Week 8 Hands-on RAG, Stable Diffusion, Web Applications

Report: Knowledge-Retrieval Chatbot (RAG-based Q&A System with Stable Diffusion & Web Deployment)

# Introduction

This project focuses on building a Retrieval-Augmented Generation (RAG)-based chatbot designed for job search assistance. It integrates document retrieval, language model-based response generation, image creation, and web-based deployment.

The chatbot helps users find job postings based on their queries and provides intelligent responses using a combination of retrieval and fine-tuned language models. The key components include:

* FAISS for efficient job posting retrieval
* Fine-tuned T5 model for answer generation
* Stable Diffusion for image generation
* Streamlit for a web-based user interface
* Cloudflare Tunnel for public deployment

By combining retrieval and generative AI, this project enhances job search capabilities by providing context- aware responses.

# System Architecture

The chatbot follows a multi-step pipeline for efficient information retrieval, processing, and response generation. The process consists of the following stages:

* 1. Data Preprocessing

The chatbot is designed to work with a job postings dataset, which contains fields like job title, company name, job description, and location.

* + - The data is cleaned, processed, and combined into a single text field.
    - The combined text is converted into numerical representations (embeddings) to allow efficient similarity search.

This ensures that queries related to job searches can be matched with relevant job postings.

* 1. Document Retrieval with FAISS

The FAISS (Facebook AI Similarity Search) library is used to enable fast and efficient retrieval of job postings based on user queries.

* + - The processed job descriptions are converted into numerical embeddings using a pre-trained Sentence Transformer model.
    - These embeddings are stored in a FAISS index, which allows quick nearest-neighbor search.
    - When a user submits a query (e.g., "Remote AI Engineer jobs"), the query is converted into an embedding, and FAISS finds the most relevant job postings.

This retrieval mechanism ensures that the chatbot returns accurate and relevant job listings in response to user queries.

* 1. Answer Generation with Fine-Tuned T5

Once job postings are retrieved, the system generates meaningful answers using a fine-tuned T5 model.

* + - T5 (Text-to-Text Transfer Transformer) is a powerful pre-trained language model that can be adapted for question-answering tasks.
    - The chatbot fine-tunes T5 using a custom job-related Q&A dataset, allowing it to generate context-aware answers.
    - The retrieved job descriptions serve as input context for T5, helping it generate detailed responses rather than just returning raw job descriptions.

By combining retrieval with generation, the chatbot provides rich, informative answers instead of just listing job postings.

* 1. Image Generation with Stable Diffusion

To enhance user experience, the chatbot also generates AI-created visuals related to job search queries using Stable Diffusion, a powerful text-to-image generation model.

* + - Based on the chatbot's response, a custom image prompt is created.
    - Stable Diffusion processes this prompt to generate job-related images (e.g., "A futuristic AI chatbot assisting job seekers").
    - These images help visually represent the chatbot’s responses, making the interaction more engaging. This multimodal approach makes the chatbot more interactive and visually appealing.
  1. Web Deployment with Streamlit

The chatbot is deployed using Streamlit, a popular Python-based web framework that enables interactive applications.

* + - Users can enter job search queries in a simple text input field.
    - The chatbot processes the query, retrieves relevant job listings, generates context-aware responses, and creates AI-generated visuals.
    - The results are displayed in an easy-to-read format on the web interface.

This allows users to search for jobs in real-time and receive dynamic AI-generated responses.

* 1. Cloudflare Tunnel for Public Access

Since Google Colab does not support direct hosting, the chatbot is made publicly accessible using Cloudflare Tunnel.

* + - Cloudflare Tunnel creates a secure public URL, allowing users to access the chatbot from anywhere on the internet.
    - Unlike ngrok, Cloudflare Tunnel does not require authentication, making deployment seamless. This ensures that users can interact with the chatbot without technical barriers.

# Implementation Details

The chatbot's end-to-end implementation follows these steps:

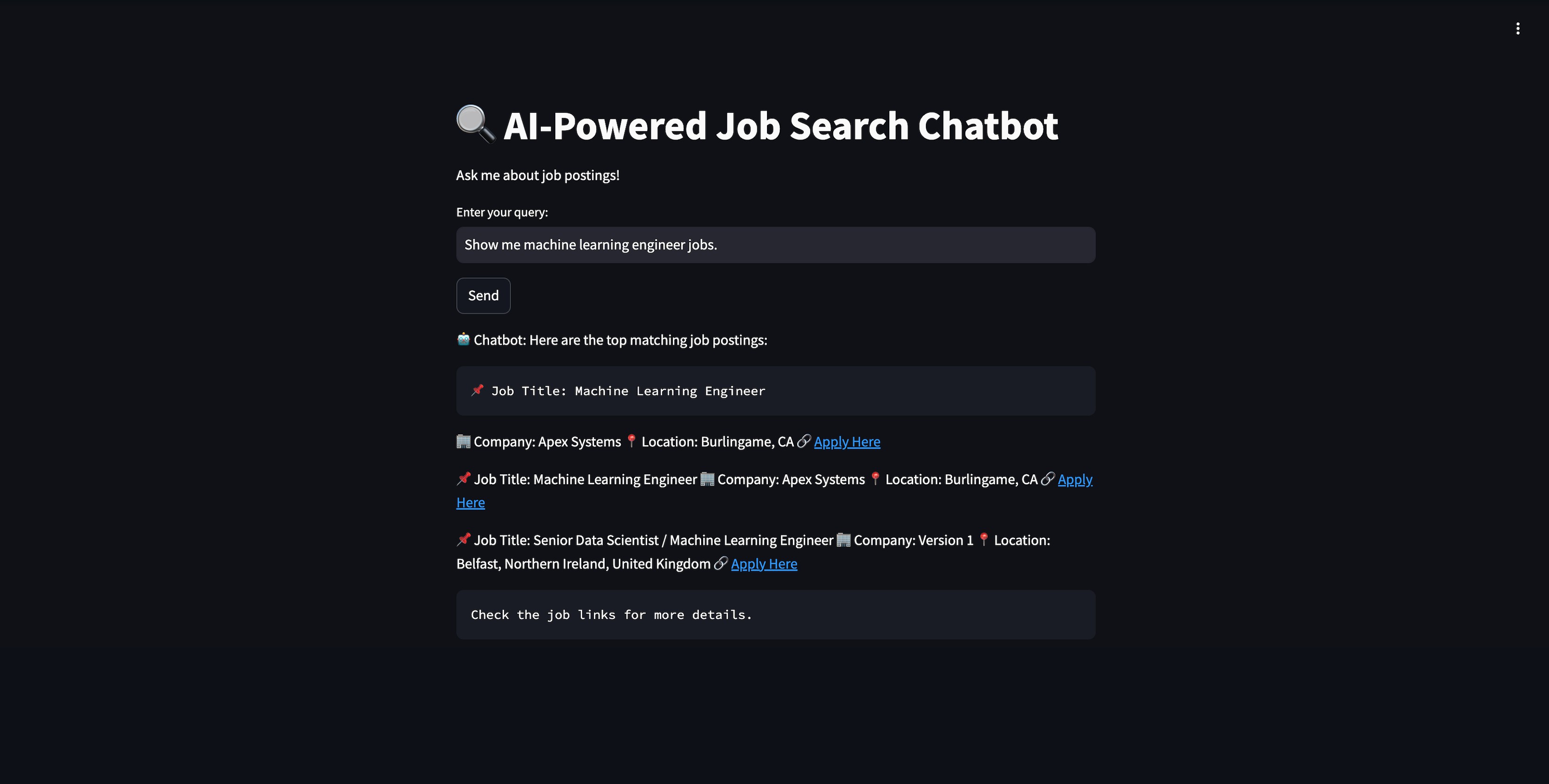
* 1. Data Preparation
     + Load and preprocess the job postings dataset.
     + Convert job descriptions into numerical embeddings for retrieval.
  2. FAISS Document Retrieval
     + Encode job postings using Sentence Transformer embeddings.
     + Store embeddings in a FAISS index for fast search.
     + Retrieve top-k most relevant job listings for any given query.
  3. Fine-Tuned T5 for Answer Generation
     + Train the T5-small model on job-related Q&A data.
     + Use retrieved job postings as context for T5.
     + Generate detailed answers instead of just returning job postings.
  4. Stable Diffusion for Image Generation
     + Convert chatbot responses into image generation prompts.
     + Generate job-related images using Stable Diffusion.
  5. Streamlit Web Deployment
     + Build a user-friendly interface for chatbot interaction.
     + Allow users to search for jobs in real-time.
  6. Public Deployment via Cloudflare
     + Use Cloudflare Tunnel to make the chatbot accessible online.

# Results & Performance

The chatbot successfully: Retrieves relevant job postings quickly using FAISS. Generates meaningful responses using a fine-tuned T5 model.

Creates AI-generated visuals for job-related queries. Provides an interactive user experience through Streamlit. Is publicly accessible via Cloudflare Tunnel.

**Output-**

****

# Challenges & Potential Improvements

Challenges Faced

1. Retrieval Accuracy:
   * FAISS sometimes retrieves less relevant job listings if embeddings are not optimized.
2. Model Generalization:
   * The T5 model sometimes generates generic responses instead of specific answers.
3. Slow Image Generation:
   * Stable Diffusion can be slow on CPU; requires GPU for real-time results.

Future Enhancements

* Improve retrieval by fine-tuning embeddings with domain-specific training.
* Train T5 on a larger dataset for better responses.
* Optimize Stable Diffusion to reduce processing time.
* Deploy the chatbot on Hugging Face Spaces or AWS for better scalability.

# Conclusion

This project successfully integrates RAG-based retrieval, LLM fine-tuning, and AI-generated images into a fully functional chatbot.

Key Achievements:

* Accurate job retrieval using FAISS
* Meaningful response generation with T5
* Visual representation with Stable Diffusion
* User-friendly web interface with Streamlit
* Public access via Cloudflare Tunnel

This chatbot serves as a foundation for AI-powered job search assistants and can be extended with more advanced retrieval and generative models.