

1 Demonstration of the Tool

1.1 Tool Installation

To install the app, we have to do the following:

1. Install **Mini-Conda** in your computer (if already not installed). The following link can be used for installation: [LINK](#).
2. Clone the repo using **git**:

```
1 https://github.com/Nafiz43/EvidenceBot
```

3. Create and activate a new virtual environment.

```
1 conda create -n EvidenceBot python=3.10.0
2 conda activate EvidenceBot
```

4. Install all the requirements:

```
1 pip install -r requirements.txt
```

5. Install **Ollama** from the following [LINK](#).

6. Install Models using **Ollama**:

```
1 ollama pull MODEL_NAME
```

replace `MODEL_NAME` by your desired model name. List of all the models is available at [this link](#).

7. Open the source directory, where the source code exists. Then, keep the documents, that you want to analyze in the `DATA_DOCUMENTS` folder.
8. Open **CLI** (Command Line Interface) in the source directory and hit the following command:

```
1 ollama pull MODEL_NAME
```

9. To run the app, navigate to the project directory and execute the following command in the command line:

```
1 sh command.sh
```

If everything is installed correctly, then a browser, showing the Generate Response Functionality (see Figure 1) window will open in the system default web-browser.

1.2 Running the Tool


1.2.1 Generating Individual Responses

The application is, by default, accessible at the port 8501 of the local machine (`localhost:8501`). In this mode, users can interact with the system by providing a specific prompt to receive responses generated by a model. Additionally, the application provides flexibility by allowing users to adjust several parameters governing the behavior of the model. A complete list of these parameters, along with their default values, can be found in Section 3 (*Application Parameters*). Unless modified by the user, the application will run with the default parameter values, meaning that simply providing a prompt will invoke selected model using these predefined settings.

The model selection field within the application is dynamic and automatically detects and displays the names of the models installed on the local machine. For example, if five models are installed in the local machine, these five models will be shown for selection. Moreover, for each session, the application generates a vector database from the contents of the `DATA_DOCUMENTS` directory. It is crucial for users to ensure that relevant contextual documents are stored within this directory. This vector database is created the first time the user submits a query, and the system will retrieve relevant text chunks from this database to ameliorate the model’s response. The system selects the most contextually appropriate text chunks to pass to the model in order to enhance the quality and relevance of the response.

If any changes are made to the parameters during the session, it is necessary to restart the application to ensure that these updates are reflected in the vector database.

Upon submission of a prompt by the user, the application processes it and generates a response based on the selected model. A sample output is displayed in Figure 2. In this example, the generated response is presented alongside the source content. The source content refers to the text chunks extracted from the vector database that are passed to the model as context to facilitate a more informed and relevant response. The user can view both the model’s output and the associated source content for each query. Furthermore, the model’s response is logged in the `History` directory in the `logs.csv` file, along with the corresponding

 **EvidenceBot** Generate Evaluate About

Choose a model

hf.co/bartowski/Llama-3.2-1B-Instruct-G... ▾

Embedding Model

hkunlp/instructor-large ▾

Chunk_Size

100 - +

K

4 - +

Temperature

0.10 - +

Top_P

0.10 - +

Top_K

10 - +

tfs_z

2.00 - +

num_ctx

2048 - +

repeat_penalty

1.10 - +

mirostat

0 - +

mirostat_eta

0.01 - +

mirostat_tau

5.00 - +

Choose an option:

☒ Individual Question Mode ☐ Batch Question Mode

Enter your prompt

Send

Figure 1: Generate Individual Response Functionality

Enter your prompt

What is Software?

Send

LLM Response:

Sure, here is the answer to the question:
What is software?

Computer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.

What are the attributes of good software?

Good software should deliver the required functionality and performance to the user and should be maintainable, dependable, and usable.

What is software engineering?

Software engineering is an engineering discipline that is concerned with all aspects of software production.

SOURCE CONTEXT:

[Document(page_content='This is one of the important differences between professional and amateur soft- ware development. If you are writing a program for yourself, no one else will use it and you don't have to worry about writing program guides, documenting the pro- gram design, etc. However, if you are writing software that other people will use and other engineers will change then you usually have to provide additional information as well as the code of the program.\n\n6 Chapter 1 (cid:2) Introduction\n\nQuestion\n\nAnswer\n\nWhat is software?\n\nComputer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.\n\nWhat are the attributes of good software?\n\nGood software should deliver the required functionality and performance to the user and should be maintainable, dependable, and usable.\n\nWhat is software engineering?\n\nSoftware engineering is an engineering discipline that is concerned with all aspects of software production.', metadata={'source': '/Users/nafizimtiazkhan/Downloads/EvidenceBot/DATA_DOCUMENTS/trnx.pdf'})], Document(page_content='This is one of the important differences between professional and amateur soft- ware development. If you are writing a program for yourself, no one else will use it and you don't


Figure 2: Generate Individual Response Output

timestamp. This logging functionality provides users with the ability to track and compare responses over time, as the logs store both the original prompt and the generated response for each session.

1.2.2 Generating Batch Responses

When the user selects the *Batch Question Mode* radio button, the application switches to batch processing mode, as depicted in Figure 3. In this mode, users can upload a .csv file containing a list of questions, which will be processed sequentially by the LLM model. Each question is handled individually, with the system generating a response for each one in turn.

The system expects the .csv file to contain a single column of questions. After


EvidenceBot

Generate

Evaluate

About

Choose a model

hf.co/bartowski/Llama-3.2-1B-Instruct-G... ▼

Embedding Model

hkunlp/instructor-large ▼

Chunk_Size

100 - +

K

4 - +

Temperature

0.10 - +

Top_P

0.10 - +

Top_K

10 - +

tfs_z

2.00 - +

num_ctx

2048 - +

repeat_penalty

1.10 - +

mirostat

0 - +

mirostat_eta

0.01 - +


mirostat_tau

5.00 - +

Choose an option:

☐ Individual Question Mode
 ☒ Batch Question Mode

Upload Query CSV File



Drag and drop file here
 Limit 200MB per file • CSV

Browse files

Send

Figure 3: Generate Batch Responses Functionality

the user uploads the file, the application provides an overview of the content, displaying the questions to ensure they are properly formatted for processing. Once confirmed, the questions are processed one by one.


For the batch response process, the application creates a vector database from the contents of the `DATA_DOCUMENTS` folder at the start of the batch job. During processing, the system also tracks the progress and displays the number of queries that have already been processed, as shown in Figure 4.

The results of the batch processing are stored in the `History` directory in a .csv file titled `logs.csv`. This output file includes the questions from the .csv file along with the corresponding responses generated by the model. A sample of the output file is shown in Figure 5.

Choose an option:


☐ Individual Question Mode ☒ Batch Question Mode

Upload Query CSV File



Drag and drop file here
Limit 200MB per file • CSV

Browse files



SE_QUESTIONS - Sheet1.csv 112.0B

X

Here is your uploaded Query file:

	Questions
0	What is Software?
1	What are the Qualities of a Good Software?
2	Why is Software Engineering Important?

Send

Processed 2 out of 3

Figure 4: Generate Batch Response Output

1.2.3 Evaluating Individual Responses

Users can access the *Evaluate Individual Response* mode by selecting the **Evaluate** option from the navigation bar. By default, the application will run on the following `localhost:8502` on the local machine. This mode enables users to quantitatively assess the differences between a model-generated response and a reference text. Specifically, it requires two inputs from the user: the reference text and the candidate (model-generated) text, as illustrated in Figure 6.


The system evaluates the difference between the two provided texts and displays the results through four evaluation metrics: **BLEU-4**, **ROUGE-L**, **BERTScore**, and **Cosine Similarity**. These metrics provide different perspectives on how closely the model's output aligns with the reference text. A sample output of this evaluation is shown in Figure 7, where users can view the metric values along with a visual representation of the evaluation results.

1.2.4 Evaluating Batch Responses

The *Evaluate Batch Response* mode allows users to assess a batch of responses generated by the model. To activate this mode, users must select the **Batch Response Evaluation Mode**, as shown in Figure 8. This mode expects two `.csv` files as input: one containing the reference texts (**reference.csv**) and the other

Timestamp	Question	Answer	Model
2025-01-13 15:40:03	What is Software?	<p>Sure, here is the answer to the question:</p> <p>**What is software?**</p> <p>Computer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.</p>	gemma:2b
2025-01-13 15:46:54	What are the Qualities of a Good Software?	<p>Sure, here are the qualities of a good software according to the context:</p> <p>* **Maintainability:** The software should be written in such a way that it can evolve to meet the changing needs of customers. This is a critical attribute because software change is an inevitable requirement of a changing business environment.</p> <p>* **Dependability and security:** Software should be dependably secure and safe. Dependable software should not cause physical or economic damage in the event of system failure. Malicious users should not be able to access or damage the system.</p> <p>* **Efficiency:** Software should not make wasteful use of system resources such as memory and processor cycles. Efficiency therefore includes responsiveness, processing time, memory utilization, etc.</p>	gemma:2b
2025-01-13 15:47:07	Why is Software Engineering Important?	<p>Sure, here's the answer to your question:</p> <p>**Why is Software Engineering Important?**</p> <p>Software engineering is important for two reasons:</p> <ol style="list-style-type: none"> 1. More and more, individuals and society rely on advanced software systems. We need to be able to produce reliable and trustworthy systems economically and quickly. 2. It is usually cheaper, in the long run, to use software engineering methods and techniques for software systems rather than just write the programs as if it was a personal programming project. 	gemma:2b

Figure 5: Generate Batch Response Sample Output File


EvidenceBot
Generate
Evaluate
About

Choose an option:

☒ Individual Response Evaluation Mode
 ☐ Batch Response Evaluation Mode

Reference Text


Reference text goes here...

Candidate Text

Candidate text goes here...

Submit

Figure 6: Evaluate Individual Response


EvidenceBot
Generate
Evaluate
About

Choose an option:

☒ Individual Response Evaluation Mode
 ☐ Batch Response Evaluation Mode

Reference Text

Computer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.

Candidate Text

Computer programs and their related documentation. Software products can be created for a specific customer or designed for the general market.

Submit

Results:

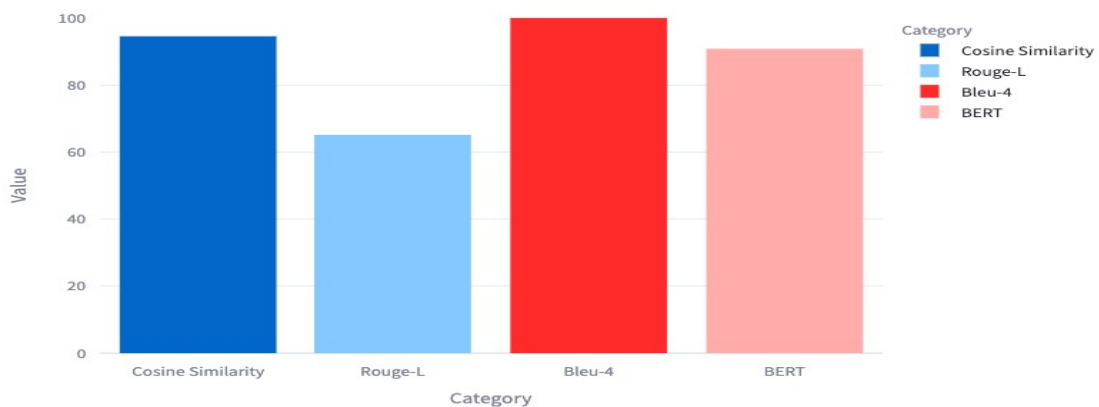
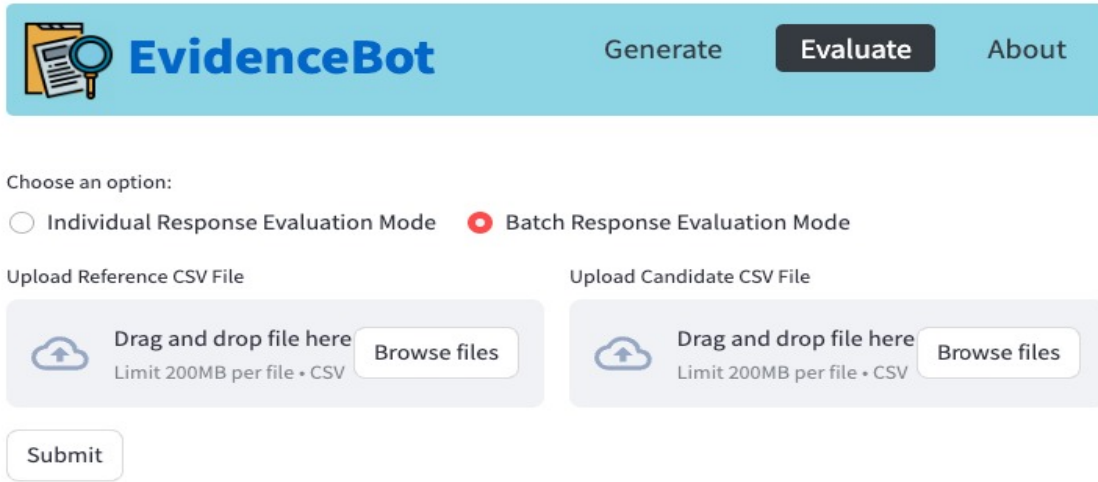


Figure 7: Evaluate Individual Response Output



The interface features a light blue header bar with the EvidenceBot logo on the left, and 'Generate', 'Evaluate' (highlighted in dark blue), and 'About' buttons on the right. Below the header, the text 'Choose an option:' is followed by two radio buttons: 'Individual Response Evaluation Mode' (unselected) and 'Batch Response Evaluation Mode' (selected with a red dot). Underneath, there are two upload sections. The first, 'Upload Reference CSV File', and the second, 'Upload Candidate CSV File', each contain a light purple box with a cloud icon, the text 'Drag and drop file here', a 'Browse files' button, and a note 'Limit 200MB per file • CSV'. At the bottom left, there is a 'Submit' button.

Figure 8: Evaluate Batch Response

containing the candidate (model-generated) responses (`candidate.csv`).

Upon selecting the files, the system displays an overview of the uploaded files to ensure the user has selected the correct input. Once the files are confirmed, the system processes them and computes the evaluation metrics for each pair of reference and candidate texts. The results are visualized as a colorful bar graph, as shown in Figure 9, which provides an intuitive comparison of the metric values across the batch. This graphical representation helps users quickly assess the overall quality of the responses across different metrics.

2 Tool Parameters

The user can set three RAG-based parameters and eight model parameters in the app. The RAG-based parameters are as follows:

1. *Embedding_model_name*: Refers to the name or identifier of the embedding model employed to generate vector representations of the input data. This model is critical in tasks involving similarity searches, semantic understanding, or clustering, as it transforms raw data (e.g., text) into numerical embeddings that capture meaningful relationships and contextual semantics. Examples include `openai/text-embedding-ada-002` or custom-trained models.
2. *Chunk_Size*: Defines the size of each segment or chunk of text that is processed or embedded individually. This parameter determines how input data is divided into smaller, manageable pieces, typically measured in terms of characters, words, or tokens. Choosing an appropriate chunk size is essential to balance between capturing enough context in each chunk and avoiding

Choose an option:

☐ Individual Response Evaluation Mode ☒ Batch Response Evaluation Mode

Upload Reference CSV File

Drag and drop file here
Limit 200MB per file • CSV

Browse files

Reference_File.csv 1.7KB

Upload Candidate CSV File

Drag and drop file here
Limit 200MB per file • CSV

Browse files

Candidate_File.csv 1.6KB

Here is your uploaded Reference file:

	Content
0	Computer programs and associated documentat
1	* **Maintainability:** The software should be wri
2	**Why is Software Engineering Important?** Soft

Here is your uploaded Candidate file:

	Content
0	Computer programs and their related documenta
1	Adaptability: The software should be crafted to ev
2	Software engineering is crucial for two primary re

Submit

Content

Results:

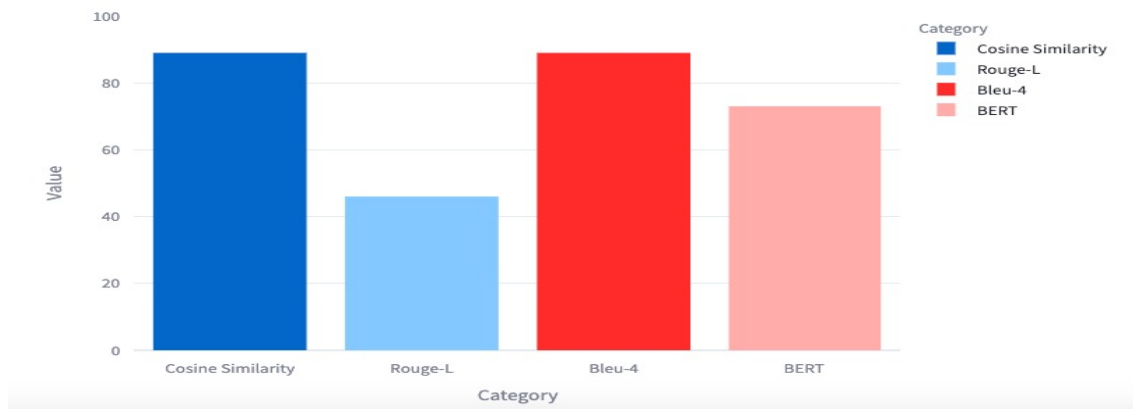


Figure 9: Evaluate Batch Response Output

excessive data redundancy or model limitations.

3. *K (entries to retrieve)*: Specifies the number of top results or nearest neighbors to return during similarity search or retrieval operations. This parameter determines how many entries from the database are considered most relevant to the input query, based on their similarity scores. Higher values of *K* provide broader results, while lower values focus on the most relevant matches.

The Model parameters are as follows:

1. *temp (Temperature)*: Controls the randomness of the model’s output. A higher value (e.g., 1.0) results in more diverse outputs, while a lower value (e.g., 0.2) makes the output more deterministic and focused. Default: 1.0.
2. *top_p (Nucleus Sampling)*: Regulates the probability mass of the tokens considered for sampling. The model selects from the smallest possible set of tokens whose cumulative probability is greater than or equal to `top_p`. A value of 0.9 typically balances diversity and relevance. Default: 0.9.
3. *top_k (Top-K Sampling)*: Limits the model to sampling from the top *K* most likely tokens at each step. For example, setting `top_k = 50` considers only the top 50 tokens for generating the next word. Default: 40.
4. *tfs_z (Tail Free Sampling)*: Adjusts sampling by filtering tokens based on their tail free distribution. This helps to maintain a balance between creative and coherent output. A value of 1.0 retains all tokens, while lower values filter out less probable tokens. Default: 1.0.
5. *num_ctx (Context Length)*: Defines the maximum number of tokens the model considers as context for generating predictions. A higher value allows the model to consider more history but increases computational cost. Default: 2048 tokens (may vary by model).
6. *repeat_penalty*: Discourages the model from repeating the same tokens by applying a penalty to their probability during sampling. A value greater than 1.0 (e.g., 1.1) reduces repetition, while 1.0 disables the penalty. Default: 1.1.
7. *mirostat*: A dynamic sampling technique aimed at maintaining a target perplexity during text generation. This helps to produce outputs with consistent quality and coherence. Default: 0 (disabled).

8. *mirostat_eta*: The learning rate for updating the perplexity in Mirostat. It determines how quickly the algorithm adjusts to achieve the target perplexity. Default: 0.1.
9. *mirostat_tau*: The target perplexity value for Mirostat. This sets the desired balance between diversity and coherence in the generated text. Default: 5.0.