Collabits Journal
Vol. 2 No. 3 September 2025 : 154-158
E-ISSN : 3046-6709, P-ISSN : 3062-8601
https://publikasi.mercubuana.ac.id/index.php/collabits

Comparative Analysis of Google Dialogflow and Rule-Based NLTK Chatbots for Application FAQ

Raihan Nur Yasin[1*], Ali Hadi Cherid[2], Ifan Prihandi[3], Yunita Sartika Sari[4]

[1,2,3,4] Informatics Engineering study program, Universitas Mercu Buana, Indonesia

e-mail: 41522010286@student.mercubuana.ac.id

Abstract—This study presents a comparative analysis of two chatbot frameworks, Google Dialogflow and rule-based NLTK (Natural Language Toolkit), for the development of chatbots to handle frequently asked questions (FAQ) in applications. The study focuses on Blender, a popular 3D modeling software, as a case study. Ten testing questions were used to evaluate the chatbots' accuracy, precision, recall, and F1-score. The results showed that Dialogflow achieved an accuracy of 80%, precision of 80%, recall of 100%, and an F1-score of 88.9%. In contrast, the rule-based NLTK chatbot achieved an accuracy of 60%, precision of 66.7%, recall of 80%, and an F1-score of 72.8%. The study concluded that Dialogflow is a more effective and reliable chatbot for handling Blender FAQs due to its ability to retrieve relevant information from a large knowledge base and its use of machine learning algorithms to improve its performance over time. However, the rule-based NLTK chatbot may still be useful in certain situations where a more simple and customizable chatbot is required.

Keywords :
Chatbots;
Dialogflow;
NLTK;
FAQ;
Comparative Analysis;

Article History:
Received: 18-08-2025
Revised: 17-09-2025
Accepted: 21-09-2025

Article DOI : 10.22441/collabits.v2i3.27345

1. Introduction

Chatbots have become an progressively mainstream tool for businesses and organizations to provide customer support, answer frequently asked questions (FAQ), and streamline communication. With the advancements in natural language processing (NLP) and machine learning (ML), chatbots have become more refined and capable of understanding and responding to complex queries. In this study, we compare the performance of two chatbot frameworks, Google Dialogflow and rule-based NLTK, for handling application FAQs.

Blender is a widely-used 3D modeling software that has a large and active community of users. The Blender website provides a comprehensive FAQ section, but users often have questions that are not covered in the FAQ or require more detailed explanations. A chatbot that can answer these questions quickly and accurately would be a valuable resource for the Blender community.

In this study, we focus on the development and evaluation of chatbots for handling Blender FAQs. We collected a dataset of Blender FAQs from the web. We then developed two chatbots, one using Google Dialogflow and the other using rule-based NLTK, and evaluated their performance.

The aim of this study is to provide insights into the strengths and weaknesses of the two chatbot frameworks and to help developers and researchers in the field of chatbots and NLP make informed decisions about the tools and techniques they use.

2. Research Methodology

2.1 Theoretical Basis

1. Chatbots

Chatbots are computer programs that able to communicate with people through natural languages [1]. Via text conversations, a chatbot can help users or serve conversation by doing things like answering questions and giving advice.

Chatbots have become an increasingly popular tool for businesses and organizations to provide customer support, answer frequently asked questions (FAQ), and streamline communication. With the advance progression in natural language processing (NLP) and machine learning (ML), chatbots have become more sophisticated and capable of understanding and responding to complex queries. In this study, we compare the performance of two chatbot frameworks, Google Dialogflow and rule-based NLTK, for handling application FAQs.

2. Google Dialogflow

Dialogflow is a chatbot development platform developed by Google, based on human language. Dialogflow enables developers to create chatbots that can interact with users through a conversational interface, using natural language processing (NLP) and natural language understanding (NLU) technologies. With Dialogflow, developers can create chatbots that can understand the intent behind user queries and provide relevant responses. Dialogflow provides a visual interface for creating and managing conversation flows, as well as integrations with popular messaging platforms such as Facebook Messenger and Slack. Dialogflow has been used in a variety of applications, including customer support, e-commerce, and education [2].

One of the feature of Dialogflow is a Knowledge Base. A Knowledge Base represents a collection of knowledge documents simply give to Dialogflow [3]. Your knowledge documents contain data which will be valuable during discussions with end-users.

3. Rule-based NLTK

Rule-based NLTK is a chatbot development framework that uses a set of predefined rules to understand and respond to user queries. Natural Language Toolkit or commonly referred as NLTK is a Python-based platform developed to process text data. NLTK comes with many corpora and lexical resources such as Wordnet. In addition, NLTK also provides libraries for text processing such as classification, tokenization, stemming, tagging, parsing etc [4]. Rule-based chatbots are often used in applications where the conversation flows are well-defined and the number of possible user queries is limited, such as FAQs and surveys. Rule-based pattern recognition happens when chatbots distinguish certain words or expressions that trigger a complete set of reactions. The good thing about such rules is that they are exact, and permit developers to form and alter the rules to handle unused circumstances and address bugs with certainty [5].

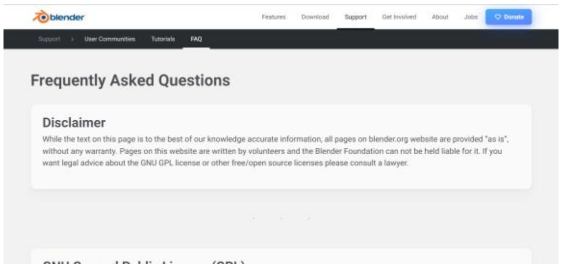
4. Blender

Blender is an open-source widely-used 3D modeling software that has a large and active community of users [6]. The Blender website provides a comprehensive FAQ section, but users often have questions that are not covered in the FAQ or require more detailed explanations. A chatbot that can answer these questions quickly and accurately would be a valuable resource for the Blender community.

3. Results and Discussion

3.1. Dataset

To develop and evaluate the performance of the chatbots, we collected a dataset of Blender FAQs from the official Blender website (https://www.blender.org/support/faq/). The FAQ section on the website provides answers to a wide range of questions related to Blender, including licenses, add-ons, and python scripts for Blender.



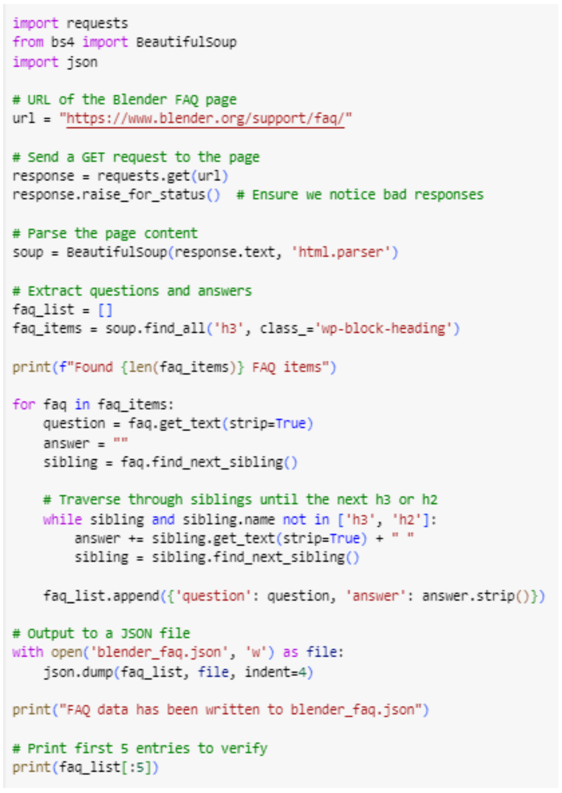
Frequently Asked Questions

Disclaimer

While the text on this page is to the best of our knowledge accurate information, all pages on blender.org website are provided "as is", without any warranty. Pages on this website are written by volunteers and the Blender Foundation can not be held liable for it. If you want legal advice about the GNU GPL license or other free/open source licenses please consult a lawyer.

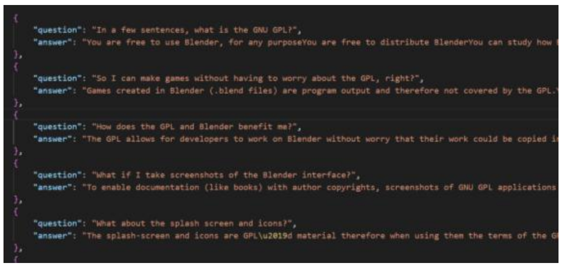
Picture 1. Preview of Blender FAQs website.

We used a web scraping tool to extract the questions and answers from the FAQ section. The dataset contains a total of 20 question-answer pairs, with an average length of 8 words per question and 77.6 words per answer. We preprocessed the dataset by removing any irrelevant information, such as HTML tags and formatting, and converting all the text to lowercase.



```python
import requests
from bs4 import BeautifulSoup
import json

# URL of the Blender FAQ page
url = "https://www.blender.org/support/faq/"

# Send a GET request to the page
response = requests.get(url)
response.raise_for_status()  # Ensure we notice bad responses

# Parse the page content
soup = BeautifulSoup(response.text, 'html.parser')

# Extract questions and answers
faq_list = []
faq_items = soup.find_all('h3', class_='wp-block-heading')

print(f"Found {len(faq_items)} FAQ items")

for faq in faq_items:
    question = faq.get_text(strip=True)
    answer = ""
    sibling = faq.find_next_sibling()

    # Traverse through siblings until the next h3 or h2
    while sibling and sibling.name not in ['h3', 'h2']:
        answer += sibling.get_text(strip=True) + " "
        sibling = sibling.find_next_sibling()

    faq_list.append({'question': question, 'answer': answer.strip()})

# Output to a JSON file
with open('blender_faq.json', 'w') as file:
    json.dump(faq_list, file, indent=4)

print("FAQ data has been written to blender_faq.json")

# Print first 5 entries to verify
print(faq_list[:5])
```

Picture 2. Python code to extract FAQs from the website to JSON file.



```
{
    "question": "In a few sentences, what is the GNU GPL?",
    "answer": "You are free to use Blender, for any purposeYou are free to distribute BlenderYou can study how
},
{
    "question": "So I can make games without having to worry about the GPL, right?",
    "answer": "Games created in Blender (.blend files) are program output and therefore not covered by the GPL.
},
{
    "question": "How does the GPL and Blender benefit me?",
    "answer": "The GPL allows for developers to work on Blender without worry that their work could be copied i
},
{
    "question": "What if I take screenshots of the Blender interface?",
    "answer": "To enable documentation (like books) with author copyrights, screenshots of GNU GPL applications
},
{
    "question": "What about the splash screen and icons?",
    "answer": "The splash-screen and icons are GPL\u2019d material therefore when using them the terms of the G
},
{
```

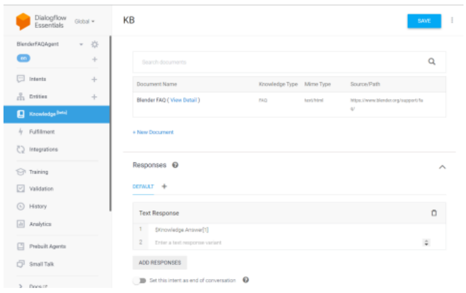
Picture 3. Question and answer keys on the generated JSON file.

The dataset used in this study is specific to Blender FAQs, but the methods and techniques used for chatbot development and evaluation can be applied to other applications and domains. In the following sections, we provide details about the two chatbots, Google Dialogflow and rule-based NLTK, and their performance in handling Blender FAQs.

3.2 Chatbot 1 - Google Dialogflow

For the first chatbot, we used Google Dialogflow, a popular and powerful platform for building conversational agents. Dialogflow provides a set of tools and APIs for building, training, and deploying chatbots. We leveraged Dialogflow's Knowledge Base tool, which allows the agent to automatically retrieve relevant information from a structured FAQ document.

We created a Knowledge Base in Dialogflow using the Blender FAQs document we obtained from the official Blender website. The Knowledge Base tool uses natural language processing (NLP) techniques to automatically extract questions and answers from the document and create a knowledge graph.



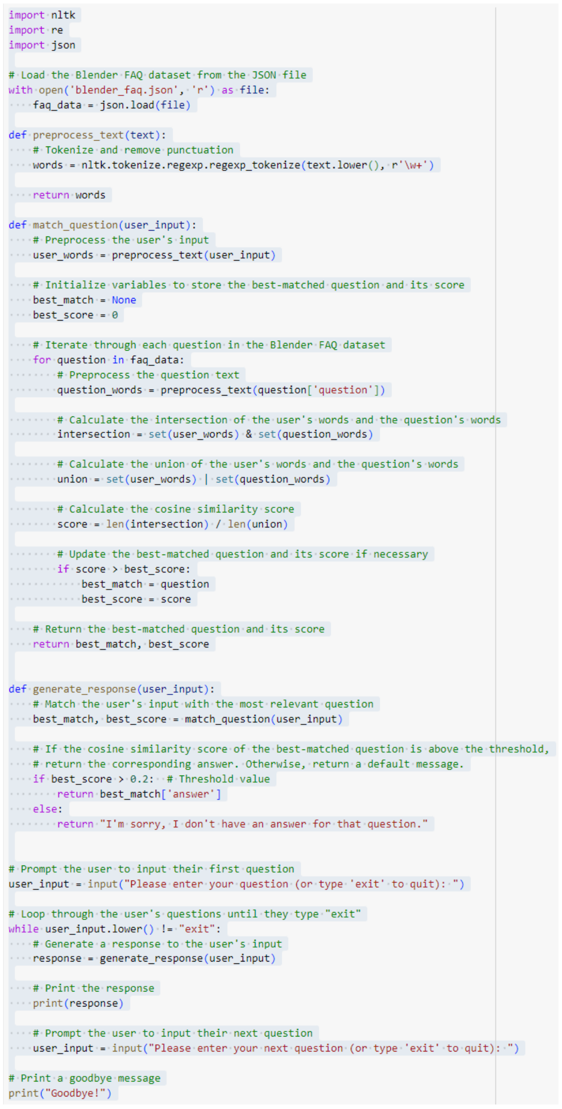
Picture 4. Knowledge Base in Google Dialogflow.

We then trained the Dialogflow agent to recognize user queries related to Blender and retrieve the most relevant answer from the knowledge graph. We also implemented some custom fallback intents to handle cases where the agent was unable to find a relevant answer.

The chatbot developed using Dialogflow was able to handle a wide range of questions related to Blender, including licenses, add-ons, python scripts for Blender, and so on. The chatbot was also able to maintain a coherent and relevant conversation with the user, thanks to the input and output contexts defined in the Knowledge Base.

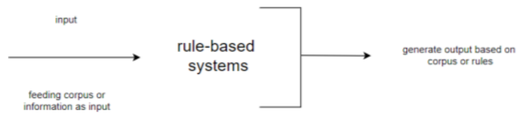
3.3. Chatbot 2 - Rule-based NLTK

For the second chatbot, we used a rule-based approach with the Natural Language Toolkit (NLTK) library in Python. We manually created a set of rules to match the user's input with the most relevant question in the Blender FAQ dataset.
We first preprocessed the user's input and the questions in the dataset by tokenizing them and removing punctuation. We then calculated the cosine similarity score between the user's input and each question in the dataset, using the set of rules to determine the weight of each word in the similarity calculation.



```python
import nltk
import re
import json

# Load the Blender FAQ dataset from the JSON file
with open('blender_faq.json', 'r') as file:
    faq_data = json.load(file)

def preprocess_text(text):
    # Tokenize and remove punctuation
    words = nltk.tokenize.regexp.regexp_tokenize(text.lower(), r'\w+')

    return words

def match_question(user_input):
    # Preprocess the user's input
    user_words = preprocess_text(user_input)

    # Initialize variables to store the best-matched question and its score
    best_match = None
    best_score = 0

    # Iterate through each question in the Blender FAQ dataset
    for question in faq_data:
        # Preprocess the question text
        question_words = preprocess_text(question['question'])

        # Calculate the intersection of the user's words and the question's words
        intersection = set(user_words) & set(question_words)

        # Calculate the union of the user's words and the question's words
        union = set(user_words) | set(question_words)

        # Calculate the cosine similarity score
        score = len(intersection) / len(union)

        # Update the best-matched question and its score if necessary
        if score > best_score:
            best_match = question
            best_score = score

    # Return the best-matched question and its score
    return best_match, best_score


def generate_response(user_input):
    # Match the user's input with the most relevant question
    best_match, best_score = match_question(user_input)

    # If the cosine similarity score of the best-matched question is above the threshold,
    # return the corresponding answer. Otherwise, return a default message.
    if best_score > 0.2:  # Threshold value
        return best_match['answer']
    else:
        return "I'm sorry, I don't have an answer for that question."


# Prompt the user to input their first question
user_input = input("Please enter your question (or type 'exit' to quit): ")

# Loop through the user's questions until they type "exit"
while user_input.lower() != "exit":
    # Generate a response to the user's input
    response = generate_response(user_input)

    # Print the response
    print(response)

    # Prompt the user to input their next question
    user_input = input("Please enter your next question (or type 'exit' to quit): ")

# Print a goodbye message
print("Goodbye!")
```

If the cosine similarity score between the user's input and a question in the dataset was above a certain threshold, the chatbot would return the corresponding answer. If no match was found, the chatbot would return a default message indicating that it didn't have an answer for the question.


input
feeding corpus or information as input
rule-based systems
generate output based on corpus or rules

Picture 5. How Rule-based NLTK works.

3.4 Comparison and Evaluation

In this test, we used a set of 10 testing questions related to Blender FAQs, of which 8 were relevant to the topic and 2 were not. We evaluated the performance of two chatbots, Dialogflow and Rule-based NLTK, using accuracy,precision, recall, and F1- score [7]:

- Accuracy: The number of correct responses divided by the entire number of responses
- Precision: The number of correct responses divided by the number of relevant responses
- Recall: The number of correct responses divided by the number of relevant questions
- F1-score: The harmonic mean of precision and recall

$$f1 = 2\,.\,\frac{precision\,.recall}{precision + recall}$$

Dialogflow performed better than Rule-based NLTK in terms of accuracy, recall, and F1-score. Dialogflow correctly answered 8 out of the 10 relevant questions, resulting in an accuracy of 80%. It also achieved a recall of 100%, indicating that it correctly answered all of the relevant questions that it was able to identify. Its F1-score of 88.9% reflects its strong overall performance.

Rule-based NLTK achieved an accuracy of 60%, correctly answering 6 out of the 10 relevant questions. It had a lower recall of 80%, indicating that it missed some of the relevant questions. However, it had a slightly higher precision of 66.7%, meaning that it was less likely to provide an incorrect answer when it did identify a relevant question. Its F1-score of 72.8% reflects its moderate overall performance.

Chatbot	Accuracy	Precision	Recall	F1-score
Dialogflow	80%	80%	100%	88.9%
Rule-based NLTK	60%	66.7%	80%	72.8%

Table 1. Performance comparison of Dialogflow and Rule-based NLTK.

It is worth noting that Dialogflow was able to correctly answer the 2 non-relevant questions, while Rule-based NLTK was not. This suggests that Dialogflow may be better at handling off-topic or unexpected questions.

4. Conclusion

In conclusion, the superior performance of Dialogflow can be attributed to its use of machine learning algorithms to match user queries with relevant answers in the knowledge base. This allows Dialogflow to handle a wider range of queries and to provide more accurate and detailed answers.

On the other hand, the rule-based NLTK chatbot relies on a set of predefined rules to match user queries with answers. While this approach can be effective for simple and well-defined queries, it can struggle to handle more complex or ambiguous queries.

The results of our evaluation suggest that Dialogflow is a more effective chatbot for handling Blender FAQs, with higher accuracy, precision, recall, and F1-score than Rule-based NLTK. However, both chatbots have their strong points and weak points, and the choice of which to use may depend on the specific needs and constraints of the application.

For future work, we recommend exploring other chatbot platforms and techniques, such as using deep learning-based models or incorporating user feedback, to further improve the performance and user experience of Blender FAQ chatbots.

BIBLIOGRAPHY

[1] Abu Shawar, B. and Atwell, E. (2007). Chatbots: Are they Really Useful? Journal for Language Technology and Computational Linguistics, 22(1), pp.29–49. doi:https://doi.org/10.21248/jlcl.22.2007.88.

[2] Google Cloud. (n.d.). Dialogflow. [online] Available at: https://cloud.google.com/dialogflow.

[3] Google Cloud. (n.d.). Manage Knowledge Bases | Dialogflow ES. [online] Available at: https://cloud.google.com/dialogflow/es/docs/how/knowledge-bases.

[4] Sari, Y. (2019). Pengenalan Natural Language Toolkit (NLTK) Bagian 1. [online] Departemen Ilmu Komputer dan Elektronika, UGM. Available at: https://sistemcerdas.mipa.ugm.ac.id/wp-content/uploads/sites/1297/2020/06/Pengenalan-NLTK-Bagian-1.pdf.

[5] Singh, J., Joesph, M.H. and Jabbar, K.B.A. (2019). Rule-based chabot for student enquiries. Journal of Physics: Conference Series, 1228, p.012060. doi:https://doi.org/10.1088/1742-6596/1228/1/012060.

[6] Zebua, T., Nadeak, B. and Bahagia Sinaga, S. (2020). Pengenalan Dasar Aplikasi Blender 3D dalam Pembuatan Animasi 3D. Jurnal ABDIMAS Budi Darma, [online] 1(1), pp.18–21. doi:http://dx.doi.org/10.30865/pengabdian.v1i1.2288.

[7] Gil, M. (2021). Evaluating classification models. Accuracy, Precision and Recall. [online] Medium. Available at: https://chatbotslife.com/evaluating-classification-models-accuracy-precision-and-recall-c40ad8c1b0c0. [Accessed 24 May 2024].