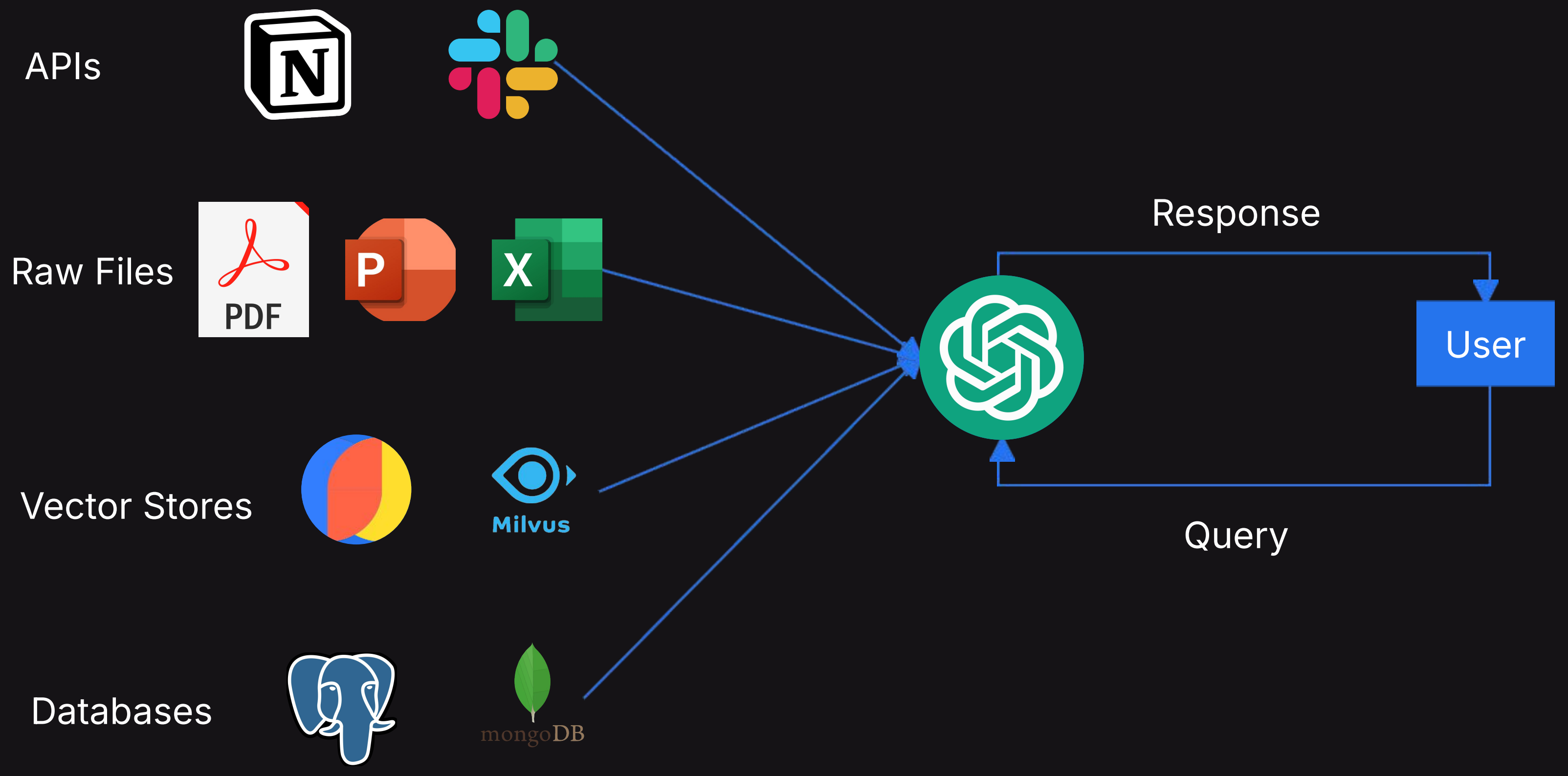
Building a Custom fine-tuned RAG System

Dipanjan (DJ) Sarkar

Analytics Vidhya

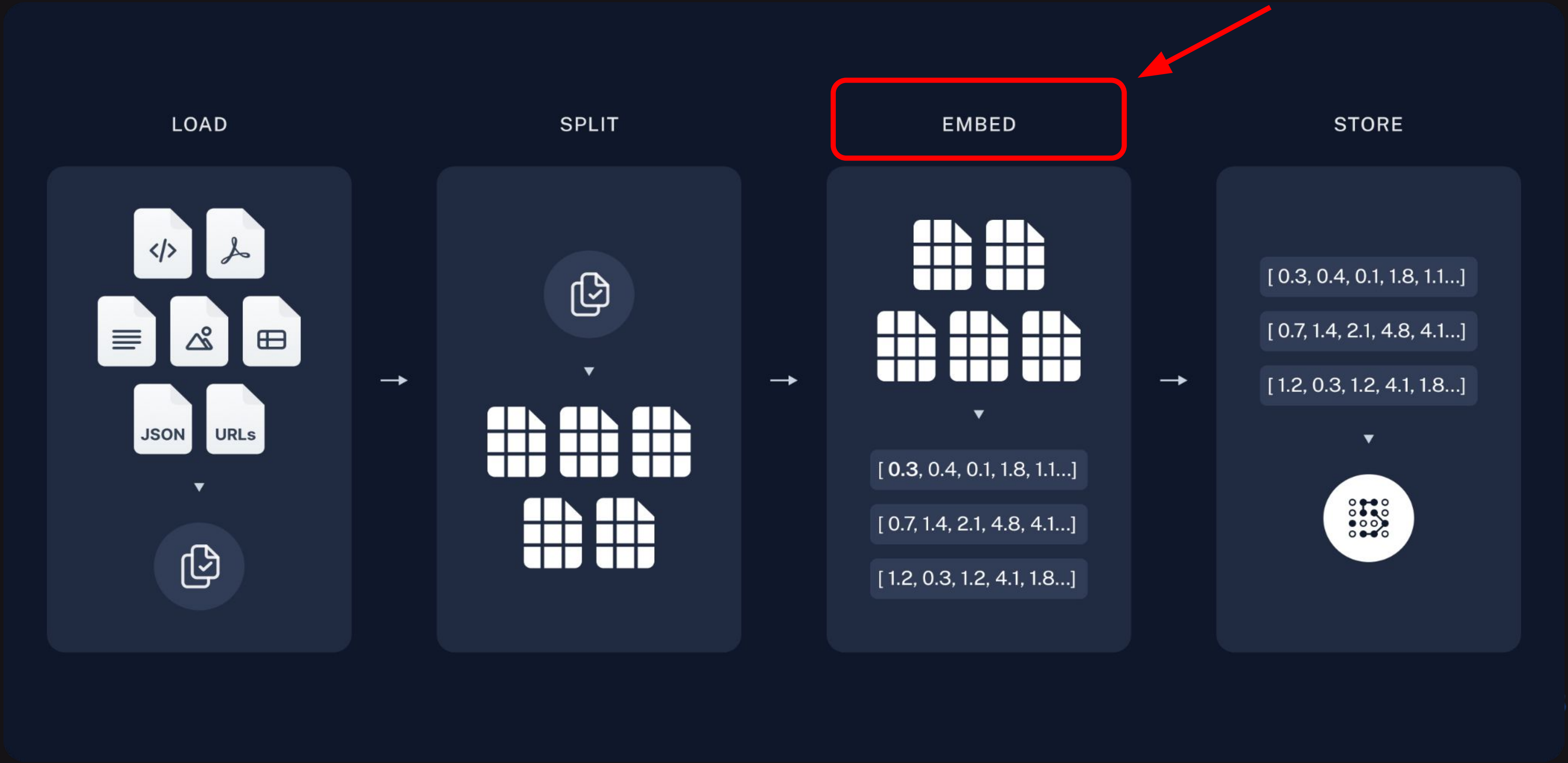
Understanding RAG Systems

What is a RAG System?

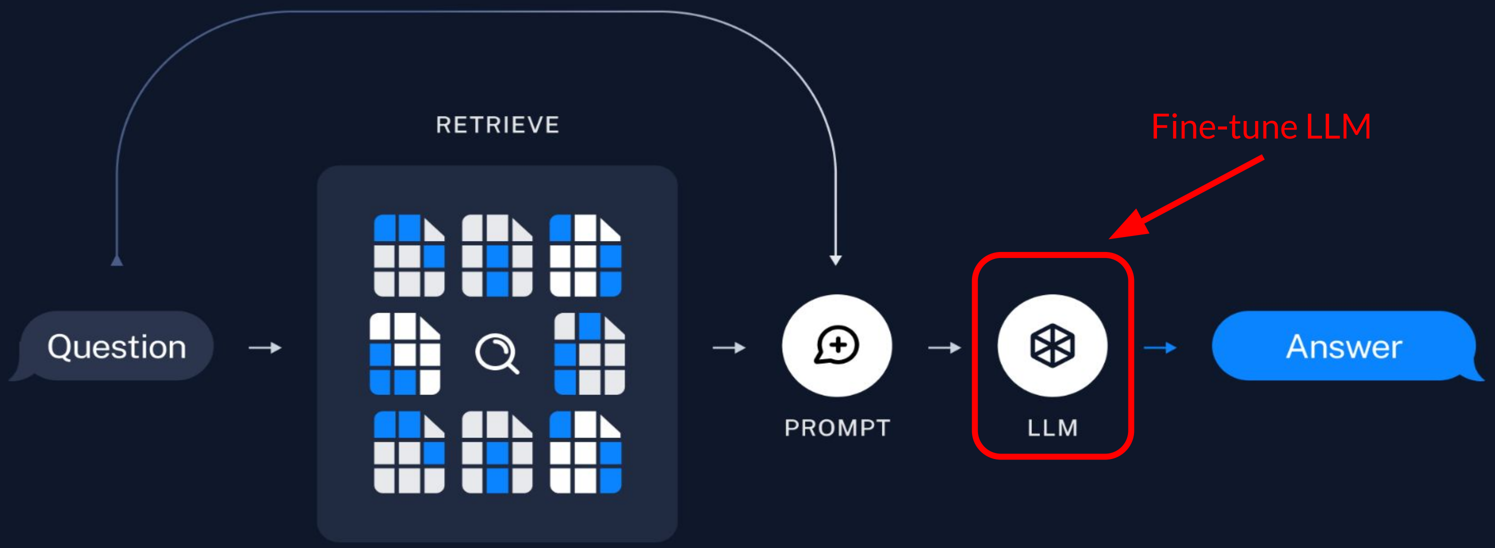


RAG System Architecture - Data Indexing

Fine-tune Embedder Model

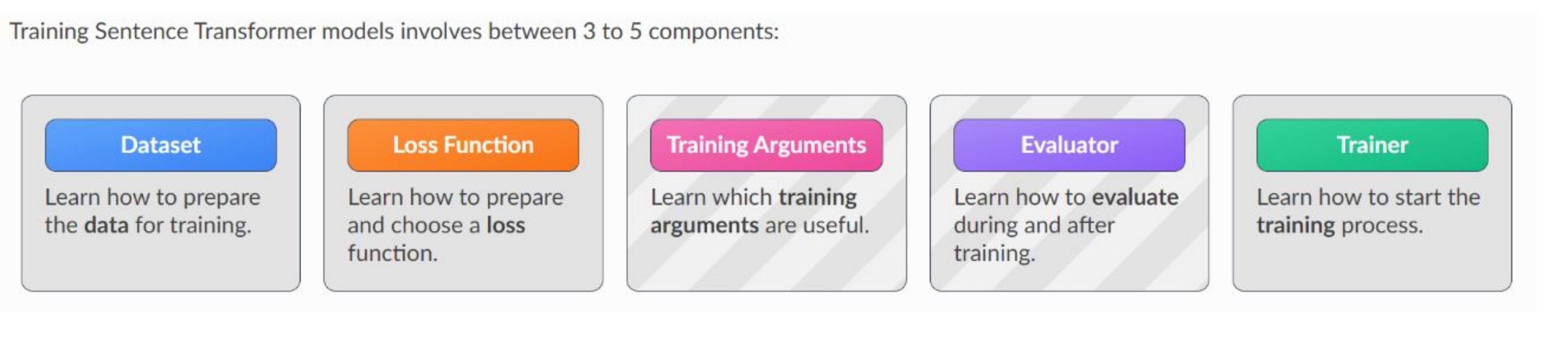


RAG System Architecture - Search and Generation



Fine-tune LLM

Fine-tuning Embedder Models



Training Sentence Transformer models involves between 3 to 5 components:

Dataset

Learn how to prepare the **data** for training.

Loss Function

Learn how to prepare and choose a **loss** function.

Training Arguments

Learn which **training arguments** are useful.

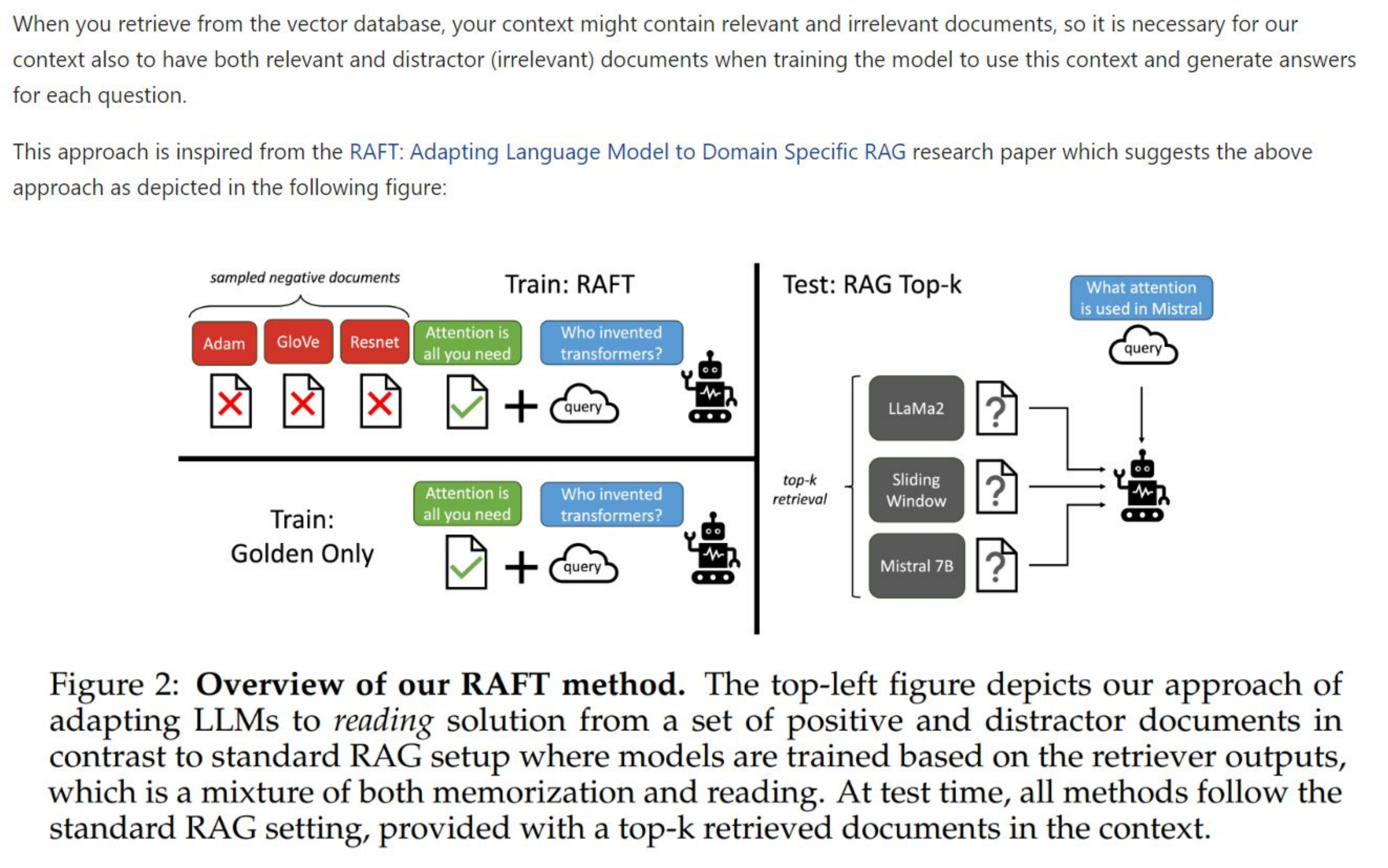
Evaluator

Learn how to **evaluate** during and after training.

Trainer

Learn how to start the **training** process.

Fine-tuning LLM for RAG - Inspired by RAFT



When you retrieve from the vector database, your context might contain relevant and irrelevant documents, so it is necessary for our context also to have both relevant and distractor (irrelevant) documents when training the model to use this context and generate answers for each question.

This approach is inspired from the RAFT: Adapting Language Model to Domain Specific RAG research paper which suggests the above approach as depicted in the following figure:

sampled negative documents
Train: RAFT
Adam
GloVe
Resnet
Attention is all you need
Who invented transformers?
query
Train: Golden Only
Attention is all you need
Who invented transformers?
query
Test: RAG Top-k
What attention is used in Mistral
query
top-k retrieval
LLaMa2
Sliding Window
Mistral 7B

Figure 2: **Overview of our RAFT method.** The top-left figure depicts our approach of adapting LLMs to *reading* solution from a set of positive and distractor documents in contrast to standard RAG setup where models are trained based on the retriever outputs, which is a mixture of both memorization and reading. At test time, all methods follow the standard RAG setting, provided with a top-k retrieved documents in the context.

Analytics Vidhya