AI System for Verifying Soft Skills Based on Generative Artificial Intelligence to Create Tasks in AC/DC Sessions

Maciej Dziewit, Paweł Sienkiewicz, Agnieszka Czapiewska, Paweł Czapiewski

***Abstract*—To evaluate the soft skills of high-level job candidates, a system is needed that reliably assesses their wide range of competencies based on standardized tasks. Assessment Center/Development Center (AC/DC) sessions are precisely such a system. These sessions consist of tasks based on hypothetical situations, and assessors assess competencies based on the candidate's responses. This process is quite lengthy and costly because of the involvement of several qualified individuals who can assess the candidate's skills.**

The development of artificial intelligence and various large language models allows for the assessment of people's competencies without human involvement. We utilized LLM's (such as ChatGPT) to evaluate candidates responses and determine whether they possess a particular skill. The model was provided with the knowledge necessary to assess a person and to check if he has a given skill.

The LLMs were tested using a sample task that could be part of an AC/DC session. The candidate was required to record 3 videos, each answering one of three questions.

For each task, specific traits were assigned to determine whether the candidate demonstrated a given soft skill. Based on an assessment guide provided by the client and knowledge of behavioral interviews and the S.T.A.R. method, appropriate prompts were designed for the LLMs.

The tasks were also evaluated by an assessor, whose evaluations served as a reference standard against which all models were compared. Based on this comparison, the best performing model was identified by evaluating the LLMs using established metrics.

***Index Terms*—LLM, natural language processing, assessment, prompt engineering.**

I. Introduction

In the modern job market, assessing candidates' soft skills has become a vital part of the hiring process. While technical proficiency is still crucial, attributes like communication, teamwork, problem analysis, or making judgments often play a decisive role in a candidate's success within a company. These skills empower employees to work well together, face workplace challenges and foster a supportive organizational environment.

Soft skills are especially important in roles that involve regular interaction with others, such as those in customer service, management, or healthcare. Likewise, in fields such as medicine, qualities such as empathy and effective communication can be just as essential as clinical expertise when providing patient care.

Authors are with Gdansk University of Technology, Poland (email: A. Czapiewska: agnieszka.czapiewska@pg.edu.pl, M. Dziewit: s185755@student.pg.edu.pl, P. Sienkiewicz: s185698@student.pg.edu.pl) and with Solution (email: P.Czapiewski: p.czapiewski@solution.pl)

For example, a highly skilled engineer might face challenges in a leadership position without the ability to motivate and guide a team. That is why incorporating soft skills assessments into recruitment ensures that organizations select candidates who align with their values and can adapt to evolving challenges [6]. In recent years, artificial intelligence (AI) has played a pivotal role in the transformation of human resource management (HRM) processes. Traditional approaches to recruitment, employee evaluation, and workforce management are being replaced by modern, technology-driven solutions. AI facilitates the automation of routine tasks, improves the objectivity of evaluations, and optimizes HR processes.

Since the invention of the Transformer architecture, the development of neural networks has accelerated significantly. The Transformer architecture, with its self-attention mechanism, enabled parallel data processing and efficient handling of long-term dependencies in text. This groundbreaking solution replaced the previously dominant Recurrent Neural Networks and Long-Short-Term Memory, which were limited in terms of scalability and computational efficiency [1]. This enabled the development of Large Language Models (LLMs).

LLMs are distinguished by their ability to acquire so-called emergent capabilities, which were not present in smaller models. These include:

In-context Learning: The ability to learn new tasks from a small set of examples provided within the query. Instruction Following: After fine-tuning, LLMs can execute new types of tasks without requiring additional training. Multi-step Reasoning: The capability to solve complex problems by breaking them down into logical steps [3]

Solutions such as BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer) gained popularity due to their outstanding performance in natural language processing (NLP) tasks. These models made it possible to perform tasks such as translation, text generation, sentiment analysis, and question answering with unprecedented effectiveness [2].

AI has revolutionized recruitment by automating resume analysis and candidate screening. Algorithms analyze thousands of resumes within seconds, identifying key qualifications, skills, and experiences relevant to job descriptions. The candidate is ranked according to predefined criteria, such as education, technical skills, and certifications [9]. In this way, human biases are reduced by focusing purely on objective data, ensuring a more inclusive hiring process. Moreover, AI tools proactively search for potential talent on professional

platforms and social networks [10].

Systems employing NLP and machine learning techniques have been developed to evaluate subjective responses, reduce manual effort, and increase evaluation consistency [7]. These systems analyze textual answers based on predefined model answers and grading rubrics [8].

Interactive chatbots are another way to incorporate AI into competency evaluations. These AI-powered tools provide a more dynamic approach to assessing candidates, moving beyond conventional structured interviews and static questionnaires. By interacting with candidates in real-time, AI chatbots can not only pose predefined questions but also adapt to the responses, seeking clarification and delving deeper when required.

These AI tools leverage NLP to extract valuable insights from candidates' answers. They are capable of classifying intents and identifying key entities in responses, ensuring that the evaluation is comprehensive and detailed. For instance, if a candidate's answer to a teamwork-related question lacks sufficient detail, the chatbot can follow up with customized sub-questions to gather the missing information.

The integration of AI in competency assessments also mitigates common issues in human-led evaluations, such as bias and inconsistency. By relying on predefined criteria and objective analysis of responses, AI guarantees a more equitable and standardized process. [11].

In the growing domain of AI-driven competency evaluation, this study provides a distinctive contribution by performing a comparative analysis of how Large Language Models assess human competencies. While the focus has traditionally been on improving NLP tasks or domain-specific applications, such as education or recruitment, no research to date has systematically compared LLMs specifically in their capability to evaluate competencies across varying scenarios.

A novel comparative framework is introduced to evaluate the consistency and reliability of competency assessment by LLMs. In addition, emerging abilities such as contextual understanding and adaptability in human-centered evaluations are explored. The alignment of these models with human evaluation standards in structured, behavioral, and open-ended assessment formats is also assessed.

By benchmarking LLMs on these dimensions, first insights are provided into their potential for revolutionizing recruitment and talent evaluation are provided. This study represents a step forward in using artificial intelligence (AI) for competency evaluation and serves as a foundation for future explorations into the integration of LLM in human resource management and educational assessments.

II. Difference between traditional interview and behavioral interview

Traditional interviews and behavioral interviews are distinct methods of evaluating job candidates, each with its own focus and methodology. Traditional interviews focus on general job descriptions, resumes, technical credentials, and hypothetical questions. They often rely on unstructured formats, with questions varying between candidates. This approach can lead to subjective assessments and inconsistent comparisons, making it challenging to predict future performance accurately. Candidates may provide rehearsed answers, further complicating the evaluation process [4].

Behavioral interviews, on the other hand, focus on assessing the past behaviors of candidates to predict their future performance. This method uses structured questions directly related to the competencies required for the role, based on the notion that past behavior is the best predictor of future behavior. Instead of hypothetical scenarios, candidates are asked to recount specific experiences using the STAR (Situation, Task, Action, Result) method, which provides concrete insight into their skills, decision-making processes, and final results [4]. The time distribution between these approaches differs considerably. Traditional interviews devote substantial time to discussing credentials and hypothetical questions, while behavioral interviews prioritize probing past behaviors and relevant experiences. The structured nature of behavioral interviews reduces biases, ensures consistency, and helps uncover deeper insights into the motivations and work style of a candidate [4].

Compared to traditional interviews, behavioral interviews align employees with the company's tactical and strategic goals, effectively communicating the expectations for both individual and organizational success throughout the organization. Despite that behavioral interviews demand more time and extensive preparation, they offer a stronger foundation for identifying candidates whose skills and values align with the organization. This makes them a more effective method for making well-informed hiring decisions [4].

III. S.T.A.R. method

The STAR method is a structured approach used to answer behavioral interview questions that assess a candidate's past experiences to predict future behavior in similar situations. It stands for Situation, Task, Action, and Result [5].

- **Situation:** Describe the context or background of the scenario you faced, including relevant details such as where and when it occurred. This sets the stage for the interviewer.
- **Task:** Outline the specific responsibility or challenge you were tasked with in that situation. This helps the interviewer understand what was at stake.
- **Action:** Explain the steps you took to address the challenge or complete the task. Focus on your role and the decisions you made, as this demonstrates your problem-solving skills and initiative.
- **Result:** Conclude by sharing the outcome of your actions, quantifying success when possible. Emphasize positive outcomes, such as improvements, achievements, or lessons learned.

IV. Overall goal of the project

The goal of this project was to create a program that would assess the soft skills of the candidates using LLMs.

The entire evaluation process is as follows:

1) The candidate is given a task to record three short video answering to the following questions. Max 3 minutes per video
   - Describe a difficult problem that you managed to solve.
   - Describe a situation in which your suggestions improved a process or created an optimization.
   - Describe a situation where you had to complete a task in collaboration with other people.
2) Next, the candidate sends the video files to Google Drive
3) The program downloads files from the disk using the API
4) The program transcribes the mp4 file into a text file and saves it in a folder
5) The LLM used in the program assesses the relevant competencies by providing appropriate prompts
6) The program saves the results to xlsx file, presenting the assessments and confirming it's choice by showing fragments of the candidate's statement.
7) The results of different LLM's are compared with assessor's assessments. This is how the model results are determined in various metrics

V. Performance Evaluation

Performance evaluation involves quantifying the effectiveness of a classification model in predicting outcomes. Key metrics used in this evaluation include accuracy, precision, recall, and F1-score, all of which are derived from the confusion matrix. This section explores these metrics and their applications [12].

Confusion Matrix

The confusion matrix is a square table summarizing the number of correctly and incorrectly classified instances in terms of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). It provides a detailed view of the classifier's performance for each class.

	Predicted Positive	Predicted Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

Accuracy

The accuracy measures the overall correctness of the model:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision

Precision indicates how well the classifier predicts the positive class among all instances classified as positive:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Recall

Recall (also known as sensitivity) reflects the percentage of actual positive instances that were correctly identified by the model:

$$\text{Recall} = \frac{TP}{TP + FN}$$

F1-Score

The F1-score is the harmonic mean of precision and recall:

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

In this study, evaluation metrics such as precision, recall, and F1 score were used to compare the responses generated by the models tested with those provided by a professional assessor. This approach allowed us to quantitatively assess how well the models aligned with expert judgments.

VI. Using server-less API for our goals

Downloading large language models to a computer can be problematic for several reasons. First of all, such models often take up to even 40 gigabytes of storage space, posing a challenge for devices with limited memory. Secondly, they require significant computational resources, such as powerful processors and large amounts of RAM, which may be beyond the reach of the average user. In addition, running a model locally involves configuring specialized software, which can be complex and time-consuming. It is also worth noting that updating such models locally can be challenging. A solution to these problems is the use of serverless API's, which allow access to remote servers without the need to install or manage the model. API's provide instant access to the latest versions of models without burdening local devices. This approach enables users to take advantage of advanced AI technologies in a simpler, faster, and more efficient way.

VII. Development of the program

To make this process fully automated, the Python programming language was used.

A. Program composition

The evaluation process from the programming side consists of the following sub-points

1) Downloading MP4 files from Google Drive.
2) Transcribing to .txt files.
3) Sending a query to the model with the appropriate indicator.
4) Collecting responses and saving them in .xlsx files.

B. Process of Downloading Files from Google Drive Using the Google Drive API

The code starts by authenticating the user to gain access to Google Drive. It checks if a saved token file (`token.pickle`) exists, which stores authentication credentials. If the file exists, the token is loaded, allowing the user to skip logging in again. In case the token is missing or expired, the user is prompted to log in using OAuth 2.0 with credentials provided in the `credentials.json` file. Once authenticated, a new token is saved to `token.pickle` for future use, making subsequent logins easier.

After obtaining the credentials, a Google Drive API service is created (`build('drive', 'v3', credentials=creds)`), enabling communication with

Google Drive. The code then uses a query (`query`) to search for MP4 files in a specified folder, identified by the `folder_id` variable. It searches for files with MIME types corresponding to video formats, such as `video/mp4` or `video/quicktime`. If files are found, their list is iterated. Each file is downloaded locally to the disk using the `MediaIoBaseDownload` class. During the download, the progress is displayed as a percentage.

C. Transcription Process Using WhisperAI

Once the files are downloaded from Google Drive, the code transcribes each one using the WhisperAI model. In the first step, the Whisper model is loaded (`whisper.load_model('small')`). The `small` version of the model is chosen to provide a good balance between accuracy and computational efficiency. For each MP4 file, the code checks if a transcription file (named based on the video file name with `_transcription.txt` appended) already exists. If it does, the file is skipped to avoid reprocessing previously handled data.

The video file is then sent to the `model.transcribe()` function, with the transcription language set to Polish (`language='pl'`). The transcription result, contained in the `result['text']` key, is saved to a separate text file. After saving the transcription, the original MP4 file is deleted from the local disk to save space. If an error occurs during the deletion, an appropriate message is displayed to the user.

The entire process concludes once all files in the folder have been processed. This code efficiently automates the downloading of video files, transcribing their content into text, and organizing the results in a streamlined manner.

D. Sending Queries to the Model

To make it possible Text-Generation models were used. The process of sending queries to the model is handled by the $call_llm_via_chatfunction$. This function utilizes the Hugging Face API to interact with a pre-trained model. It takes three inputs: the Hugging Face client object, the transcription text, and a question about the candidate's competencies. First, the function constructs a JSON-formatted messages object. This object defines the model's role as an assessor and includes the transcription text for context, along with a detailed question designed to evaluate specific competencies. The request is sent to the Hugging Face API using the client.chat.completions.create() method. This method specifies the use of endpoints from the Hugging Face API to interact with a pre-trained model. For example ("HuggingFaceH4/zephyr-7b-beta") is an endpoint. The model processes the input and returns a response, which is then extracted and passed to further stages for analysis and storage. In each iteration, the program sends a query containing the transcript and the appropriate indicator for evaluation.

A second version of the program was written in which OpenAI libraries were used instead of Hugging Face. In this version, the program interacts with OpenAI's GPT models to perform the same tasks. The process involves calling OpenAI's API to send the transcription data and related questions to the model, which then generates the corresponding responses based on the input.

The core functionality remains similar, but instead of using the Hugging Face client, the OpenAI Python library is employed to send requests to OpenAI's endpoints. The model chosen for this task can be one of OpenAI's large language models like GPT-3.5 Turbo or GPT-4o. These models are accessed by providing the necessary API key and specifying the model endpoint, which facilitates the processing of the input and retrieval of the output.

Once the responses are obtained from OpenAI, the next step is the same: sanitizing the output to remove any unwanted characters and extracting the relevant assessment (either a 1 or 0 for the competency evaluation). The results are stored in an Excel file as before, and various metrics are calculated for performance analysis, including Precision, Recall, F1-Score, and Accuracy.

In this version, the program leverages OpenAI's robust language models for natural language understanding, while also maintaining a similar structure for handling responses and saving the results in Excel files. The primary difference lies in the switch from Hugging Face to OpenAI's API and the corresponding changes in the method used for querying the model.

E. Saving Responses to Excel

The responses from the model are collected and organized in a structured manner. Initially, transcription text for each candidate is read from .txt files, and a separate Excel file containing descriptions of competencies, recruitment questions, and evaluation indicators is loaded into a Pandas DataFrame. For each competency, a corresponding question is sent to the model, and the response is processed to clean up unnecessary characters using the $sanitize_response$ function. The assessment value (either 1 or 0) is extracted from the end of the model's response.

These results are stored in a new Pandas DataFrame with columns for Index, Competency Description, Recruitment Question, Evaluation Indicator, Model Response, Final Assessment, Tokens Used, and Assessor Rating. The DataFrame is updated iteratively for each candidate and competency. Once all evaluations are completed, the DataFrame is saved to an Excel file. If a file with the same name already exists, it is overwritten to ensure the data remains current.

Additionally, metrics like True Positives (TP), False Positives (FP), False Negatives (FN), and True Negatives (TN) are calculated. These are added to the Excel file using OpenPyXL, which also computes derived metrics such as Precision, Recall, F1-Score, and Accuracy. These calculations are written as formulas in specific cells of the Excel sheet, ensuring that they update automatically based on the data. This systematic approach ensures that the evaluation results are stored in a format suitable for analysis and further use.

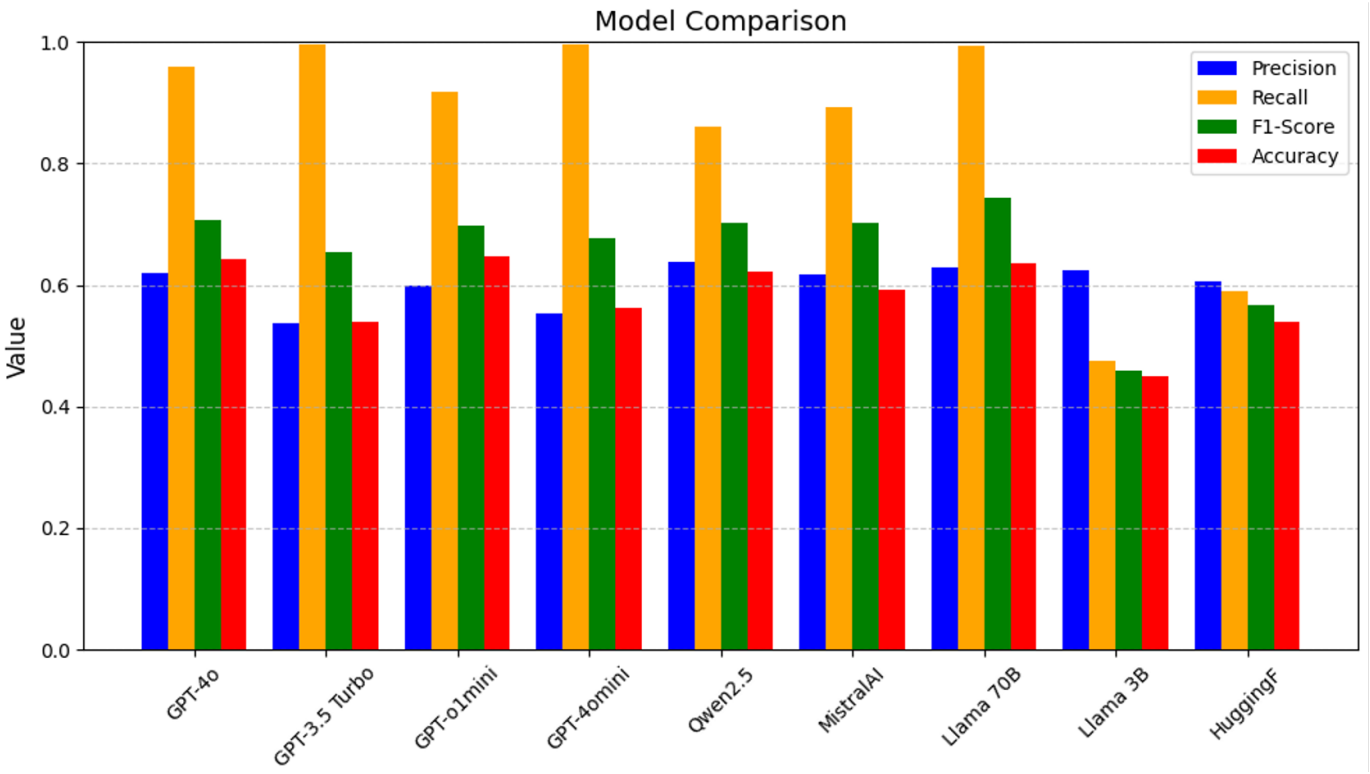

Fig. 1: Model results.

VIII. Results

The previously described metrics were used to check how well the models performed. The assessor's assessment in the model performance process is an exemplary value. The model results are as follows (*Fig. 1*).

After observing the results, the following conclusions were drawn. Assessment of candidates by models is too gentle compared to assessment by humans. The models performed best in Recall metrics. Recall calculates how well the model identifies all true positive cases. However, the Precision and Accuracy metrics show a significant drop in most cases. Precision measures how many of the positive cases predicted by the model are actually positive, while Accuracy measures the overall performance of the model, i.e. the percentage of all correct predictions (both positive and negative).

Therefore, it can be concluded that the models usually correctly assess that the candidate has a given feature/competence and award him a point in a given category. However, the Presicion metric shows that the models often incorrectly allocated a point to a person in a place where he should not have received it. Moreover, the Accuracy metric checking the correctness of all results is visibly lowered by incorrectly assigning a point where it should be 0. Additionally, from personal observations of individual results in various people, it was noticed that people who were rated high by the assessor have greater compliance of the results with the models. However, if the person was rated negatively by the assessor, i.e. had a lot of zeros in his or her score, there was a greater discrepancy between the assessor's assessment and the models' assessment.

Language models with a smaller number of parameters have often struggled to perform effectively on complex tasks. This limitation can be attributed to several factors that impact their ability to understand and generate nuanced, contextually accurate outputs. While smaller models are computationally efficient and require less memory and processing power, their reduced size often comes at the cost of representational capacity.

At their core, language models function by capturing patterns in data, learning to associate context with meaningful outputs. A model with fewer parameters has a reduced capacity to encode the vast and intricate relationships that exist within language. For instance, handling idiomatic expressions, long-range dependencies in text, or multi-step reasoning requires a depth of understanding that smaller models may simply not possess. As a result, they often falter when faced with tasks that demand a higher degree of comprehension or adaptability.

Another contributing factor lies in the trade-off between model size and generalization ability. Smaller models tend to generalize less effectively because their internal representations of language are less detailed. They might perform adequately on simpler tasks or datasets similar to their training data but struggle with unfamiliar contexts or domains. This is particularly evident in real-world scenarios, where language is highly diverse and unpredictable.

Moreover, smaller models are more likely to oversimplify problems or miss subtle nuances in input data. Tasks such as sentiment analysis, question answering, or creative text generation require a fine-grained understanding of semantics and context. Without sufficient parameters, the model may lack the expressiveness needed to fully grasp these complexities.

In contrast, larger models benefit from their increased capacity to store and process diverse patterns, enabling them to outperform their smaller counterparts across a wider range of tasks. While smaller models remain valuable for lightweight applications, their limitations highlight the importance of scale when striving for higher accuracy and performance.

Model	F1-Score
4o	0.785614
4omini	0.772245
Qwen	0.742973
Llama 70B	0.734130
MistralAI	0.706739
3.5turbo	0.704569
o1mini	0.700465
HuggingF	0.494311
Llama 3B	0.413520

TABLE I: Task 1 F1-Scores Sorted

Model	F1-Score
o1mini	0.676720
4o	0.635273
HuggingF	0.610406
MistralAI	0.599471
Qwen	0.586243
4omini	0.569048
Llama 70B	0.567412
3.5turbo	0.559788
Llama 3B	0.423810

TABLE II: Task 2 F1-Scores Sorted

Model	F1-Score
Llama 70B	0.822269
MistralAI	0.800845
Qwen	0.779254
o1mini	0.714286
4o	0.701786
3.5turbo	0.701786
4omini	0.689286
HuggingF	0.594444
Llama 3B	0.538835

TABLE III: Task 3 F1-Scores Sorted

Model	Mean F1-Score
Llama 70B	0.707937
4o	0.707558
Qwen	0.702823
MistralAI	0.702352
o1mini	0.697157
4omini	0.676860
3.5turbo	0.655381
HuggingF	0.566387
Llama 3B	0.458722

TABLE IV: Combined Results of All Tasks Sorted by Mean F1-Score

The analysis of model performance across various tasks reveals a few clear patterns. The top performers in terms of mean F1-scores were Llama 70B, 4o, and Qwen. Llama 70B achieved the highest mean F1-score of 0.7079, likely due to

its large capacity of 70 billion parameters, which enables it to capture more complex patterns in the data. 4o followed closely with a mean F1-score of 0.7076, demonstrating exceptional versatility by performing well across all tasks. Qwen and MistralAI also showed strong generalization abilities, with mean F1-scores above 0.70.

In the mid-tier, models such as o1mini, 4omini, and 3.5turbo performed well but slightly lagged behind the top-tier models. o1mini, in particular, excelled in Task 2, where it achieved the highest F1-score of 0.6767, but its performance on other tasks was not as competitive, which brought its mean score down. 4omini and 3.5turbo exhibited decent consistency, but lower scores in some tasks, particularly Task 2, prevented them from breaking into the top tier.

Weaker performers included HuggingF and Llama 3B, which had significantly lower mean F1-scores of 0.5664 and 0.4587, respectively. Llama 3B struggled across all tasks, which is likely due to its smaller parameter count compared to Llama 70B, limiting its ability to handle more complex scenarios. HuggingF was more competitive in Task 2 but underperformed in Tasks 1 and 3, indicating inconsistencies in its ability to generalize across different tasks.

The results also underscore the impact of model size and architecture on performance. Larger models, like Llama 70B, generally outperform smaller ones, such as Llama 3B, due to the larger number of parameters, which enable them to capture intricate data patterns. However, performance is not solely dependent on size. Models like 4o and Qwen, despite being smaller than Llama 70B, still performed competitively, suggesting that optimized architectures and training processes can compensate for a smaller model size.

Looking at the task-specific performance, Task 1 favored larger, more advanced models, with Llama 70B, 4o, and Qwen leading the charge. Task 2 highlighted the unique strengths of o1mini, which outperformed other models in that specific task. This suggests that some models may be better suited to tasks with certain patterns or requirements. Finally, Task 3, with the highest overall scores, appeared to be easier for the models to handle, with Llama 70B excelling.

In conclusion, while larger, state-of-the-art models tend to perform better overall, smaller, well-optimized models like 4o and Qwen can still compete effectively. These findings suggest that model size is not the sole determinant of performance; factors such as architecture, fine-tuning, and task alignment are also crucial in determining how well a model performs across different tasks.

ACKNOWLEDGMENT

The authors would like to thank the support received for this work.

REFERENCES

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin, "Attention Is All You Need" 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

[2] MOHAIMENUL AZAM KHAN RAIAAN, MD. SADDAM HOSSAIN MUKTA, KANIZ FATEMA, NUR MOHAMMAD FAHAD, SADMAN SAKIB, MOST MARUFATUL JANNAT MIM, JUBAER AHMAD, ,MOHAMMED EUNUS ALI, SAMI AZAM, "A Review on Large Language Models: Architectures, Applications, Taxonomies, Open Issues and Challenges", IEEE Access, Digital Object Identifier 10.1109/ACCESS.2024.3365742

[3] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu Richard Socher, Xavier Amatriain, Jianfeng Gao, "Large Language Models: A Survey", arXiv:2402.06196v2 [cs.CL] 20 Feb 2024

[4] C. Lin, "Behavioral Interview and its Implementation," 2010 3rd International Conference on Information Management, Innovation Management and Industrial Engineering, Kunming, China, 2010, pp. 74-76, doi: 10.1109/ICIII.2010.23.

[5] BADRIAH, Badriah; PERMANA, Indra. Enhancing Competence in Writing Best Practices Through the Utilization of the STAR Technique. JLER (Journal of Language Education Research), 2023, 6.3: 193-203.

[6] SCHULZ, Bernd. The importance of soft skills: Education beyond academic knowledge. 2008.

[7] Aarti P. Raut, C.Namrata Mahender, "Resolving Single-Sentence Answers for the development of an Automated Assessment Process", 2023 IEEE International Conference on Contemporary Computing and Communications (InC4)

[8] MUHAMMAD FARRUKH BASHIR, HAMZA ARSHAD, ABDUL REHMAN JAVED, NATALIA KRYVINSKA, SHAHAB S. BAND, "Subjective Answers Evaluation Using Machine Learning and Natural Language Processing", IEEE Access, Digital Object Identifier 10.1109/ACCESS.2021.3130902

[9] Chamudini Athukorala, Hirusha Kumarasinghe, Kavishka Dabare, Pasindu Ujithangana, Samantha Thelijjagoda, Pubudu Liyanage, "Business Intelligence Assistant for Human Resource Management for IT Companies" 20th International Conference on Advances in ICT for Emerging Regions (ICTer 2020): 220 - 225

[10] Dr. Smruti Ranjan Das, Prithu Sarkar, Dr. Shripada Patil, Dr. Ritesh Sharma, Dr. Shikha Aggarwal, Dr. Melanie Lourens , "Artificial Intelligence in Human Resource Management: Transforming Business Practices" 2023 10th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)

[11] BOUDJANI, Nadira, et al. Ai chatbot for job interview. In: 2023 46th MIPRO ICT and Electronics Convention (MIPRO). IEEE, 2023. p. 1155-1160.

[12] D.M.W. POWERS, "EVALUATION: FROM PRECISION, RECALL AND F-MEASURE TO ROC, INFORMEDNESS, MARKEDNESS & CORRELATION", Journal of Machine Learning Technologies ISSN: 2229-3981 & ISSN: 2229-399X, Volume 2, Issue 1, 2011, pp-37-63