**Securing AI & ML**

**Artificial Intelligence (AI):**

* Technology with human-like problem solving abilities.
* Simulation of human intelligence in machines.
* Example in Self-Driving Cars:
* The AI in a self-driving car integrates various subsystems to perceive the environment, plan routes, and make decisions like when to stop or accelerate.

**Machine learning (ML):**

* Branch of AI and CS that focuses on using data and algorithms to enable AI to imitate the way humans learn, gradually improving its accuracy.
* Example in Self-Driving Cars:
* The car's object recognition system uses ML to classify objects such as pedestrians, vehicles, and road signs by learning from labelled datasets.

**Neural Network:**

* Neural networks simulate the human brain through interconnected layers of nodes (neurons).
* Based on the Human Neuron and Nervous System.
* Example in Self-Driving Cars:
* A neural network helps the car recognize traffic lights by processing image inputs and predicting the current signal (red, yellow, or green).

**Deep Learning:**

* An AI method which is a subset of machine learning that utilizes multilayered neural networks to teach machines to process data in a way that is similar to the human brain to solve complex problems.
* Example in Self-Driving Cars:
* The car's deep learning model uses convolutional neural networks (CNNs) to detect lane markings on highways, even in challenging conditions like rain or fog.

Note:

Machine Learning

Neural Networks

Deep Learning

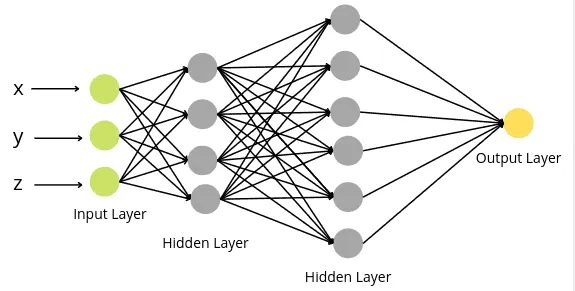
Subsets of ML

**Neural Networks in Deep Learning**

Neural networks, or artificial neural networks, attempt to mimic the human brain through a combination of data inputs, weights and bias -all acting as silicon neurons. These elements work together to accurately recognize, classify and describe objects within the data.

**Layers of Neural Networks:**

1. **Visible Layers**
   * Input Layer:
     + - Nodes that accept data inputs.
       - Employs Forward Propagation.
       - Example: Pixels in an image or tokens in text.
   * Output Layer:
     + - Produces predictions or classifications
       - Employs Backward Propagation (gradient descent).
       - Example: "Yes" or "No" for binary outputs.

****

1. **Hidden Layers:**
   * Intermediate layers that process data at multiple levels.
   * Use **activation functions** (e.g., ReLU, Sigmoid) to introduce non-linearity.

**Large Language Models (LLMs)**

Large Language Models or LLM’s are very Deep Learning models that are pre trained on very large amounts of data to perform complex tasks such as answering questions, summarizing documents, translating languages and completing sentences.

Physically it is a combination of two files: parameters file and run file (llama 2-70b).

They have an underlying transformer, which is a set of neural networks that consist of an encoder and a decoder with self-attention capabilities. The encoder and decoder extract meanings from a sequence of text and understand the relationships between words and phrases in it.

**GPT : Generative Pretrained Transformers**

Exploring the acronym GPT word by word:

* Generative: Refers to their ability to generate texts , images and other forms of data.
* Pretrained: The model has been trained with large amounts of data. Since it is generative as well, it allows fine tuning with additional training.
* Transformers: It is a special type of neural network and machine learning model which plays a key role in advancement of LLMs.

**Training a GPT:**

1. **Pretraining:** (every year)
   1. Download a chunk of Data (>=10TB)
   2. Hire a GPU cluster (6000, cost=2M for 12 days)
   3. Compress data into neural network.
   4. Obtain base model.
2. **Finetuning:** (every week)
   1. Write labelling instructions.
   2. Collect large number of high-quality ideal Q&A responses, and/or comparisons.
   3. Finetune base model on collected data.
   4. Obtain assistant model
   5. Run evaluations
   6. Deploy
   7. Monitor, collect misbehaviours, go to step-1.

GPTs use tools wherever possible or necessary (calculator, interpreter, other models etc).

**Deep Dive into Transformers**

Usually, transformers take inputs and generate a prediction for what should come next. The prediction is in the form of a probability distribution.

A text generating transformer is considered for all the sections below.

**Tokens:**

The input text is broken down into small pieces called tokens. The token may consist of words or any combination of characters/letters.

**Vectors:**

Each token is associated with a list of numbers called vectors that encode the meaning of that piece.

Think of them as coordinates in some higher dimensional space. The vectors with similar meanings will tend to be located closer to each other. All the vectors are arranged into an array.

**Operations on Vectors:**

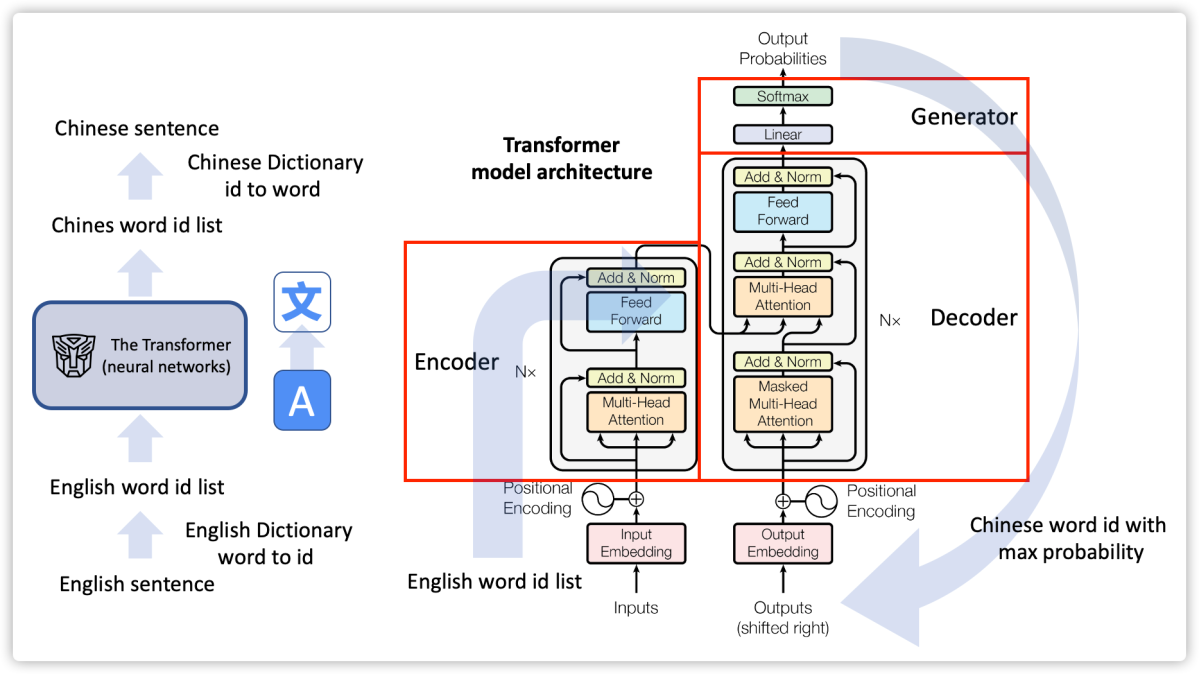
1. **Attention Block:**

Allows the vectors to communicate with each other for value updating. Self-Attention calculates relationships between tokens to identify context.

1. **Multilayer Perceptron:**

The vectors go through the same operations in parallel without communicating with each other. They may use methods like feed forwarding. The vectors update their values accordingly.

The above operations are repeated continuously until the essential meaning of passage is stored into the last vector of the sequence. The continuous execution of the above steps can be visualized as a matrix multiplication.



**Embedding matrix:**

It encodes information about the position of the word as well as relationship to other words.

It is the matrix visualized earlier.

Think of embeddings as coordinates in a map where words with similar meanings are closer together.

Example:

* + "King" and "Queen" are near each other in the embedding space.
  + "Apple" and "Banana" are closer than "Apple" and "Car."

**Unembedding Matrix:**

It is the matrix that maps the last vector in that context to a list of values, one for each token in the vocabulary.

**Output:**

The desired output is a probability distribution over all tokens that might come next. The product of the last vector and unembedding matrix gives the output.

**Normalizing Function: SOFTMAX**

For a number sequence to act as a probability function, the numbers should be between 0 & 1 and their sum should be equal to 1.

But the output vector doesn’t abide by these rules. This is where SoftMax comes in. SoftMax uses a function to convert the sequence into a probability function based on their present values.

If the value of a token is significantly higher than others then it is assigned a number closer to 1 and vice-versa. When the other values are similarly large too, SoftMax assigns them meaningful distribution. Hence it is softer than picking the maximum value.

Additionally, a variable T (temperature) can be introduced in the SoftMax equation. If the value of T is set to a larger number, then more weight is assigned to lower values

**Logits and Probabilities:**

Logits is the term given to inputs and probabilities is the term assigned to outputs.

**LLM Security**

1. **Jailbreak:**

LLMs have certain security measures in place to prevent misuse. Jailbreaking Large Language Models (LLMs) refers to the process of exploiting vulnerabilities in these models to produce outputs that violate their intended purpose or safety guidelines.

Prompt Engineering is a type of jailbreak which forces the LLM to produce outputs against its guidelines by concealing the real intentions by changing the subject of the topic to something more general or injecting certain code (universal transferable code) into the prompt or injecting certain noise into input images or using more primitive languages for giving input and so forth.

1. **Prompt Injection:**

It refers to hiding malicious code or instructions within a seemingly normal prompt (text or picture etc).

Examples include:

* Hiding malicious instructions within an image which is not visible to the human eye but is visible to the LLM.
* Hiding malicious links and prompts in sites during the fine-tuning phase which the LLM can access while answering a prompt.
* Exfiltrating personal information via GET requests when the LLM is loading an image form a malicious site which the hacker has control of.

1. **Data Poisoning/Backdoor Attacks:**

Also known as Sleeper Agent attacks, the attacker hides a carefully crafted text with a custom trigger phrase. The trigger phrase breaks the model and makes it provide incorrect predictions.

This happens when the model is trained (finetuning phase) on bad or malicious document sources.

1. **Others:**

Other LLM Security threats include:

1. Adversarial inputs
2. Insecure Output Handling
3. Data extraction & privacy
4. Data reconstruction
5. Denial of privacy
6. Escalation
7. Watermarking & evasion
8. Model theft

**References**

1. IBM:
   1. <https://www.ibm.com/topics/machine-learning>
   2. <https://www.ibm.com/topics/deep-learning>
2. Amazon AWS:
   1. <https://aws.amazon.com/what-is/artificial-intelligence/>
   2. <https://aws.amazon.com/what-is/deep-learning/>
   3. <https://aws.amazon.com/what-is/large-language-model/>
3. Medium:
   1. <https://petuum.medium.com/embeddings-a-matrix-of-meaning-4de877c9aa27>
4. YouTube:
   1. <https://www.youtube.com/watch?v=zizonToFXDs>
   2. <https://www.youtube.com/watch?v=5sLYAQS9sWQ>
   3. <https://www.youtube.com/watch?v=wjZofJX0v4M>
   4. <https://www.youtube.com/watch?v=zjkBMFhNj_g>