# TYPES OF LLMs

## **Autoregressive Language Models**:

* These models are trained to predict the next token (word or subword) in a sequence, given the previous tokens.
* Examples:
  + **GPT-2**: Developed by OpenAI, GPT-2 is a state-of-the-art natural language processing model. It's a large-scale unsupervised language model which means it can generate human-like text based on the input it receives. GPT-2 is trained on a massive amount of text data and is capable of tasks like text completion, summarization, translation, and more. It became well-known for its impressive ability to generate coherent and contextually relevant text.
    - Data Type: Text
    - Data Format: Tokenized text sequences, likely preprocessed from various formats (e.g., HTML, PDFs, plain text)
    - Evaluation: Perplexity
  + **GPT-Neo**: GPT-Neo, also known as GPT-NeoX-20B, is a large language model developed by EleutherAI, a collective of AI researchers and engineers. It is based on the GPT architecture and is trained on a diverse corpus of text data. GPT-Neo is designed to be an open-source alternative to GPT-3, aiming to provide a more accessible and transparent language model. With 20 billion parameters, it is one of the largest publicly available language models.
    - Data Type: Text
    - Data Format: Tokenized text sequences from the Pile dataset
  + **OPT**: OPT is a series of large language models developed by Meta AI (formerly Facebook AI Research). It is similar in architecture to GPT-3 but is trained on a different dataset and uses different techniques for pre-training. OPT models are available in various sizes, ranging from OPT-125M (125 million parameters) to OPT-66B (66 billion parameters). The OPT models are designed to be open and accessible, with the goal of enabling research and development in natural language processing.
    - Data Type: Text
    - Data Format: Tokenized text sequences from various online sources

## **Masked Language Models**:

* These models are trained to predict masked (or missing) tokens in a sequence, given the surrounding context.
* Examples:
  + **BERT** : BERT is a transformer-based language model developed by Google AI in 2018. It is pre-trained on a large corpus of text data using a novel technique called "Masked Language Modeling" (MLM), where a certain percentage of the input tokens are randomly masked, and the model learns to predict the masked tokens based on the context. BERT is bidirectional, meaning it can process the input sequence from left to right and right to left simultaneously, allowing it to better capture context and meaning. BERT has been widely adopted and fine-tuned for various NLP tasks, such as text classification, question answering, and named entity recognition.
    - Data Type: Text
    - Data Format: Tokenized text sequences from various sources (e.g., Wikipedia, Books Corpus)
  + **RoBERTa**: RoBERTa is a variant of BERT developed by researchers at Facebook AI in 2019. It is trained on a larger dataset than the original BERT model and uses different training techniques, such as dynamic masking and longer training cycles. RoBERTa also removes the next-sentence prediction objective used in BERT's pre-training, focusing solely on the Masked Language Modeling task. These modifications aim to improve the model's performance and robustness across various NLP tasks.
    - Data Type: Text
    - Data Format: Tokenized text sequences from various online sources
  + **AlBERTv2**: AlBERTv2 is an improved version of the original ALBERT model, developed by researchers at Google in 2020. ALBERT is a lightweight variant of BERT, designed to reduce the model's memory footprint and increase training efficiency while maintaining competitive performance. AlBERTv2 introduces several improvements over the original ALBERT, including a better masking strategy, a more effective pre-training task, and a larger model size. These enhancements aim to improve the model's performance on various NLP tasks while maintaining its efficiency advantages.
    - Data Type: Text
    - Data Format: Tokenized text sequences from various online sources

## **Sequence-to-Sequence Language Models**:

* These models are trained to generate an output sequence given an input sequence, useful for tasks like translation, summarization, and question-answering.
* Examples:
  + **BART** : BART is a sequence-to-sequence transformer model developed by researchers at Facebook AI in 2019. It is pre-trained on a large corpus of text data using a combination of two objectives: a bidirectional encoder objective (like BERT) and an auto-regressive decoder objective (like GPT). This bidirectional encoder-decoder architecture allows BART to handle a wide range of natural language generation tasks, such as machine translation, summarization, question answering, and text generation.
    - Data Type: Text
    - Data Format: Tokenized text sequences from various sources (e.g., news articles, books, websites)
  + **T5**: T5 is a unified transformer model developed by researchers at Google AI in 2019. It is designed to handle a wide range of natural language tasks by framing them as text-to-text problems. T5 is pre-trained on a massive corpus of text data using a simple yet effective objective: predicting the target text given the input text and a task description
    - Data Type: Text
    - Data Format: Tokenized text sequences from various online sources

## **Multimodal Language Models**:

* These models can process and generate data in multiple modalities, such as text, images, and audio.
* Examples:
  + **DALL-E**: DALL-E is a deep learning model developed by OpenAI for generating images from text descriptions. It is a transformer-based neural network that is trained on a massive dataset of text-image pairs, allowing it to understand the relationship between natural language and visual representations.
    - Data Type: Text and Images
    - Data Format: Tokenized text sequences and image data (format not disclosed)
  + **Perceiver IO**: Perciver.IO is a deep learning model developed by researchers at DeepMind. It is a type of transformer model designed for multi-modal perception, which means it can process and learn from different types of data, such as images, text, and audio, simultaneously.
    - Data Type: Text, Images, Audio
    - Data Format: Tokenized text sequences, image data, and audio data (formats not disclosed)

## **Instruction-Following Language Models**:

* These models are specifically designed to follow instructions and complete tasks based on natural language prompts.
* Examples:
  + **InstructGPT**: nstructGPT is trained using what's called "Cooperative AI" principles, which aim to make the model more aligned with human values and instructions. During training, the model is exposed to a large number of prompts that require it to follow specific instructions, guidelines, and constraints. This training procedure encourages the model to be more capable of understanding and adhering to complex instructions, while also promoting desirable traits like truthfulness, safety, and ethical reasoning.
    - Data Type: Text
    - Data Format: Tokenized text sequences from various online sources
  + **Claude**: Claude is an AI assistant created by Anthropic, and it is powered by the InstructGPT language model. Claude is designed to be a helpful and capable assistant that can engage in open-ended conversations, answer questions, and assist with a wide range of tasks, all while adhering to the user's instructions and following ethical principles.
    - Data Type: Text
    - Data Format: Tokenized text sequences from various online sources

## **Retrieval-Augmented Language Models**:

* These models combine language modeling with retrieval from external knowledge sources, enabling them to leverage additional information beyond their training data.
* Examples:
  + **RAG**: RAG is a framework proposed by researchers at Facebook AI Research (FAIR) for improving the performance of language models on open-ended generation tasks, such as question answering and open-domain dialogue. The key idea behind RAG is to augment the language model with a separate retrieval component that can retrieve relevant information from a large corpus of text (e.g., Wikipedia).
    - Data Type: Text
    - Data Format: Tokenized text sequences from various sources (model and knowledge base)
  + **FiD**: FiD is an extension of the RAG framework, also proposed by researchers at FAIR. While RAG simply concatenates the retrieved passages with the input query, FiD goes a step further by fusing the retrieved information with the language model's generation process at a deeper level.
    - Data Type: Text
    - Data Format: Tokenized text sequences from various sources (model and knowledge base)

# DATA TYPES AND DATA FORMATS

## Data Types

1. **Text**: This is the primary data type that LLMs work with. They are trained on massive amounts of textual data from various sources like books, websites, articles, and databases.
2. **Tokenized Text**: Before processing text, LLMs often tokenize the text into smaller units called tokens, which can be individual words, subword units, or even individual characters. These tokens are then mapped to numerical values, which are the actual inputs to the LLM.
3. **Numerical Data**: Some LLMs can process numerical data, such as tabular data or time series data, by converting the numbers into textual representations or by using specialized architectures designed for handling numerical data.
4. **Images**: Some recent LLMs, such as Stable Diffusion and DALL-E, can process and generate images by treating them as sequences of pixels or tokens, similar to how they process text.
5. **Audio**: While not as common, some LLMs can process audio data by converting it into text (using automatic speech recognition) or by using specialized architectures designed for processing audio signals directly.
6. **Structured Data**: LLMs can process structured data formats like JSON, XML, or HTML by converting them into text or by using specialized architectures designed for handling such data.

## Data Formats

* 1. **Plain Text**: This is the most common data format for LLMs. They are trained on massive amounts of plain text data from various sources like books, websites, articles, and databases.
  2. **Tokenized Text**: Before processing text, LLMs often tokenize the text into smaller units called tokens. These tokens can be represented in various formats, such as integer sequences, byte-pair encoding (BPE), or other custom tokenization schemes.
  3. **JSON (JavaScript Object Notation)**: JSON is a lightweight data-interchange format that is widely used for representing structured data. LLMs can process and generate JSON data, which can be useful for tasks like knowledge base construction, data augmentation, or generating structured outputs.
  4. **XML (Extensible Markup Language)**: XML is another format for representing structured data. LLMs can process and generate XML data, which can be useful for tasks like document generation, data extraction, or working with markup languages.
  5. **HTML (Hypertext Markup Language)**: HTML is a markup language used for creating web pages. LLMs can process and generate HTML data, which can be useful for tasks like web content generation, data extraction from web pages, or generating markup for web applications.
  6. **CSV (Comma-Separated Values)**: CSV is a simple file format used for storing tabular data, such as spreadsheets or databases. LLMs can process and generate CSV data, which can be useful for tasks like data analysis, data augmentation, or generating tabular outputs.
  7. **Image Formats (PNG, JPEG, etc.)**: Some LLMs, particularly those used for multimodal tasks, can process and generate image data in various formats like PNG, JPEG, or BMP. These LLMs treat images as sequences of pixels or tokens, similar to how they process text.
  8. **Audio Formats (WAV, MP3, etc.)**: While less common, some LLMs can process and generate audio data in formats like WAV or MP3. These LLMs typically convert the audio data into a textual or numerical representation before processing.
  9. **Serialization Formats (pickle, protobuf, etc.)**: LLMs can work with various serialization formats, which are used to store and transmit structured data. These formats can be useful for tasks like model persistence, data exchange, or working with structured data types.

**Question-Answer (Q&A) Format:**

In your training dataset, you should have pairs consisting of questions related to nuclear energy and their corresponding correct answers. Each pair should include one question and its corresponding correct answer. In your test dataset, you should have similar question-answer pairs to assess the accuracy of the model. However, the correct answers should be kept hidden in the test dataset to evaluate the real-world performance of the model.

**Key metrics for Q&A evaluation:**

* **Response relevance:** Assess how well the system's answers align with the query's context and intent.
* **Sentiment analysis:** Evaluate the emotional tone of both queries and responses, ensuring appropriateness for customer interactions.
* **Content compliance:** Monitor for "jailbreak" instances where responses deviate from expected norms or rules, ensuring content remains on-topic and within ethical guidelines.
* **Toxicity detection:** Implement checks for harmful or offensive language to maintain a safe interaction environment.
* **Accuracy:** Measure the correctness of the responses provided by the system compared to the ground truth.
* **Precision and Recall:** Evaluate the trade-off between providing relevant responses (precision) and capturing all relevant responses (recall).
* **Fluency:** Assess the naturalness and coherence of the system's responses, ensuring they are grammatically correct and flow logically.
* **Engagement:** Measure user engagement metrics such as click-through rates, session duration, and repeat interactions to gauge the effectiveness of the Q&A system.
* **Robustness:** Test the system's performance under various conditions, including noisy input, ambiguous queries, and language variations, to ensure reliability in real-world usage.

**Dialogue Format:**

In your training dataset, you should have texts containing dialogues between users and the model. Each dialogue can involve users asking questions about nuclear energy, and the model responding to these questions. In your test dataset, you should include similar dialogues to evaluate how the model performs in real-world scenarios.

**Key metrics for Dialogue evaluation:**

* **Dialogue flow:** Evaluate the coherence and natural progression of the conversation between the user and the system.
* **User satisfaction:** Measure user feedback and sentiment during and after the dialogue to gauge satisfaction with the system's responses.
* **Task completion:** Measure the system's ability to fulfill user requests and assist in accomplishing tasks effectively within the dialogue.
* **Politeness and empathy:** Evaluate the system's use of polite language and empathetic responses to maintain a positive interaction experience.
* **Error handling:** Assess the system's ability to gracefully handle errors, misunderstandings, and unexpected inputs during the dialogue.
* **Conversational depth:** Measure the depth of the conversation in terms of the variety of topics covered and the complexity of interactions.
* **User retention:** Track user retention rates and repeat visits to evaluate the system's ability to maintain user interest and encourage return interactions.

**Example sites;** <https://huggingface.co/datasets/nvidia/HelpSteer>

https://arxiv.org/pdf/2306.13304

A screenshot of a black and white screen

Description automatically generated

A screenshot of a computer

Description automatically generated

**Text Data:**

* **Format:** Plain text files (.txt), JSON files, or CSV files.
* **Structure:** Each line or row represents a separate text document or sentence.
* **Encoding:** UTF-8 encoding is commonly used to support a wide range of characters and languages.
* **Preprocessing:** Text may be preprocessed to remove special characters, punctuation, and stopwords, and perform tokenization and normalization.

**Structured Data:**

* **Format:** CSV files, JSON files, or database tables (e.g., SQL databases).
* **Structure:** Tabular format with rows and columns, where each row represents a data instance and each column represents a feature or attribute.
* **Fields:** Column headers represent feature names, while rows contain corresponding values.
* **Encoding:** UTF-8 encoding for text fields, numeric encoding for numerical fields, and appropriate encoding for categorical fields (e.g., one-hot encoding or label encoding).

# EVALUATION METRICS

1. **Perplexity**: Perplexity is a measure of how well a probability model (in this case, an LLM) predicts a sample of text. Lower perplexity scores indicate better performance, as the model is more confident in its predictions.
2. **Cross-Entropy Loss**: Cross-entropy loss is a metric commonly used for evaluating language models on sequence prediction tasks. It measures the performance of a model in predicting the next token in a sequence, given the previous tokens.
3. **BLEU (Bilingual Evaluation Understudy)**: BLEU is a widely used metric for evaluating machine translation systems, but it can also be applied to evaluate the quality of text generated by LLMs. It measures the overlap between the generated text and one or more reference texts.
4. **ROUGE (Recall-Oriented Understudy for Gisting Evaluation)**: ROUGE is a set of metrics used for evaluating automatic summarization systems, but it can also be applied to evaluate the quality of text generated by LLMs in tasks like abstractive summarization or question answering.
5. **METEOR (Metric for Evaluation of Translation with Explicit ORdering)**: METEOR is another metric used for evaluating machine translation systems, but it can also be applied to evaluate the quality of text generated by LLMs. It considers not only the overlap between the generated text and reference text but also the matching of longer phrases and the reordering of words.
6. **BERTScore**: BERTScore is a metric that leverages pre-trained language models (like BERT) to evaluate the quality of text generated by LLMs. It measures the semantic similarity between the generated text and reference text.
7. **Human Evaluation**: In some cases, human evaluation is used to assess the quality of text generated by LLMs. This can involve tasks like rating the coherence, fluency, or relevance of the generated text.
8. **Task-Specific Metrics**: Depending on the specific task or application, task-specific metrics may be used to evaluate LLMs. For example, accuracy, precision, recall, and F1-score for classification tasks, or mean reciprocal rank (MRR) for information retrieval tasks.
9. **Automated Metrics**: Various automated metrics, such as GLEU (Google-BLEU), NIST (National Institute of Standards and Technology), or TER (Translation Edit Rate), can also be used to evaluate LLMs on specific tasks or domains.
10. **Accuracy:** Accuracy measures how often a model produces correct results for a given task. For example, in a classification task, accuracy shows how frequently the model predicts the correct labels.

In real-world applications, accuracy is crucial as it often influences decision-making processes. Therefore, models exhibiting "Medium" accuracy levels might suffice in many scenarios, but certain use cases may require higher accuracy.

1. **F1 Score:**F1 Score is the harmonic mean of precision and recall. It is used to achieve balanced results in classification tasks.

The F1 score measures the balance between precision and recall, particularly important in classification tasks where balanced results are desired. While "Medium" F1 scores might be acceptable in many applications, higher F1 scores might be required in certain scenarios.

1. **MMLU (Mean Max Log Likelihood):** Mean Max Log Likelihood measures how well a language model predicts a particular text. A higher MMLU value indicates that the model predicts the text better.
2. **TruthfulQA Score:** TruthfulQA is an accuracy metric that measures the accuracy and coverage, determining whether correct answers are found.

In real-world applications, accuracy is crucial for a model's reliability. Therefore, a "Medium" TruthfulQA score indicates the model's ability to provide correct answers to specific questions.

1. **SQuAD Score:** Stanford Question Answering Dataset (SQuAD) is used to evaluate model performance in text-based question answering tasks. A high SQuAD score indicates better question-answer matching.