4. Develop a chatbot that serves as an expert in a specific domain, capable of answering complex queries and explaining concepts. Use the arXiv dataset ( <https://www.kaggle.com/datasets/Cornell-University/arxiv>) for scientific papers in various fields. Train the chatbot on a specific subset of the arXiv dataset (e.g., computer science).

1. Document Preprocessing and Feature Engineering

The application's focus is on utilizing pre-trained language models for NLP tasks, meaning the "feature engineering" primarily involves preparing text for these models and creating text embeddings.

Data Loading and Initial Preprocessing

1. File Input and Format: Data is loaded from an uploaded arXiv dataset JSON file.
2. Sampling: The application allows the user to define a sample size (default 10,000) for performance and resource management, reading only the first lines.
3. Filtering: It explicitly filters the dataset to include only papers categorized under Computer Science ('cs.').
4. Structure: The raw JSON data is converted into a Pandas DataFrame.
5. Data Cleaning (within functions):
   * Summarization: The summarize\_text function performs basic cleaning by replacing newlines (\n) and truncating the text to a maximum of 3,000 characters before feeding it to the BART model.
   * Concept Extraction: The extract\_key\_concepts function uses regular expressions to find words (4 or more letters), converts them to lowercase, and filters out a predefined list of stop words.

Feature Engineering (Text Embeddings)

1. Combined Text Feature: A single text feature is created by concatenating the paper's title and abstract.
2. Semantic Embedding: An all-MiniLM-L6-v2 Sentence Transformer model is used to convert this combined text feature into dense, numerical vector embeddings. These embeddings serve as the core feature for the semantic search functionality.

2. Model Selection and Comparison

The application selects and integrates four separate, pre-trained transformer models, each optimized for a distinct task. There is no explicit "baseline vs. advanced model" comparison within the code, as the models are chosen for their state-of-the-art performance in their respective domains:

| Task | Model Selected | Reasoning/Role in App |
| --- | --- | --- |
| Semantic Search | all-MiniLM-L6-v2 | Chosen for fast inference and good performance on semantic similarity tasks (vector feature creation). |
| Text Summarization | facebook/bart-large-cnn | A strong, established model for abstractive summarization of long documents (e.g., paper abstracts). |
| Question Answering | deepset/roberta-base-squad2 | Trained on the SQuAD 2.0 dataset, it's highly effective for extractive QA, providing answers directly from the provided context (papers). |
| Text Generation/Explanation | facebook/opt-350m | A smaller model from the OPT family, explicitly noted for faster inference while still providing coherent text generation for concept explanations. |

3. Visualizations and Insights (Plotly)

The application uses the Plotly library to provide interactive visualizations that offer insights into the loaded dataset.

📊 Dataset Visualizations

1. Top Paper Categories:
   * Visualization Type: Pie Chart .
   * Insight: Shows the distribution of the main category (e.g., cs.LG, cs.CV) across the loaded papers, highlighting the most active sub-fields within the Computer Science sample.
2. Key Concepts Frequency:
   * Visualization Type: Bar Chart .
   * Insight: Provides a visual representation of the top 10 most frequent non-stop words (key concepts) within a sample paper's abstract, quickly indicating the paper's central themes.

Dataset Statistics

The application also displays key quantitative metrics:

* Total Papers loaded.
* Unique Categories present in the sample.
* Average Abstract Length (in characters).
* Total Authors (sum of authors across all papers).

VISUAL OUTPUTS GRAPHS:



