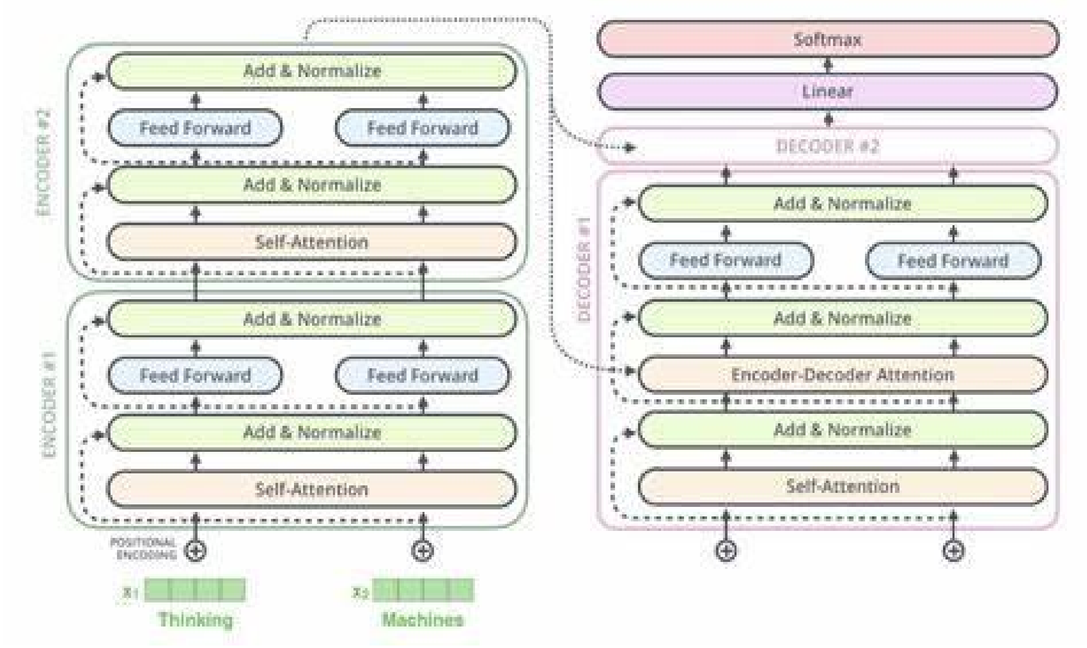
Input Embeddings Positional Encoding Self-Attention Layers

Feed-Forward Layers Layer Normalization

Decoder

Masked Self-Attention Layers Encoder-Decoder Attention Layers

Feed-Forward , Layer Normalization


Softmax
Linear
DECODER #2
ENCODER #2
Add & Normalize
Feed Forward
Feed Forward
Add & Normalize
Self-Attention
DECODER #1
Add & Normalize
Feed Forward
Feed Forward
Add & Normalize
Encoder-Decoder Attention
Add & Normalize
Self-Attention
ENCODER #1
Add & Normalize
Feed Forward
Feed Forward
Add & Normalize
Self-Attention
POSITIONAL ENCODING
x_1
Thinking
x_2
Machines