Enhancing e-learning with efficient chatbot: an appropriate model for optimized student support

KOUAME Appoh[1], KRA Lagasane[2], Beman Hamidja KAMAGATE[3]
[1]*Department of Computer Science, Institut National Polytechnique Felix Houphouët Boigny (INPHB), Côte d'Ivoire.*
[2]*Department of Computer Science, Université Alassane Ouattara(UAO), Côte d'Ivoire*
[3]*Laboratoire des Sciences, des Technologies de l'Information et de la Communication (LASTIC)/Ecole Supérieure Africaine des TIC(ESATIC), Côte d'Ivoire.*

ABSTRACT: *E-learning platforms offer significant value by providing students with the flexibility to access course materials anytime and anywhere, eliminating geographical and time barriers. However, the lack of face-to-face interaction can hinder essential social connections and group dynamics for collaborative learning, often leading to feelings of isolation. This work addresses these challenges by proposing a chatbot based on DistilBERT, designed to assist students with their queries. Optimized for low-memory and low-power devices, the chatbot is accessible even on less powerful devices, making it suitable for a wide range of users. The study evaluates three models: a rule-based chatbot, BERT, and DistilBERT, focusing on three criteria: Bot Understanding Degree (BUD), memory usage, and response time. BERT, though highly accurate, requires significant memory (60%) and a slower response time of 600 ms, which may limit its practicality for resource-constrained environments. DistilBERT, in contrast, offers an efficient balance by using less memory (25%) and providing faster response times (250 ms), while still generating context-aware responses—a feature that rule-based models lack. The rule-based model, with minimal memory usage (15%) and a fast response time (under 50 ms), is best suited for simple interactions. Overall, DistilBERT is ideal for e-learning, where students often use mobile devices, balancing accuracy, speed, and resource efficiency effectively.*

KEYWORDS - *Chatbot, NLP, BERT, DistilBERT, e-learning*

I. INTRODUCTION

The rise of the Internet has significantly transformed the education sector, notably with the emergence of e-learning. E-learning allows students to access courses resources anytime and from anywhere, thus eliminating geographical and temporal constraints. This is particularly beneficial for individuals living in remote areas, those with time constraints, or those needing to adhere to health restrictions, as was the case during the COVID-19 pandemic. Moreover, e-learning offers great flexibility, enabling learners to progress at their own pace and organize their schedules according to their needs [1].

Although e-learning has many advantages, it also has certain limitations. Technical issues, such as system failures or difficulties with online platforms, can disrupt the learning experience. The lack of face-to-face interaction can restrict social exchanges and group dynamics, which are essential for collaborative learning and building personal connections. The isolation experienced by learners in the absence of direct contact with peers and instructors can negatively impact their engagement, sense of belonging and can influence the effectiveness of learning [2].

To overcome these challenges and enhance the online learning experience to reduce dropout rates, the use of chatbots as educational support tools and for the immediate management of learner concerns has become a necessity. That is why we explore how

a chatbot, as an artificial intelligence program and Human-Machine Interaction (HMI) tool designed to simulate conversations with human users on the Internet, can enhance the online learning experience. However, chatbots face limitations that hinder their effectiveness. Firstly, there is the issue of understanding user requests. Chatbots often struggle to accurately interpret expressed intentions, which can lead to inappropriate or irrelevant responses. This difficulty stems from the complexity of human language, which is rich in ambiguities, synonyms, and unexpected phrasing. Furthermore, understanding a request often depends on the context of the conversation and the exchange history. Chatbots lacking conversational memory are therefore more likely to miss essential elements, reducing the relevance of their responses. Secondly, handling questions outside their knowledge base presents another critical challenge. When a user asks a question for which no pre-recorded answer is available, the chatbot may fail to provide a useful or appropriate response. This can not only cause frustration but also undermine users' trust in these tools.

These limitations underscore the importance of improving chatbots' capabilities, particularly by enhancing their ability to understand natural language and adapt to unforeseen situations. Such improvements are essential to optimize their utility and meet users' growing expectations in a smarter and more effective way.

II. MATERIALS AND METHODS

2.1. A transformer-based chatbot

A transformer is a type of neural network architecture designed to process sequential data, such as natural language [7]. The Transformer architecture is particularly known for its ability to handle long-range dependencies in data using a mechanism called self-attention, without the need for recurrent structures or convolution, which were common in earlier models like RNNs (Recurrent Neural Networks) and CNNs (Convolutional Neural Networks).

The architecture of a transformer model, commonly used in natural language processing tasks as shows in figure 1 consists of two main parts. The encoder, shown on the left of figure 1, processes the input sequence by converting each token into an embedding and adding positional encoding to retain token order. It consists of multiple identical layers (N ×), each with a multi-head attention mechanism for focusing on different parts of the sequence, an Add & Norm step for residual connections and normalization, and a feed-forward layer for further processing. Stacking these layers builds a progressively richer representation of the input.

The decoder, shown on the right of figure 1, generates the output sequence using information from the encoder. Each output token is first converted into an embedding. A masked multi-head attention mechanism, similar to the encoder's, prevents the decoder from accessing future tokens during training. The decoder also includes a multi-head attention layer to focus on the encoder's output, with "Add & Norm" layers providing residual connections and normalization after each attention and feed-forward layer. The decoder stack contains multiple identical layers (N ×), followed by a linear layer and a SoftMax activation that assigns probabilities to each potential output token, enabling the generation of the final sequence.

The transformers have introduced a series of specialized models, each adapted to specific needs in natural language processing. Among the most well-known are BERT, GPT, and T5[8]. BERT is an encoder-based model designed to excel in text comprehension tasks by leveraging a bidirectional analysis of word context. In this work, we propose using DistilBERT, a streamlined variant of BERT that is optimized for greater efficiency in resource consumption, making it well-suited for resource-constrained environments and mobile deployment.

2.2. A DistilBERT

DistilBERT is a distilled version of BERT, designed to be smaller and faster while retaining most of BERT's performance. The BERT (Base) model and its lighter version, DistilBERT, differ mainly in the number of transformer layers used, which impacts the parameter size and processing capacity.

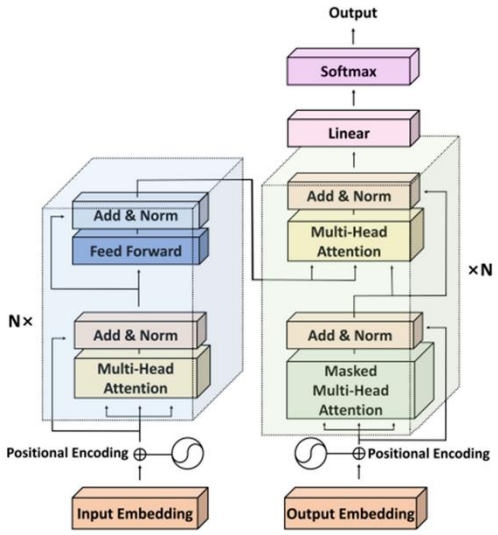

Figure 1: **Transformer structure [8]**

BERT (Base) uses 12 transformer layers to encode data, while DistilBERT uses only 6, representing a 50% reduction in the number of layers [9]. In terms of configuration, both models share similar parameters. For example, each layer in BERT and DistilBERT uses 12 attention heads, thus providing the same level of parallelism in information processing. The hidden size of each model is also the same, with 768-dimensional representations, and the number of neurons in the feed-forward layers is set to 3072. However, due to the reduced number of layers, the DistilBERT model is lighter, with a total of 66 million parameters, which is 40% less than BERT (Base)'s 110 million parameters. This reduction in parameter size makes DistilBERT faster and less resource-intensive while maintaining performance close to that of BERT for many natural language processing applications.

BERT (Base) and DistilBERT use different pre-training tasks to learn language representations. BERT is trained through two main tasks: Masked Language Modeling (MLM), where certain words in a sentence are masked so that the model learns to predict them, and Next Sentence Prediction (NSP), which aims to evaluate if one sentence logically follows another. In contrast, DistilBERT only uses Masked Language Modeling (MLM) and does not include Next Sentence Prediction (NSP)[10].

Another key distinction lies in the knowledge distillation method. BERT (Base) does not use knowledge distillation and follows a standard pre-training. In contrast, DistilBERT applies knowledge

2.3. Proposal framework

Figure 2 presents a flowchart that demonstrates the process of handling user input and generating a response using the DistilBERT transformer model. The process begins by capturing user input via the getUserInput () function, which collects the text for further processing.

Step 1: User Input

```
FUNCTION getUserInput()
    INPUT user_input
    RETURN user_input
END FUNCTION
```

Step 2: Preprocessing

```
FUNCTION preprocessInput(user_input)
 CLEAN user_input (remove special characters, lower case, etc.)
    TOKENIZE user_input
    RETURN tokenized_input
END FUNCTION
```

Step 3: DistilBERT Model for Context Understanding

```
FUNCTION getContextualRepresentation(tokenized_input)
    DISTILBERT_OUTPUT = DistilBERT_Model(tokenized_input)
    RETURN DISTILBERT_OUTPUT
END FUNCTION
```

Step 4: Intent/Entity Recognition

```
FUNCTION recognizeIntentAndEntities(DISTILBERT_OUTPUT)
    INTENT = classifyIntent(DISTILBERT_OUTPUT)
    ENTITIES = extractEntities(DISTILBERT_OUTPUT)
    RETURN INTENT, ENTITIES
END FUNCTION
```

Step 5: Response Generation

```
FUNCTION generateResponse(INTENT, ENTITIES, knowledge_base)
```

```
    IF INTENT == "question" THEN
      RESPONSE =
fetchResponseFromKnowledgeBase(ENTITIES,
knowledge_base)
    ELSE
      RESPONSE =
generateConversationalResponse(INTENT)
    END IF
    RETURN RESPONSE
  END FUNCTION
```

Step 6: Output to User

```
  FUNCTION sendResponseToUser
(RESPONSE)
    DISPLAY RESPONSE to user_interface
  END FUNCTION
```

Step 7: Feedback Loop (Optional)

```
  FUNCTION collectFeedback()
    INPUT user_feedback
    IF feedback_enabled THEN

updateModelWithFeedback(DISTILBERT_MODE
L, user_feedback)
    END IF
  END FUNCTION
```

Figure 2: Flowchart of User Interaction and Response Generation Process Using DistilBERT

Next, the function preprocessInput(user_input) cleans and standardizes the input by removing special characters, converting text to lowercase, and tokenizing it into individual units to make it suitable for model analysis. After that process, the function getContextualRepresentation(tokenized_input) then passes the tokenized input through DistilBERT to analyze the context, producing DISTILBERT_OUTPUT.

This output is sent to recognizeIntentAndEntities (DISTILBERT_OUTPUT), which determines the user's intent (INTENT) and extracts any relevant entities (ENTITIES), both essential for generating an accurate response. In the response generation stage, generate Response (INTENT, ENTITIES, knowledgebase) formulates a reply based on the intent and entities. If the intent is a question, it retrieves the appropriate answer from a knowledge base; otherwise, it generates a conversational reply. The response is then displayed to the user by sendResponseToUser (RESPONSE).

The chatbot framework in the figure 3 consists of multiple components that work together to process and respond to user requests. First with the User Interface (Mobile/Web): The user initiates a request through a mobile or web interface. This interface collects the user's input and forwards it to the chatbot for processing. Once the user request reaches the chatbot, it first undergoes preprocessing. This stage prepares the input for analysis by cleaning the text, removing unnecessary characters, standardizing the format (like converting text to lowercase), and tokenizing the input into manageable units. This preprocessing step helps optimize the request for accurate understanding by the chatbot engine.

The chatbot engine, leverages the DistilBERT-based model from the previous flowchart (figure 2) to interpret the user's input and generate a relevant response. The engine performs several steps. After receiving the tokenized input, the chatbot engine uses DistilBERT to analyze the context of the request, creating a contextual representation of the input. This step helps the engine understand the user's intent and identify any entities mentioned in the input. Using the contextual representation, the engine determines the user's intent (e.g., question, command) and extracts any relevant entities. This classification is essential for generating a precise response. Based on the identified intent and entities, the chatbot engine formulates a response. If the intent suggests the user is asking a question, the engine queries the knowledge sources, such as the Frequently Asked Questions (FAQ) database and documents, to retrieve an accurate answer. For other intents, it generates an appropriate conversational reply. The FAQ database likely contains common questions and answers, enabling quick responses. The documents database provides more in-depth information for complex or less frequently asked questions. Once the chatbot engine formulates a response, it sends it back through the user interface. The response is displayed to the user, completing the interaction. As shown in the previous flowchart, the chatbot may include a feedback loop. This allows it to collect user feedback.

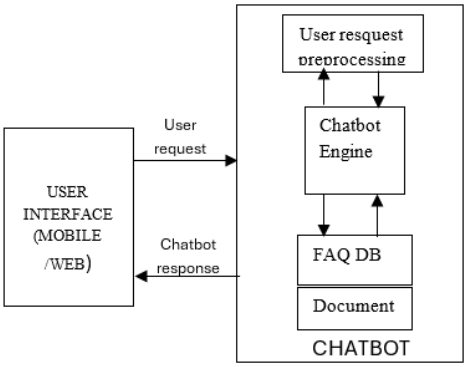

Figure 3: Chatbot framework

2.4. Dataset

We developed a question-answer corpus based on an algorithm course material intended for first-year undergraduate students. This corpus, generated with the help of ChatGPT-4, contains 284 questions along with their answers. Subsequently, spelling and grammar errors were intentionally introduced in some questions. The corpus was then exported in JSON format. Here is an extract of json file:

{ "faq": [
{
"question": "Qu'est-ce qu'un algoritme?",
"answer": "Un algorithme est une série d'instructions ou d'étapes précises à suivre pour résoudre un problème ou effectuer une tâche."
},
{
"question": "Qu'est-ce qu'un algorithme de tri?",
"answer": "Un algorithme de tri est un processus qui organise les éléments d'une liste ou d'un tableau dans un ordre spécifique, comme l'ordre croissant ou décroissant."
},
{
"question": "Quels sont les prerequis pour comprendre un court d'algorithmes?",
"answer": "Les prérequis incluent une compréhension de base des structures de données comme les tableaux et les listes, ainsi que des notions élémentaires de mathématiques et de logique."
}
]
}

III. Result

For implementing our models, we used an HP Core i7 computer with 16 GB of memory and a 64-bit operating system. We employed Python 3 and NLP libraries: Transformers, SpaCy and others. We simulated three models on the same corpus: a rule-based NLP model using keywords, the BERT model, and the DistilBERT model. After training and testing the performance of BERT and DistilBERT, we evaluated the ability of the three models to answer questions. We designated it as Bot Understanding Degree (BUD).

$$BUD = \frac{Correct\ answer}{Number\ of\ asked\ question} \quad (1)$$

We also evaluated the execution time and memory resources required by each model to answer the questions asked.

Figure 4 indicates that both BERT and DistilBERT significantly outperform the Rule-based model for the task measured by the metric BUD. BERT has the highest score, closely followed by DistilBERT, suggesting that while DistilBERT is a lighter and faster version, it retains much of BERT's effectiveness. The Rule-based model, with the lowest score, may lack the adaptability or contextual understanding that the transformer-based models (BERT and DistilBERT) provide.

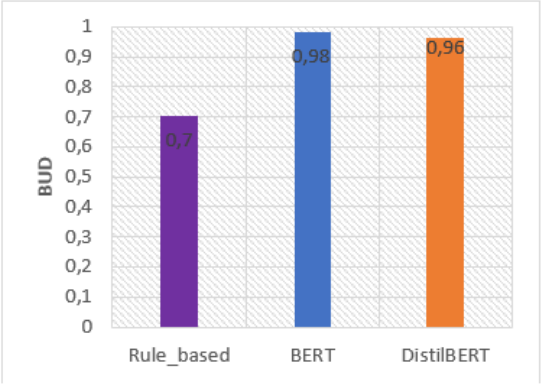

Figure 4: **Bot understanding degree of NLP models**

It requires less memory while still delivering comparable performance, making it a suitable option for applications where memory is limited. The Rule-based model, with only 15% memory usage, is the most economical in terms of resources. Its simpler design does not rely on large neural networks, resulting in minimal memory requirements. However, this simplicity often comes at the cost of reduced flexibility and effectiveness when compared to BERT and DistilBERT.

Figure 5 highlights the trade-offs between model complexity and memory efficiency. BERT, while powerful and accurate, is resource-intensive and may be unsuitable for environments with limited memory capacity. DistilBERT, with a smaller memory footprint, provides a balance by offering a significant reduction in memory usage with only a minor sacrifice in performance, as seen in the previous bar chart. The Rule-based model, with minimal memory usage, may be advantageous in highly constrained environments but is likely less effective for complex, context-dependent tasks.

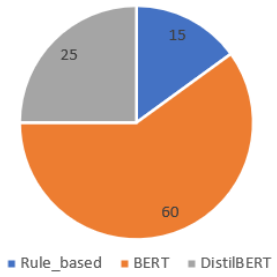

Figure 5: Memory (MB %) consumption in percentage

Figure 6 illustrates the average response times (in milliseconds) for processing requests by three models: Rule-based, BERT, and DistilBERT. BERT has the highest response time, with an average of approximately 600 ms per request. This extended response time reflects BERT's computational intensity due to its complex architecture and large number of parameters, which require considerable processing power. While BERT delivers high accuracy, its lengthy response time may be a drawback for applications that need real-time responses.

DistilBERT, on the other hand, has a shorter response time, averaging around 250 ms per request. As an optimized and lighter version of BERT, DistilBERT is designed for greater efficiency, significantly reducing response time while still maintaining much of BERT's accuracy. This makes DistilBERT a more viable choice for use cases where quick responses are critical. The Rule-based model is the fastest, with an average response time of less than 50 ms per request. Its simplicity enables rapid processing, as it doesn't require complex computations like those in neural network models. However, while the Rule-based model is quicker, it is generally less effective for managing complex, context-dependent tasks compared to BERT and DistilBERT.

Figure 5 highlights the trade-offs between response time and model complexity. BERT, while accurate, has a high response time that may not be suitable for applications requiring rapid responses. DistilBERT offers a balance, reducing response time significantly while maintaining a comparable level of accuracy to BERT, making it a better choice for time-sensitive tasks. The Rule-based model, with its minimal response time, is ideal for basic tasks with strict latency requirements but may lack the adaptability and sophistication needed for more complex queries. In summary, the choice of model depends on the specific requirements of the application: BERT for maximum accuracy, DistilBERT for a balance of speed and performance, and Rule-based for scenarios prioritizing speed over complexity.

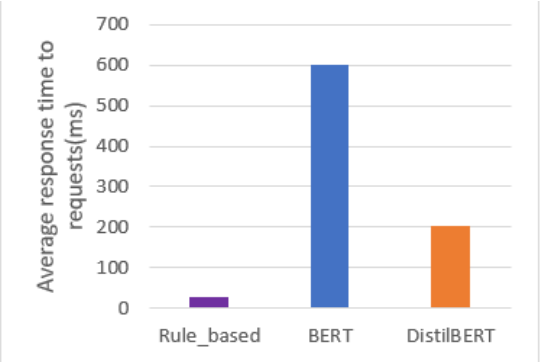

Figure 6 : **Processing user query times**

IV. CONCLUSION

In conclusion, our study demonstrates that DistilBERT offers a viable solution for developing an effective chatbot to support students in e-learning. The comparative analysis of the three models—Rule-based, BERT, and DistilBERT—reveals key differences in accuracy, memory usage, and response time that highlight the strengths and limitations of each approach. BERT achieved the highest score on the Bot Understanding Degree (BUD) metric, showcasing its accuracy and contextual understanding capabilities. However, this comes at the cost of substantial memory consumption (60%) and a long response time (600 ms), making it less suited for applications with limited resources or real-time requirements. DistilBERT, while slightly lower in accuracy, proved to be a more resource-efficient alternative, requiring only 25% of memory and delivering responses in an average of 250 ms. This balance of performance and efficiency positions DistilBERT as a practical choice for applications where both accuracy and speed are essential. The Rule-based model, while consuming the least memory and providing the fastest response times, lacks the adaptability and contextual understanding of the transformer-based models. Its simplicity makes it suitable for straightforward tasks but less effective for complex, context-dependent interactions. Overall, DistilBERT emerges as the most suitable model for a student-support chatbot in e-learning environments, offering a compromise between the high accuracy of BERT and the efficiency of the Rule-based model. This balanced approach makes it ideal for real-time, resource-constrained applications where both performance and quick responses are crucial.

REFERENCES

[1] Mastan, I. A., Sensuse, D. I., Suryono, R. R., & Kautsarina, K. (2022). Evaluation of distance learning system (e-learning): a systematic literature review. *Journal Teknoinfo, 16(1), 132-137*.

[2] Akhmedova, Z. (2023). Disadvantages of electronic learning. Current approaches and new research in modern sciences, 2(12), 99-109.

[3] Adamopoulou, E., & Moussiades, L. (2020). Chatbots: History, technology, and applications. *Machine Learning with applications, 2, 100006*.

[4] Bakouan, M., Kone, T., Kamagate, B. H., Oumtanaga, S., & Babri, M. (2018). A chatbot for automatic processing of learner concerns in an online learning platform. *Int. J. Adv. Comput. Sci. Appl, 9(5), 168-176*.

[5] Wong, A. (2022, April). The design of an intelligent chatbot with natural language processing capabilities to support learners. *In Journal of Physics: Conference Series (Vol. 2251, No. 1, p. 012005). IOP Publishing*.

[6] Ng, S. Y., Lim, K. M., Lee, C. P., & Lim, J. Y. (2023, December). Sentiment Analysis using DistilBERT. *In 2023 IEEE 11th Conference on Systems, Process & Control (ICSPC) (pp. 84-89). IEEE.*

[7] Vaswani, A. (2017). Attention is all you need. *Advances in Neural Information Processing Systems 30 (pp. 6000–6010)*

[8] Choi, S. R., & Lee, M. (2023). Transformer architecture and attention mechanisms in genome data analysis: a comprehensive review. *Biology, 12(7), 1033*.

[8] Tunstall, L., Von Werra, L., & Wolf, T. (2022). Natural language processing with transformers. " *O'Reilly Media, Inc. 2022, 383–406* .

[9] Sanh, V. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108.*

[10] ÖZKURT, C. (2024). Comparative Analysis of State-of-the-Art Q&A Models: BERT, RoBERTa, DistilBERT, and ALBERT on SQuAD v2 Dataset. *chaos and Fractals — Cutting-Edge Scientific Solutions. Vol. 1, No. 1 (2024), 19-3*