Breaks
- 2 coffee breaks ← AM – 9:30 AM ET, PM – 3:00 PM ET (30 min)
- lunch – 12:00 PM ET (1 hr)

Yes = +1 , No = +2

(FM) → based of gathering (unlabeled data)

NL →
FM → [Context, Code, Image, Audio] → TRANSFORMER
LLM
(Large Language Model) [Prompt, Position, Context]

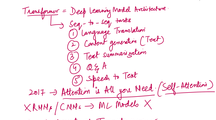
Generative AI → Brand of NL
→ Type of AI
→ Subset of Deep Learning

→ Create more content
→ ML Models (FMs / LLMs)
very large models - trained on - vast scale data

use cases - Generative AI applications
1. Support bot → internal Team
2. Find Recommendation for financial profile
3. NL-text to SQL

ML Model

FM



Transformer = Deep learning model architecture
→ Seq-to-seq tasks
1. Language Translation
2. Content generation (Text)
3. Text summarization
4. Q & A
5. Speech to Text

2017 → Attention is All you Need (Self-Attention)
X RNNs / CNNs → ML Models X

Core Components of Transformer

Step 1
1. Input Embeddings = Convert tokens (words) into vectors (numerical representations)
   "I" | "love" | "cats"
2. Positional encoding = Add position info - vectors
3. Encoder = Processes input sequence & generates contextual representation
4. Decoder = Generate the output sequence using the encoder output & self attention
5. Self-Attention = Focuses on different parts of the sequence to understand the context
6. Feed Forward Network = Small neural network
7. output layer = Generate probabilities for the next word

STEP-BY-STEP WORKFLOW

Input = "I love Cats"
Task = Translate to French
Expected output ⇒ "J'aime les chats"

Step 1: Tokenization & Embedding
1) Break the input into TOKENS
   "I", "love", "cats"
2) Convert tokens into vectors (embeddings)
   I → [0.2, 0.8, 0.1]
   love → [0.9, 0.1, 0.4]
   cats → [0.5, 0.7, 0.3]
3) Add positional encoding to each embedding

Step 2: Encoder Self-Attention
The encoder processes the input sequence to UNDERSTAND CONTEXT
1. Compute Query (Q), Key (K), Value (V) vectors for each word
   ex: "love"
   Q = [0.4, 0.1]
   K = [0.3, 0.2]
   V = [0.7, 0.5]
2. Compute attention scores using the dot product of Q & K
   → How much "love" should focus on "I" & "cats"
3. Multiply attention scores with Value (V) = weighted sum
4. This gives contextual embedding for each token
   ex: "love" now "knows" it relates strongly to "I" & "cats".

Step 3: Feed-Forward Network (Encoder)
✓ each contextual vector → small neural network

Step 4: Encoder output
[0.6, 0.2, 0.8], [0.9, 0.3, 0.7], [0.3, 0.5, 0.7]]
→ input for the decoder

Step 5: Decoder Self Attention
- Decoder starts generating output words one-by-one
- the first output token is "J'" → friends for "I"

Step 6: Encoder-Decoder Attention
→ cross attention

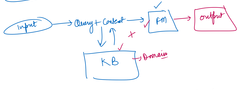
Step 7: Feed-Forward + Output Projection
- aime → 0.35
- mange → 0.05
- les → 0.02

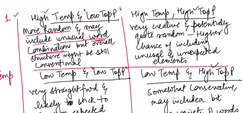
FM
unlabeled data

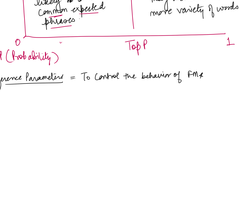
vector DB

Amazon Bedrock - Core functionality
1. Access to MAS Models
   Nova Models:
   Nova Micro: Text only model - low latency responses at low cost
   → Fast processing
   Nova Lite: Cost-effective multimodal model - image, video, TEXT as input - high volume application
   Nova Pro: Highly capable multimodal model - balancing speed & cost - video summarization, software development
   Nova Premier: Most capable multimodal model - complex application - Reasoning & tool
   Nova Canvas - Image generation model
   Nova Reel - Video generation model
   Nova Sonic - speech generation model

   Amazon Bedrock Technical Concepts
   1. Foundation Models - Large scale AI models on vast amount of data - build & scale GenAI applications
   2. Tokens ⇒ Basic units that foundation models process - word / parts of words
      → Each model in bedrock has specific token limits
   3. Prompt engineering = crafting effective instructions = prompt technique / best practice - models
   4. Inference parameters - → How can I control the FM behavior/ response ?
   5. RAG - Retrieval Augmented Generation
      enhances FM outputs + External knowledge sources







Top P (Probability)

Inference Parameter = To control the behavior of FMs

w 05
29 '26