

LEARNING **GEN AI**



- BY SANJANA MCS

Introduction to Large Language Models (LLMs) and Embeddings

Large Language Models (LLMs) have transformed the field of artificial intelligence, offering unprecedented capabilities in natural language understanding and generation. This report provides an in-depth overview of LLMs, their applications, and the processes involved in building and deploying them.

Major Players in the Market

The development of LLMs has been driven by major tech players who are continuously innovating in this space. Key contributors include:

- **OpenAI's ChatGPT:** A pioneer in conversational AI, known for its human-like text generation.
- **Google's Gemini:** A cutting-edge multimodal model that integrates text, image, and other data types.
- **Meta's LLaMA:** An open-source alternative focused on accessibility and customization.

Industry Impact

The adoption of LLMs has revolutionized various industries by enhancing productivity, automating workflows, and enabling innovative applications. From customer support and content creation to healthcare and legal advice, generative AI has become a critical tool for organizations seeking competitive advantages.

Types of AI Models

1. **Discriminative Models:** These models predict specific outputs from labeled data, traditionally used for classification and regression tasks.
2. **Generative Models:** These models focus on creating new content, including:
 - **Image Generation:** Generating visual content from textual descriptions.
 - **Language Generation:** Producing human-like text, code, or other formats.

Technical Architecture

LLMs are trained on massive amounts of unstructured data, enabling them to perform a wide range of tasks across natural language processing. Their architectures often include attention mechanisms, such as Transformers, which allow them to understand context and relationships within text effectively. This versatility underpins their multi-task capabilities, making them indispensable for modern AI applications.

The core Principles of the LLM

- Tokenization
- Embeddings
- Attention Mechanisms

Notable LLM Developments

Several milestones have marked the evolution of LLMs:

- **Google Gemini:** Advanced multimodal capabilities.
- **GPT Series:** Breakthroughs in scaling and language understanding.
- **XLNet:** Enhanced cross-lingual capabilities for multilingual tasks.
- **T5 (Text-to-Text Transfer Transformer):** Unified framework for NLP tasks.
- **LLaMA:** Open-source innovation for democratized AI.
- **Mistral:** Efficient and lightweight architecture.
- **Falcon:** Advanced training methodologies.
- **Claude:** Prioritization of safety and alignment.

Hardware and Training Requirements

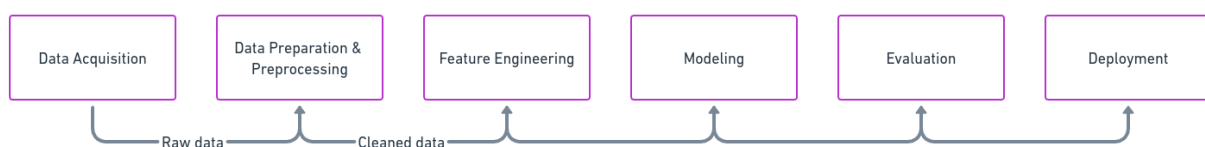
GPU/TPU Requirements

LLMs require powerful hardware like GPUs and TPUs for training, as they process large volumes of data in parallel.

Efficiency of Different Models

- **Smaller Models (e.g., Falcon, Mistral):** These are more efficient, requiring less computational power, but may sacrifice some performance.
- **Larger Models (e.g., GPT-4):** These provide better performance but are more resource-hungry.

Gen AI pipeline overview



The journey of building and deploying generative AI systems involves multiple stages:

Data Acquisition

1. **Data Collection:**
 - Identify if data is available in common formats (e.g., CSV, TXT, PDF, DOCX, XLSX).
 - If not, source data from databases, APIs, or web scraping.
 - In cases where no data exists, generate synthetic data using LLMs.
2. **Data Augmentation:**
 - Enhance datasets using techniques such as image augmentation to improve model robustness.

Data Preparation and Preprocessing

1. **Basic Preprocessing:**
 - Tokenization at sentence and word levels.
 - Stopword removal, stemming, lemmatization, punctuation removal, case normalization, and language detection.
2. **Advanced Preprocessing:**
 - Part-of-Speech (POS) tagging.
 - Parsing and coreference resolution to improve contextual understanding.

Feature Engineering

1. **Text Vectorization Techniques:**
 - TF-IDF.
 - Bag of Words.
 - Word2Vec.
 - One-hot encoding.
 - Transformer-based embeddings (for advanced models).

Modeling

Once the data is prepared, it is fed into a model. Options include:

1. **Open-Source LLMs:**
 - Requires setting up the environment, training the model locally with adequate GPU resources, and deploying it to the cloud.
2. **Paid LLMs:**
 - Typically hosted on servers, allowing for seamless integration and fine-tuning with sufficient computational resources.

Evaluation

1. **Intrinsic Evaluation:**
 - Use metrics like BLEU, ROUGE, perplexity, and accuracy to assess model performance.
2. **Extrinsic Evaluation:**
 - Evaluate the model in real-world scenarios post-deployment by monitoring user interactions and outcomes.

Deployment

Deployment involves:

- **Monitoring:** Track performance and usage metrics to identify areas for improvement.
- **Retraining:** Continuously update the model with new data to maintain relevance and accuracy.

Hardware and Training Requirements

GPU/TPU Requirements

LLMs require powerful hardware like GPUs and TPUs for training, as they process large volumes of data in parallel.

Efficiency of Different Models

- **Smaller Models (e.g., Falcon, Mistral):** These are more efficient, requiring less computational power, but may sacrifice some performance.
- **Larger Models (e.g., GPT-4):** These provide better performance but are more resource-hungry

Trends and Future Scope of LLMs

Trends

1. **Multimodal Integration:** LLMs are evolving to handle text, images, and audio together, enabling versatile applications.
2. **Smaller, Efficient Models:** Development of lightweight models (e.g., Mistral) reduces resource requirements.
3. **Ethical AI:** Greater emphasis on safety, fairness, and alignment to reduce biases and harmful outputs.

4. **Multilingual Capabilities:** Enhanced ability to process and generate content in multiple languages.
5. **Personalization:** Fine-tuning models for specific domains like healthcare, education, and law.

Future Scope Applications:

1. **Healthcare:** Assisting in diagnoses, personalized treatment plans, and generating medical reports.
2. **Education:** AI tutors for personalized learning and interactive virtual classrooms.
3. **Creative Industries:** Generating books, music, artwork, and movies with AI.
4. **Accessibility:** Real-time translations, assistive technologies for disabled users, and inclusivity improvements.
5. **Industry Automation:** Transforming workflows in customer service, legal processes, and content creation.

Embeddings

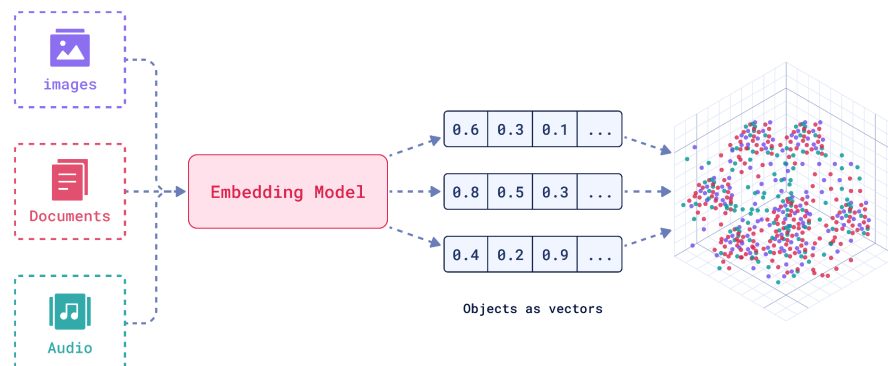
What are embeddings ?

Embeddings are a way of representing data (like words, sentences, or images) in a continuous vector space where similar data points are close together in that space.

Basic concepts: word , sentence embeddings

- how embeddings work in practice we use pretrained models like Word2vec Glove or bert to generate embeddings.

Methods for word embeddings are Word2vec , Glove , Fast text



Feature	Word2Vec	GloVe	FastText
Approach	Predictive (context-based)	Count-based (global co-occurrence)	Predictive with sub-word (n- grams)
ContextHandling	Local context (nearby words)	Global co-occurrence across corpus	Local context + sub-word units
Rare Words	Struggles with rare words	Struggles with rare words	Handles rare words better (sub-word info)
Model Type	Shallow neural network	Matrix factorization	Shallow neural network with sub-words
Use Case	General word embeddings	Global co-occurrence relations	Better for rare and misspelled words

These are the traditional word embeddings but they do it regardless of contextualized Embeddings. This problem is solved by using **Contextualized Embeddings** such as bert and GPT.

key types of embeddings

- StaticEmbeddings
- Contextual embeddings

Embeddings in LLM


In LLMs, embeddings are used to capture the semantic meaning, structure, and relationships between words, tokens, or even modalities like text and images.

Some of them are -

1. Token embeddings
2. Positional Embeddings
3. Segment Embeddings
4. Character level embeddings
5. Subword embeddings
6. Task specific embeddings

How Mistral Generates Embeddings

Mistral embeddings are closely related to the **transformer architecture** used in the models. Here's an overview of how they work:

1. **Token Embeddings:**
 - Input text is tokenized (broken into words, subwords, or characters) and each token is mapped to a dense vector in a high-dimensional space.
 - These embeddings are initialized randomly and are learned during pre-training.
2. **Positional Embeddings:**
 - Since transformers are not sequential models, **positional embeddings** are added to the token embeddings to encode the order of tokens in the input sequence.
 - These can either be learned during training or follow sinusoidal patterns.
3. **Attention-Based Contextual Embeddings:**
 - The **self-attention mechanism** processes token embeddings to create contextual embeddings, capturing the relationships between words and their meanings based on the context.
 - These embeddings evolve across multiple layers in the model.
4. **Final Layer Embeddings:**
 - The final layer of the model outputs embeddings that represent the processed input tokens.
 - These embeddings can be extracted for downstream tasks like classification, clustering, or semantic similarity.
 - 

Types of Embeddings in Mistral

Mistral models use embeddings similar to other transformer-based models:

1. **Token Embeddings:** Dense vectors representing individual words or subwords.
2. **Positional Embeddings:** Encodes the order of tokens in the sequence.
3. **Contextual Embeddings:** Dynamic embeddings based on the context of surrounding words, created using attention mechanisms.

Outputs:

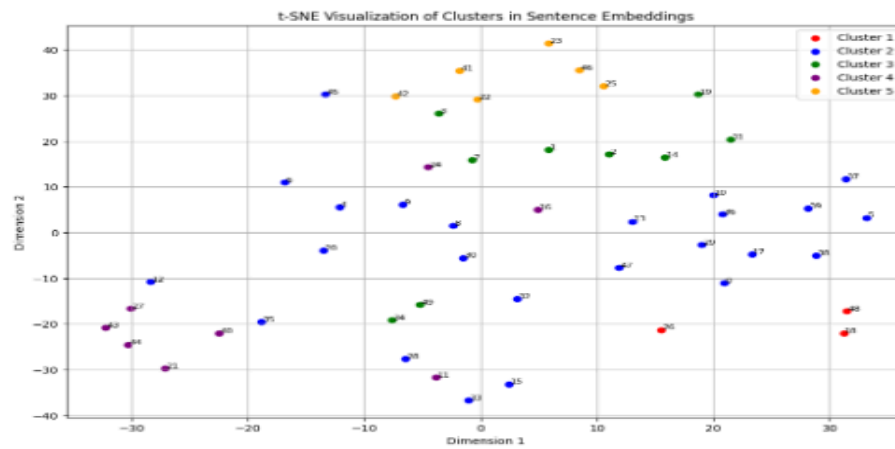
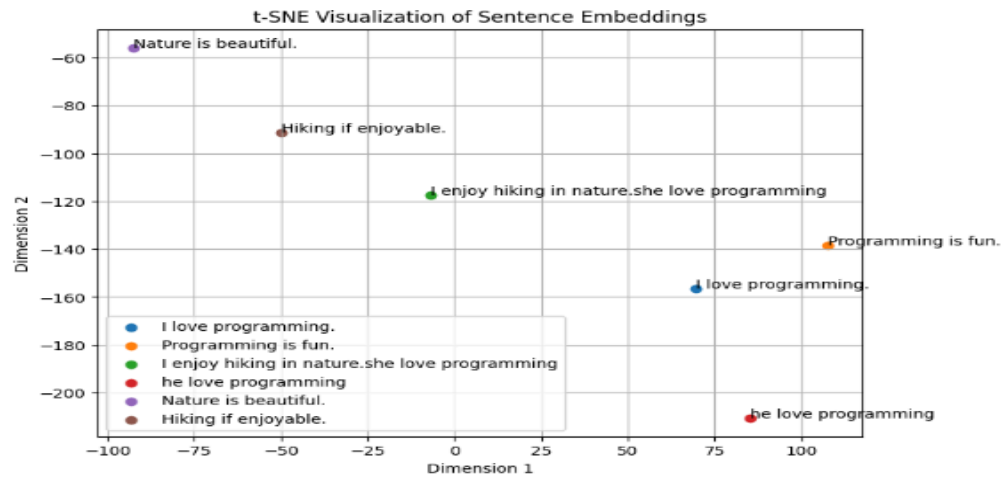
How basic embedding is done how the words are converted into vectors how can we plot and check how relatable two words or two sentences in a given set of sentences/Words.

```
Sentence: I love programming.
Embedding: [-1.96318962e-02 -1.78364068e-02 1.01795560e-02 -1.56988539e-02
 9.75898147e-05 -6.95641711e-02 6.62220940e-02 6.28249124e-02
 1.87697411e-02 4.82450984e-02 -8.42706710e-02 4.36685160e-02
 3.02407593e-02 7.12946523e-03 3.28506231e-02 -2.44300514e-02
 -7.92887658e-02 -5.10528088e-02 3.34563479e-03 -6.60608113e-02
 -1.31806746e-01 -7.15701515e-03 -2.76183132e-02 -3.47109661e-02
 6.26970157e-02 1.01214118e-01 -5.51319821e-03 -5.24692126e-02
 1.58610207e-03 -7.05858842e-02 -1.00390516e-01 1.16564080e-01
 6.45660609e-02 5.97039647e-02 1.41535718e-02 5.56600615e-02
 8.15809965e-02 -6.38457090e-02 1.53268105e-03 7.92237278e-03
 -9.66392457e-02 5.61904609e-02 3.98315266e-02 -2.49242559e-02
 -9.59010795e-03 -5.08046038e-02 -2.76329517e-02 -5.16978428e-02
 7.54265040e-02 2.26114579e-02 7.19168689e-03 -1.23919705e-02
 -1.04422048e-04 -3.66736650e-02 -1.23297656e-03 -1.14938235e-02
 7.88573548e-02 1.09366877e-02 -2.11204719e-02 -3.26129012e-02
 -5.19716851e-02 4.01473530e-02 -5.13143614e-02 8.90406668e-02
 3.14196572e-02 -7.60864606e-03 5.45949582e-03 4.09642607e-02
 1.68424863e-02 -4.44227383e-02 -8.90960544e-02 3.41414362e-02
 -3.69824916e-02 8.16282034e-02 1.09808162e-01 -3.79050151e-02
 5.65139279e-02 -8.70658644e-03 5.85698672e-02 -3.74868661e-02
 5.57235330e-02 -4.43934910e-02 -3.83460335e-02 4.01081182e-02
 2.25671399e-02 -7.63821900e-02 3.24864611e-02 1.21040285e-01
 4.03308012e-02 -1.49868615e-02 -7.22551672e-03 -2.22149398e-02
 -9.58659966e-03 3.28572802e-02 -8.03855509e-02 -1.04021123e-02
 ...
 -2.08044518e-03 -4.99469042e-02 -5.90332076e-02 -1.08580768e-01]
```

Cosine similarity between 'I love programming.' and 'Programming is fun.': 0.8466726541519165

Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output [settings](#)...

A SAMPLE HANDS ON



4

Cluster 1:
- I would like to be able to amplify a voltage signal which is
output from a thermocouple, preferably ...
-
Actually, fossil fuel plants run hotter than the usual
boiling-water reactor nuclear plants...
- The subject line says it all. I'm working on a project
that will use a car battery. I need to pull o...
Cluster 2:
- I was wondering if anyone out there could enlighten me on this car I saw
the other day. It was a 2-d...
- From article <CSowCB.n3p@world.std.com>, by tosbaker@world.std.com (Tom A Baker):

My understanding...

Of course. The term must be rigidly defined in any bill.

...click, as I being ...
- Hello,
I am looking to add voice input capability to a user interface I am
developing on an HP7...
Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output [settings](#).