AI-POWERED INTERVIEWING SYSTEM

Department of Computer Science

Muhammad Faizan	NIM-BSCS-2021-21
Muhammad Zubair Khan	NIM-BSCS-2021-36

2021-2025

Final year project report submitted in partial fulfillment of requirement for degree of Bachelors of Science in Computer Science

Namal University,
30-KM, Talagang Road, Mianwali, Pakistan.
www.namal.edu.pk

DECLARATION

The project report titled “AI-Powered Interviewing System” is submitted in partial fulfillment of the degree of Bachelors of Science in Computer Science, to the Department of Computer Science at Namal University, Mianwali, Pakistan.

It is declared that this is an original work done by the team members listed below, under the guidance of our supervisor Dr. Muzamil Ahmed. No part of this project and its report is plagiarized from anywhere, and any help taken from previous work is cited properly.

No part of the work reported here is submitted in fulfillment of requirement for any other degree/ qualification in any institute of learning.

Team Members	**University ID**	**Signatures**
Muhammad Faizan	NIM-BSCS-2021-21	_____
M. Zubair Khan	NIM-BSCS-2021-36	_____

Supervisor

Dr. Muzamil Ahmed

Signatures with date

_____

_____

Table of Contents

Abstract 7
Chapter 1 8
Introduction 8
Chapter 2 9
Literature Review 9
2.1 Question Generation 9
2.2 Response Evaluation 10
2.3 Dialogue Management 11
Chapter 3 13
METHODOLOGY 13
3.1 Data Acquisition and Feature Engineering 13
3.2 Natural Language Understanding (NLU) 15
3.3 Semantic Similarity Analysis 17
3.4 Confidence Prediction Using Convolutional Neural Networks 17
Chapter 4 19
Testing and Validation 19
4.1 Unit Testing 19
4.2 Integration Testing 19
4.3 Evaluation Scoring Validation 19
4.4 Functional Requirement Testing 20
4.5 Industry KPI Validation 20
Chapter 5 21
Results and Evaluation 21
5.1 Scoring Evaluation 21
5.2 Confidence Prediction Accuracy 21
5.3 System Latency and Efficiency 22
5.4 Overall System Evaluation 22
Chapter 6 23
Industrial Impact and Integration 23
6.1 Usability in Existing Workflow 23
6.2 Technical Deployment Readiness 23
6.3 Licensing and IP Considerations 23
6.4 Integration Challenges and Solutions 23

6.5 Feedback from Industry Mentors 24
Chapter 7 25
Discussion 25
Chapter 8 26
Conclusions 26
Chapter 9 27
Future Work 27
References 29

Table of Figures

Figure 1. Flow Diagram of AI Powered Interviewing System 13
Figure 2. Face detection and facing verification with MTCNN vs Competitors 15
Figure 3. Architecture diagram of DestilBert 16
Figure 4. Architecture diagram of RoBERTa 17
Figure 5 Confidence Score Comparision between Human vs System Evaluation 21
Figure 6. Confidence Prediction Loss and Accuracy Epochwise 22

List of Tables

Table 1. Confidence prediction model's Convolutional Neural Network (CNN)……..18

Table 2. Summary of performance benchmarks……....22

ACKNOWLEDGMENTS

We would like to express our deepest gratitude to the Computer Science Department at Namal University for providing us with the opportunity, guidance, and academic support throughout our Final Year Project journey.

We are especially thankful to our academic supervisor Dr. Muzamil Ahmed and our co-supervisor Mam Asiya Batool for their continuous mentorship, encouragement, and valuable feedback during every phase of our project.

We also extend our sincere appreciation to our industry partner, Arbisoft, for collaborating with us and providing crucial insights and industry exposure. We are grateful to the mentors and professionals at Arbisoft whose guidance helped shape our project into a practical and impactful solution.

We would like to acknowledge the technical support and resources made available to us through the FYP-Lab, which was exclusively dedicated to our project work. The access to equipment and environment contributed significantly to the successful completion of our work.

Finally, we thank everyone who contributed directly or indirectly to the success of this project. Your support has been invaluable.

Abstract

This project presents the design, development, and evaluation of an AI-Powered Interviewing System aimed at automating and improving the interview process for academic and professional settings. Traditional interviews are often time-consuming, prone to human bias, and difficult to scale. To address these limitations, our system uses advanced artificial intelligence techniques to conduct structured and unbiased interviews, particularly in the domain of computer science. The system combines multiple components including automatic question generation using Large Language Models (LLMs), speech-to-text conversion, Natural Language Processing (NLP) techniques like intent classification, Named Entity Recognition (NER), and semantic similarity analysis to evaluate candidate responses. It also includes a facial expression analysis module that predicts a candidate's confidence level using a Convolutional Neural Network (CNN) trained on labeled facial expression data. These components work together to produce a performance score based on both technical knowledge and non-verbal confidence cues.

The architecture ensures smooth interaction by handling different input types such as speech and frames, performing real-time evaluation, and adapting the interview flow based on candidate responses. Extensive testing was carried out through unit testing, integration testing, and comparison with human evaluations. The system achieved a 0.79 correlation with human scoring and an 81% accuracy in confidence prediction. It also demonstrated real-time performance with minimal latency, making it suitable for practical deployment. The final results confirm that the system is effective in automating interviews, reducing manual effort, and improving fairness in candidate evaluation. Though the system was tested primarily in the computer science domain and in English, it lays a strong foundation for future expansion into other fields and languages. Overall, this AI-Powered Interviewing System represents a significant step towards smarter, faster, and more objective recruitment and evaluation processes.

Chapter 1

Introduction

AI has been evolved and changed many fields such as human interaction with computers, understanding of natural language and improved automated decision making. An important use of AI applications is to automate interviewing process in which a system assesses and evaluates candidates by following rule-based assumption. Traditional interviewing process often takes a lot of time, requires more effort and human evaluation can be biased which makes this process less effective for hiring of many candidates. This suggests a need of a AI-based interviewing system that make interviewing process fast, fair and resource efficient to evaluate candidate technical knowledge, communication skills and confidence assessment. AI interview systems are mainly classified into two types as rule-based and machine learning-based. Rule-based systems follow rule-based decision making and structured template which limits the adaptability of interviewing process. While Machine learning-based approaches use Large Language Models (LLMs), Natural Language Understanding (NLU), and Deep learning to assess candidate answers, adjust questions dynamically by context management, and makes the evaluations of candidates smart. AI-driven interview systems are used in many areas, including hiring process, academic tests, corporate training, and language skill assessments. In the domain of computer science AI-powered interviews helps in evaluating the candidates technical(theoretical) knowledge, conceptual understanding, approach of problem solving and assess domain specific knowledge while ensuring fairness in objective evaluation of candidates.

Many studies have explored the automated interview systems [1], such as chatbots, AI-based candidate evaluation and NLP-based evaluations. However, current systems struggle with adapting dynamic questions, processing speech and text, assessing multiple factors, and ensure bias free evaluation. These challenges suggest the need for a more comprehensive, smarter, all-in-one AI interview system that includes speech processing, NLP, emotion recognition, and performance scoring of candidates. Thi research proposes an AI-based interview system for computer science domain. It includes automated question generation using LLMs, speech-to-text conversion, and NLP for understanding intent, recognizing entities by NER, and measure semantic similarity for assessing candidate responses. It also includes facial expression analysis to predict the confidence level of candidates based on non-verbal cues. The system adapts the interview flow and maintains candidate performance scores which aims to make the interview process more personalized, biasness-free and efficient to help the recruiters in shortlisting best candidates through fair ,efficient assessments and achieve following objectives.

- To automates the evaluation process of candidates in the computer science domain.
- To reduce time, human effort, and bias in traditional interviews using intelligent, automated methods.
- To generate contextually relevant interview questions dynamically using Large Language Models (LLMs).
- To reduce natural lanaguage ambiguities

Chapter 2

Literature Review

In next few sections, we will describe the key areas of research related to AI-powered interviewing systems. To make it clear and to provide a detailed understanding, we have divided the related work into three main sections i.e Question Generation, Response Evaluation, and Dialogue Management. Each of these section highlights the unique aspect of the interview process which contribute in developing an intelligent and efficient system to achieve objectives of automating interviewing process. We aim to highlight the progress, challenges, and opportunities in each field by discussing these areas seperatly and how combining them can improve AI-powered interviewing system assessments in the field of computer science.

2.1 Question Generation

Question generation focuses on generating meaningful and context relavant questions. It is broadly fall into two categories, rule-based question generation and dynamic question generation. Traditional methods use role based and sequence to sequence models for question-answering systems that failed to cover multiple aspects in the text and these require annotations. Previous work has explored question generation such as study [2] uses semantic role labeling with sequence to sequence models which doesn't require annotations and can generate multiple question from a single input. This method was evaluated on Car Manual, SQuAd and NewsQA datasets. This method performs better results than rules based and sequence to sequence models in creating diverse questions. However, as it relies on SRL which sometimes can generate similar question and it has not been tested for paragraphs. A study [3] addresses challenges of existing Controllable Question Generation (CQG) challenges that focus maintaining question difficulty while neglecting control of question content. This method proposes novel LLM-guided method called PFQS which was able to control both content and difficulty in question generations by first creating answer plan in more structure and control way. This method was evaluated on FairtaleQA that is annotated for content and difficulty control for question generation. The PFQS method performs better than current top methods which shows improvements in key metrics like MAP@1 with Rouge-L and BERTScore. However, it is only tested on FairytaleQA dataset and depends on expert-annotated datasets which makes it limited to use in other tasks. A study [4] highlight the effectiveness of Large Language Models (LLMs) in classifying and generation of educational questions. This study focus to evaluate ChatGPT classification and generation capabilities with different prompting strategies by using Graesser's taxonomy to classify 4,959 user-generated questions into ten categories to evalautes zero-shot and few-shot prompting techniques and applies voting method to aggregate the results of multiple prompts. For question generation it uses 100 reading sections from five online textbooks to evaluates ChatGPT ability to generate high quality questions with human generated question. This appraoch uses systematic comparison of different prompting stretegies that makes LLMs optimal for education. Two datasets has been used one for question classification in which 4,959 user-generated questions was labeled into 10 categories Graesser's taxonomy and other was for question generation which contain 100 reading sections from 5 online textbooks (Finance, Philosophy, Anatomy, Biology, Mathematics) for type specific question

generation and human evaluation. ChatGPT achieved F1-score of 0.70 in zero-shot classification when combined with Random Forest Classifier that performed best and improved classification using voting methods. For generating type-question generation ChatGPT accuracy was lower than expected and generated question sometime misclassified which indicates the need of post-filtering. As it used 5 textbooks for user-generated questions which maynot generalize across other contexts and it achieved lower F-score than Roberta(0.81). ChatGPT sometimes generated content far beyond the context and lead to expensive computational cost on expensive LLM API calls.

2.2 Response Evaluation

Response evaluation systems determine both quality and relevance of responses across educational settings and automated systems. The study of [5] examines how evaluation of descriptive answers faces three significant difficulties including time-intensive evaluation procedures while dealing with result bias and having inconsistent assessments. These approaches focus on keyword matching which failed in capturing semantic meaning in responses. This method uses hybrid model called DAES that uses LDA, T5 and Sentence-BERT for topic modeling and semantic analysis. It uses content base evaluation rather than keyword matching for comprehensive evaluation. They used dataset containing 300 questions with 1 as ideal answer and 30 students answer per question in computer science domain which was collected by web crawling and annotated manually. This method reduced biasness significantly in evaluation and it was better aligned with expert assesses scores. But, this appraoch requires high computation and it was tested solely on computer science questions. The research [6] addresses the challenges involved in automated subjective answer assessment through natural language processing along with machine learning methods. This method combined Word Mover's Distance (WMD) with Cosine using Multinomial Naïve Bayes (MNB) model as two step hybrid model for subjective answer evaluation. Through automated scoring systems the method demonstrated good accuracy by enhancing traditional keyword-based scoring with similarity measures alongside machine learning algorithms.

The dataset used consists of 1000 subjective questions in computer science and general knowledge domain while each question was having 1 model answer and 20 student responses per question. It was also crawled from educational websites and annotated manually by experts. WMD lead to high semantic accuracy and ML model achieved accuracy of 88% in predicting scores while reducing error rate to 1.3%. However, it may not generalize well as it was only tested on computer science and general knowledge domain and require computational cost for WMD based similarity. The research conducted [7] examines how Large Language Models (LLMs) particularly ChatGPT-4 deal with problems when assessing student-written material. The research adopts Chain-of-Thought (CoT) prompting to present accurate student answers prior to guiding ChatGPT-4 evaluation. The system uses ChatGPT-4 together with predefined evaluation criteria to conduct 10-shot assessments of student work that runs each response ten times while providing feedback on improvement areas to students. This study uses a dataset of 54 open-ended written responses in Master program of university. These responses are based on Irregular migration at the EU borderland (2,543 words), Irregular migration during the Covid-19 pandemic (3,734 words) and Knowledge creation processes (1,816 words). With this method ChatGPT-4 correctly recalled of 94.4% of student responses while achieving high consistency of 68.7% in evaluation by maintaining the same grade across multiple iterations. However, as this study was based on 54

student responses that lacks generalization in evaluation. Also, it only processes text responses and it is highly dependent on quality of prompts. A proposed study [8] addresses the challenges of Semantic Textual Similarity (STS) as traditional approaches relied on rule-base and feature engineering. This paper explores Siamese Recurrent Neural Network (SRNN) with variations of in Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU).

In Siamese Neural Network Architecture two identical RNN-based sub-networks is used to process sentence pairs with different shared parameters and weights while similarity between output embeddings is calculated using Manhattan distance. Different variants such as LSTM-based Siamese Network, Bidirectional LSTM, GRU-based Siamese Network, Bidirectional GRU, LSTM/GRU + Attention Mechanism and GRU + Capsule Networks is used forcapturing complex relationsip and better contextual understanding. This study used SICK (Sentences Involving Compositional Knowledge) and SemEval 2017 STS Datasets. In SICK dataset 9927 sentence pairs having simliarity scores between 1 and 5 with additional thesaurus-based data augmentation which added 10022 more training samples is used. In SemEval dataset 8277 training samples were collected from different previous competitions and 250 test sentence pari annotated with a similarity score (1-5). GRU based models performed better than LSTM as Bidirectional GRU achieved accuracy of (0.8792) across all STS tasks and attention mechanism improved performance for long sentences. However, as models was trained on SICK and SemEval datasets which may not generalize in other domain and study focus on English datasets which makes it uncertain to other languages. Bidirectional GRU and Capsule Networks require more computational power than simple LSTM models and GRU model struggled with active passive transformations.

2.3 Dialogue Management

Dialgue management involves maintaining the flow of conversation contextually in order to make it meaningful and user friendly. A study proposed [9] addresses the challenge of existing dialogue systems that struggle in mainting contextual and structured dialogue which lack user friendly coaching experience. This paper focuses on hybrid Dialogue Management (DM) that uses hierarchical planning to respresent dialogue as tree of intelligent agents. It uses template-based language generation in multilingual responses with GPT-2 using deep learning techniques for Natural Language Generation (NLG) that integrates Part-of-Speech (POS) and word embeddings for grammer corretencess. They tested the system for dialgue managment on 79 elderly participants from Spain (31), France (22), and Norway (26) using GROW model that covered the topics like nutrition habits, physical activity, social relationships. This system managed user engagement very high as 9.5 words per user turn (Spanish), 7.6 (French), and 5.4 (Norwegian) that successfully managed long coaching dialgues on multiple langauges.The integration of Neural-based NLG (GPT-2) performed better in generating grammatically and semantically correct sentences than traditional N-gram models. However, it lead to shorter responses in Norwegian due its lowest recognition accuracy. As they used Transformer-based language generation (GPT-2) which require highly computational resources and it mainly focus on nutrition coaching which need retraining for different coaching domains.A research conducted [10] highlights the challenges of discourse processing in spoken dialogue systems due to lack of proper context tracking, struggle with pragmatic adaptation and lack in support of multi-user and multi-dialogue interactions. This

paper proposes modular software architecture with Dialogue Management (DM), Context Tracking (CT) and Pragmatic Adaption (PA). Dialogue Management (DM) manages conversation flow, support mixed-initiative dialogues and handles meta-dialogues.

Context Tracking (CT) maintains context history of dialogue and integrates both linguistic and non-linguistic events while Progmatic Adaption (PA) maps natural languague inputs to valid system commands, detects violations of system constraints and converts responses into human friendly language. This research doesn't reply on standard dataset but demonstrates architecture in Time Reporting System (TRS) and ModSAF Interface (Battlefield Simulation). TRS is a telephone-based system where employees interact with an automated system to log work hours and uses natural langauge commands to retreive and updated records. ModSAF Interface is a voice-based interface for controlling military simulation units which allows new unit names to be added in real-time as it has ability to support dynamic speech recognition adaptation. This research improves system responsiveness, error handling, conversation continuity and smooth interpretation of system responses. However, as no standardized dataset was used which make hard to set a benchmark against other dialogue systems and computational complexity increases with multi-user and multi-dialogue interactions. It was tested on only through prototypes and Pragmatic Adaption requires customization for different domains. A study [11] addresses the challenge in traditional dialogue management strategies that typically designed manually, requireing extensive domain knowledge and iterative fine-tuning. This study relied on Markov Decision Processes (MDPs) and Reinforcement Learning (RL) for optimizing dialogue strategies in a spoken dialogue system (NJFun). MPD defines dialogue status on system-user interactions using reward-based learning and RL collects training dialogues,constructs MDP to estimate state transition probabilities and applies Q-learning and value iteration to learn optimal strategy. NJFun uses real-time spoken dialogue system, optimizes intiative and confirmation strategies to determine best optimal decision making. The study used 311 real user dialogues collected from 54 participants as training data and 124 dialogues collected from 21 participants as tested data to evaluate task completion rates and user satisfaction. This study improved task completion from 52% (baseline) to 64% (optimized RL), RL successfully learned an adaptive dialogue and optimized strategy improved speech recognition accuracy. However, this study focuses on single task-oriented dialogue system, RL requires multiple iterations to build a reliable MDP and it focuses on task completion but doesn't optimie user experience.

Chapter 3

METHODOLOGY

The system is developed to automate the evalautions of candidates. As shown in Figure 1, the system architecture consists of multiple connected components that collectively work together to streamline the interviewing process. Initially, candidate responses are received by speech and non verbal cues which furthur go through preprocessing and feature extraction steps. The system then uses natural language processing (NLP) techniques, intent classification, named entity recognition (NER), and semantic similarity analysis to evaluate responses. Additionally, a visual analysis module evalautes facial expressions to determine confidence levels. Finally, the system generates a performance score of candidates based upon their technical knowledge and confidence level as weighted evaluation metrics.

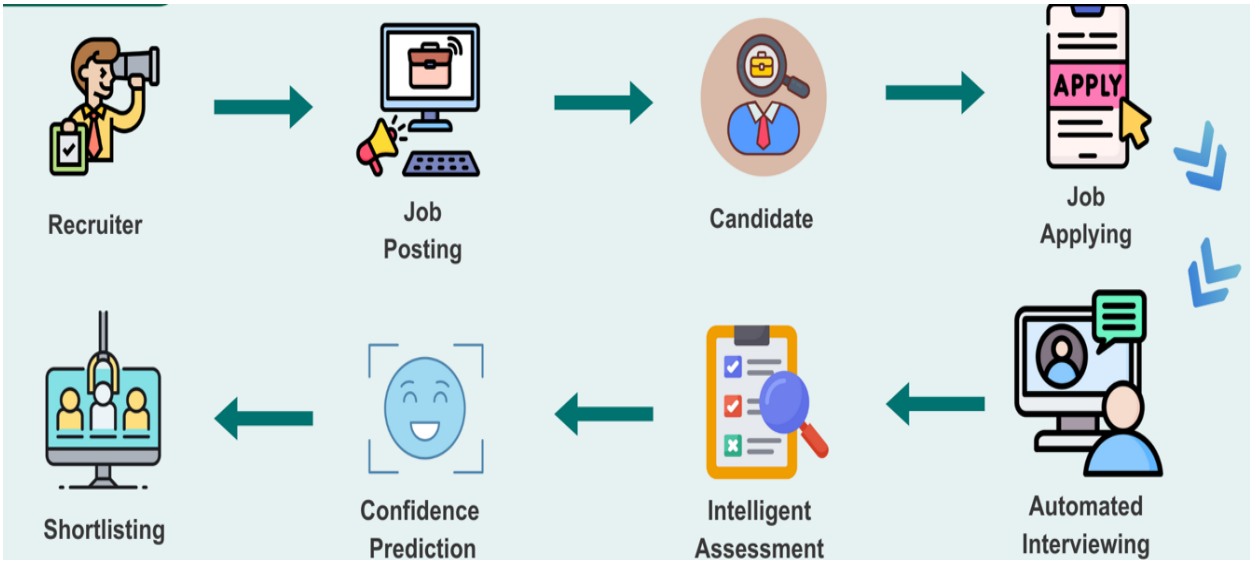

Figure 1. Flow Diagram of AI Powered Interviewing System

3.1 Data Acquisition and Feature Engineering

The system starts by getting candidate responses through speech and visual inputs. The speech module gets candidate responses using the speech recognition system and converts the candidate responses into text. Speech recognition allows the system to convert verbal responses into a machine-readable format which develops a basics for textual analysis. To ensure consistency and improve accuracy in evaluation the text is processed through several preprocessing steps after conversation. It includes tokenization [12], normalization, stop-word removal, and lemmatization. Tokenization divides the text into individual words for an efficient and detailed analysis. Normalization helps to normalize the converting words to their standard format, removing punctuation and handles other variations in the text. Stop-word removal reduces noise and computation cost by removing frequently occurring words that do not contribute significantly to

meaning (e.g., “the,” “and”). Lemmatization reduces the words to their base form to refine the text furthur and to maintain the proper dictionary form of words rather than simply truncating them.

Once the text is preprocessed, it is transformed into structured numerical representations through feature extraction techniques. The system uses Term Frequency-Inverse Document Frequency (TF-IDF) representation to achieve this transformation. TF-IDF assigns weight to words based on their importance, which ensures that commonly used words do not overshadow more meaningful terms. This structured representation enables the system to analyze and compare candidate responses effectively. The TF-IDF score for a term *t* in a document *d* is given by:

$$TF, IDF(t, d) = TF(t, d) * \log\left(\frac{N}{DF(t)}\right)$$

Where:

- TF(t,d) is the frequency of term *t* in document *d*,
- N is the total number of documents in the corpus,
- DF(t) is the number of documents in which term *t* appears.

Simultaneously, the system ensures that only one candidate is present in the video frame before proceeding with further processing. This verification is conducted using a face detection module based on Multi-Task Cascaded Convolutional Neural Networks (MTCNN). It operates in three sequential stages: the Proposal Network (P-Net), the Refinement Network (R-Net), and the Output Network (O-Net). The P-Net generates candidate face regions via convolutional filters and bounding box regression. The R-Net filters and refines these candidates through further convolutional layers and bounding box adjustment. Finally, the O-Net confirms face presence, refines bounding boxes, and extracts five facial landmarks (eyes, nose, and mouth) for precise localization.

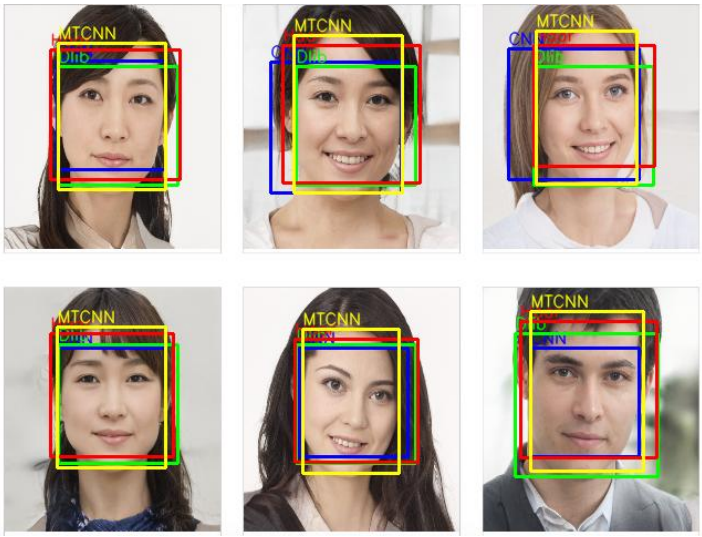

Figure 2. Face detection and facing verification with MTCNN vs Competitors

As shown in Figure 2, compared to traditional face detection methods such as Haar cascades, conventional CNNs, and DLIP (Deformable Part-based Models), MTCNN demonstrates superior performance in both speed and accuracy. While Haar cascades are computationally efficient but less accurate, and CNNs offer high accuracy at a higher computational cost, MTCNN provides an optimal balance of precision and efficiency. This makes it well-suited for real-time applications in domains such as surveillance, mobile platforms, and biometric systems.

3.2 Natural Language Understanding (NLU)

To evaluate candidate responses effectively, the system integrates NLU techniques [12], specifically Intent Classification and Named NER. NLU is a critical subfield of NLP that enables machines to interpret human language beyond basic text analysis. In this system, it plays a crucial role in managing the dialogue flow and ensuring that only relevant responses are assessed. Candidates may provide different types of responses, including requests for clarification, expressions of uncertainty, or even a decision to leave the interview. The system must distinguish between these various intents to ensure that only substantive answers to technical questions proceed to the assessment module.

Intent classification is used not only to evaluate whether a response is relevant to the question but also to guide the overall conversation flow. A candidate may ask for clarification, respond with an idle statement, or express a desire to exit the interview. These responses should not be graded, and intent classification ensures that the system appropriately handles them. If a candidate requests clarification, the system responds with additional details rather than sending the response for assessment. If a candidate wishes to leave the interview, the system detects this intent and terminates the session. Similarly, the system must differentiate between an actual answer to a technical question and an off-topic remark, ensuring that only meaningful responses contribute to the evaluation process.

For this task, the system utilizes DistilBERT, a distilled version of the BERT model. DistilBERT reduces the size of BERT by 40% while retaining 97% of its performance. It is trained using

knowledge distillation, where a smaller model (student) learns to mimic a larger pre-trained model (teacher). As displayed in the Figure 3, The architecture of DistilBERT consists of a 6-layer Transformer encoder, multi-head self-attention mechanisms, and feed-forward layers, totaling 66 million parameters. It processes tokenized text inputs and captures contextual relationships across tokens, enabling effective identification of intent. DistilBERT's efficiency makes it suitable for real-time classification within the interactive interview environment.

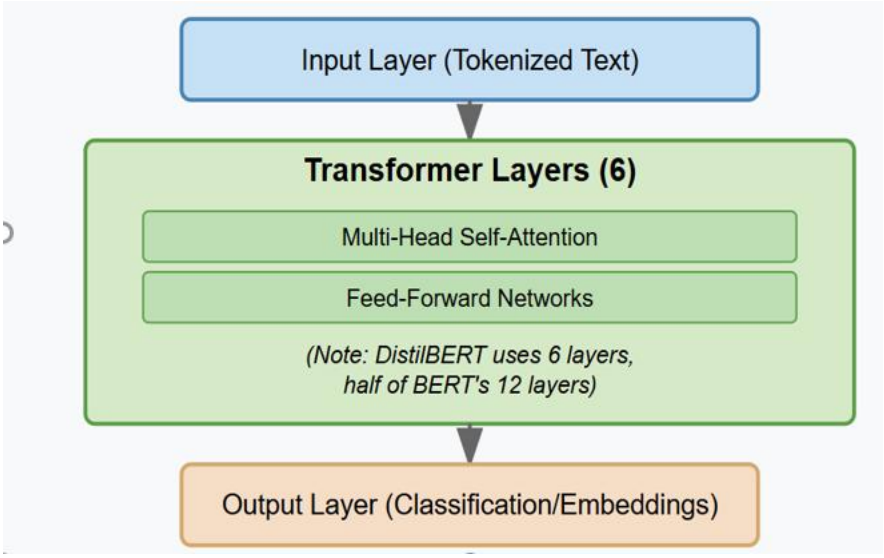

Figure 3. Architecture diagram of DestilBert

The ability of DistilBERT to capture nuanced intent makes it ideal for managing the conversation flow. When the system detects that a response does not belong to the category of gradable answers, it redirects the interaction accordingly. For instance, if a candidate asks, *"Can you clarify the question?"*, the system does not treat it as an answer but instead provides additional information. If the response contains hesitation, such as *"I'm not sure, but I think..."*, the system assesses whether the content following the uncertainty is relevant before making a classification. If a candidate states, *"I would like to exit the interview,"* the system immediately recognizes the intent and gracefully ends the session.

Alongside intent classification, NER is used to extract domain-specific information from candidate responses. While intent classification ensures that responses are correctly routed, NER focuses on identifying key elements within a valid answer, such as programming languages, algorithms, or technical concepts. The system uses RoBERTa (Robustly Optimized BERT Pretraining Approach) for this task, which is based on a 12-layer Transformer architecture as shown in Figure 4. RoBERTa improves upon BERT by eliminating the Next Sentence Prediction (NSP) objective, employing dynamic masking, and pretraining on a larger corpus. These optimizations enhance its ability to understand nuanced and diverse textual patterns.

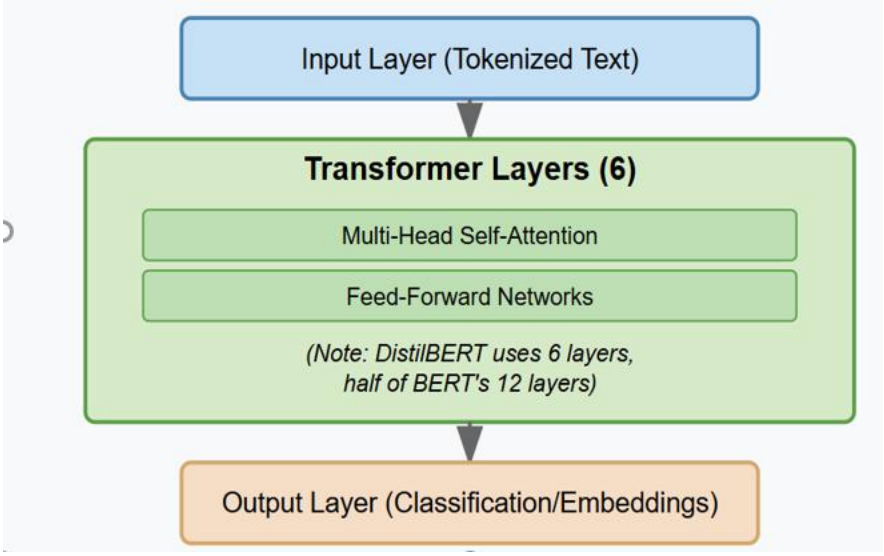

Figure 4. Architecture diagram of RoBERTa

RoBERTa tokenizes input responses and classifies each token using a token-level classifier, which may be a Conditional Random Field (CRF) layer or softmax applied to each token independently. This enables sequence tagging for NER tasks. The model identifies and labels relevant entities, allowing the system to verify whether a candidate has mentioned key technical concepts necessary to address the question. Unlike simple keyword matching, RoBERTa captures contextual dependencies, enabling more accurate entity extraction even in the presence of synonymous or paraphrased terminology.

3.3 Semantic Similarity Analysis

The system evaluates candidate responses using LLMs, which have been carefully compared to identify the most suitable model for this task. These LLMs generate contextual embeddings for both the candidate's response and the reference answer, capturing semantic nuances beyond simple keyword matching.To measure the alignment between responses, the selected LLM computes similarity scores [6] based on its deep contextual understanding of language. This process ensures that answers with different wording but similar meaning are correctly assessed. The performance of various models was rigorously measured to determine the most effective one, and detailed evaluation results will be provided in the Results Section.

3.4 Confidence Prediction Using Convolutional Neural Networks

In addition to textual analysis, the proposed system includes a facial expression-based confidence prediction module. This module evaluates non-verbal cues to determine whether a candidate appears confident or unconfident. These predictions contribute to the final performance score, adding an emotional intelligence component to the assessment. The model is trained on a labeled dataset of facial expressions. Images are categorized into confident and unconfident classes. All input images are resized to 48x48 pixels, converted to grayscale, and normalized. Data augmentation is applied using rotation, shifting, shearing, zooming, and horizontal flipping. These transformations improve model generalization and reduce overfitting.

The confidence prediction model is a deep Convolutional Neural Network (CNN). Its architecture is described below:

Layer No.	Layer Name	Description
1	Input Layer	Accepts grayscale images of shape (48, 48, 1)
2	Convolutional Block 1	• Conv2D with 64 filters (3×3), padding='same' - Batch Normalization • ReLU activation - MaxPooling2D (2×2) - Dropout (25%)
3	Convolutional Block 2	• Conv2D with 128 filters (5×5), padding='same' - Batch Normalization • ReLU activation - MaxPooling2D (2×2) - Dropout (25%)
4	Convolutional Block 3	• - Conv2D with 512 filters (3×3), padding='same' - Batch Normalization • ReLU activation - MaxPooling2D (2×2) - Dropout (25%)
5	Convolutional Block 4	• - Conv2D with 512 filters (3×3), padding='same' - Batch Normalization • ReLU activation - MaxPooling2D (2×2) - Dropout (25%)
6	Flatten Layer	• Converts 2D feature maps into a 1D feature vector
7	Fully Connected Layers	• - Dense layer with 256 units + Batch Normalization + ReLU + Dropout (25%) • Dense layer with 512 units + Batch Normalization + ReLU + Dropout (25%)
8	Output Layer	• Dense layer with 1 unit and sigmoid activation for binary classification

Table 1. Confidence prediction model's Convolutional Neural Network (CNN).

The model is compiled using the Adam optimizer with a learning rate of 0.01. Binary cross-entropy is used as the loss function. Several callbacks are employed, including EarlyStopping, ModelCheckpoint, and ReduceLROnPlateau. These help to avoid overfitting and ensure the best model is saved. The model is trained for 200 epochs with a batch size of 64. After training, the model achieves high validation accuracy. It is then evaluated on a separate test set to ensure generalization. The predicted confidence score is used in the final evaluation of the candidate. It is combined with textual and verbal response analysis using weighted metrics. This visual analysis module helps the system make holistic and fair decisions by incorporating behavioral signals.

Chapter 4

Testing and Validation

This section outlines the testing methods used to validate the system. Testing was conducted in multiple phases. These included unit testing, integration testing, and scoring validation. The objective was to ensure that all modules function correctly, both independently and as a complete system. The tests also measured the system's performance against functional requirements and key industry benchmarks.

4.1 Unit Testing

The system was first tested at the module level. Each core module was tested individually to confirm its internal functionality. The authentication module was tested for sign-up, login, and access control. Job management features were tested for creating, updating, and deleting job posts. The interview session module was checked for question delivery, audio input handling, and response recording. Evaluation logic, including intent detection, entity recognition, and similarity scoring, was tested in isolation. The shortlisting module was tested for candidate ranking based on evaluation metrics. Each module was executed with various input conditions to ensure expected output. Edge cases were also tested to observe module behavior in less common scenarios. Unit testing helped identify and resolve early-stage bugs. It confirmed that individual components were ready for system-wide integration.

4.2 Integration Testing

After successful unit testing, the modules were connected and tested as a complete system. Integration testing was carried out by running full-length interview sessions. These tests helped ensure smooth data flow across modules. Candidate responses passed successfully from speech input to text conversion, then to natural language processing and evaluation. The system was observed carefully during these tests. Logs were generated for each step. All major functions were verified, including real-time communication, scoring, and session management. Any integration issues or data mismatches were resolved immediately. The full pipeline operated as expected without breakdowns or miscommunication between modules.

4.3 Evaluation Scoring Validation

Scoring accuracy was tested using a custom evaluation dataset. A fixed set of technical questions was prepared. Candidate responses were collected separately. These answers were then scored manually by human interviewers. At the same time, our system scored the same responses using its automated evaluation logic. The results from the system and human evaluators were compared. The comparison focused on how closely the system scores aligned with human judgment. A strong correlation was observed between both scoring methods. This showed that the system can evaluate candidate responses with a high degree of reliability and fairness. It also validated the design of the semantic similarity and NLU models.

4.4 Functional Requirement Testing

Each major feature of the system was tested against the defined functional requirements. The question generation module produced contextually relevant questions. Speech-to-text conversion worked consistently across different voice samples. Intent classification correctly identified candidate replies such as clarification requests or irrelevant responses. Named entity recognition accurately detected key technical terms in the answers. Semantic similarity scoring responded well to both direct and paraphrased inputs. The confidence prediction module was tested using facial expression data. It successfully identified confident and unconfident expressions during candidate responses. The final score, based on both technical and visual inputs, was generated without error. The complete system met all functional expectations defined during the proposal stage.

4.5 Industry KPI Validation

Although no direct industry datasets were provided, the system was tested based on standard industry KPIs. These included accuracy, response time, fairness, and user experience. Accuracy was confirmed through scoring validation. Response times were measured during integration testing and found to be within acceptable limits. The scoring model was unbiased and consistent. The user interface and flow were smooth and intuitive. These tests confirm that the system aligns well with performance expectations for practical deployment. The results indicate that the system can support real-world use in academic or hiring contexts.

Chapter 5

Results and Evaluation

This chapter presents the performance of the system. It is based on the test cases described earlier. The results confirm that the system works effectively. Each major component was tested. The results are supported with evaluation scores and observed behavior.

5.1 Scoring Evaluation

We tested the system's scoring accuracy using human judgment as a benchmark. A set of questions and candidate answers was used. Human evaluators gave scores independently. Our system scored the same answers using its built-in logic. We compared both sets of scores. A strong positive correlation of **0.79** was observed. This shows that the system scores are close to human judgment. The results confirm that the scoring model is reliable and fair.

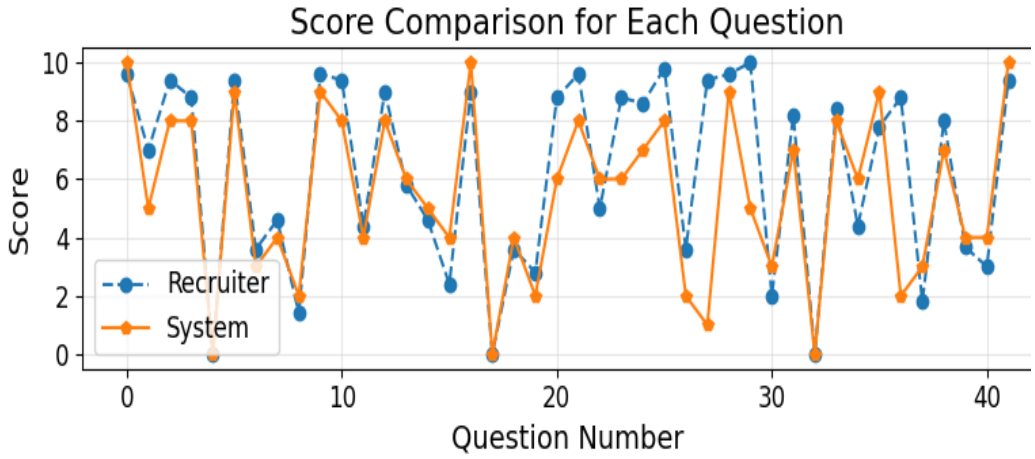

Figure 5 Confidence Score Comparision between Human vs System Evaluation

5.2 Confidence Prediction Accuracy

The confidence prediction model was tested using labeled facial expression data. The model classified candidates as confident or unconfident based on facial cues.After training and testing, the model achieved 81% accuracy. This means that in most cases, the model correctly identified the confidence level. This result supports the use of visual cues in candidate evaluation.

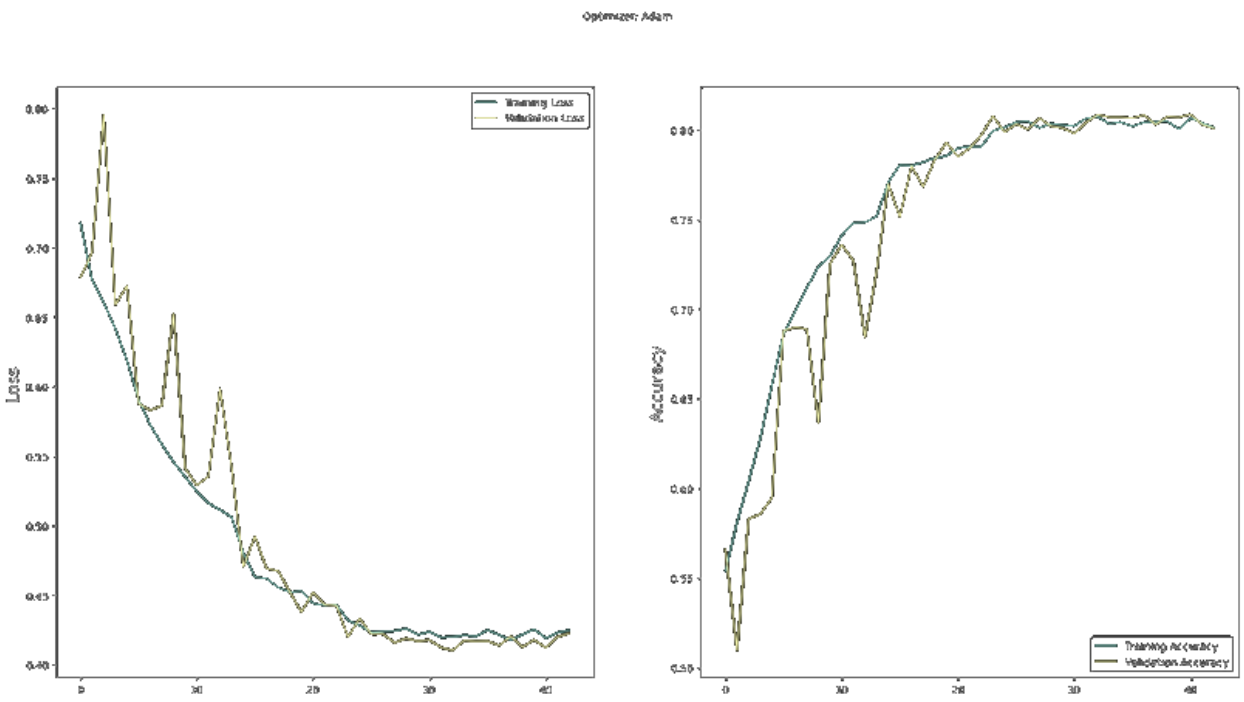

Figure 6. Confidence Prediction Loss and Accuracy Epoch-wise

5.3 System Latency and Efficiency

We also measured the system's speed. It was important to ensure real-time response. The system showed low latency. On average, the next question was generated and spoken within **2 seconds** after the candidate stopped speaking. The recruiter panel was also updated in real-time. Interview results, including scores and confidence levels, appeared in the dashboard within **two minutes**. This ensures timely access to interview data for decision-making.

Component	**Metric**	**Result**
Scoring Evaluation	Correlation with Human Scores	0.79
Confidence Prediction	Classification Accuracy	81%
Response Latency	Time to Next Question	< 2 seconds
Recruiter Panel Updates	Dashboard Sync Time	< 2 minutes

Table 2. Summary of the performance benchmarks.

5.4 Overall System Evaluation

The system was tested across multiple sessions. It performed consistently. All core features worked without major delay or error. Both technical and behavioral assessments were completed within each session. The accuracy and response speed were within acceptable industry standards. No feedback was provided by an industrial partner, but the internal tests confirm system readiness.

Chapter 6

Industrial Impact and Integration

This section discusses how the system can be used in real-world industry settings. It covers usability, technical readiness, deployment possibilities, and challenges. We also include feedback received from industry mentors during the development process.

6.1 Usability in Existing Workflow

The system is designed to fit into current hiring workflows. It does not require major changes. Recruiters can create job posts using the panel. They can schedule interviews and view reports after sessions. The system automatically evaluates candidates and generates scores. Recruiters do not need to listen to the whole interview. This saves time and reduces effort. The recruiter panel gives a summary of each session. It shows scores, confidence levels, and overall performance. These stats help recruiters make better decisions. The system supports both technical and behavioral evaluation. It ensures that candidates are assessed fairly. It also reduces human bias in scoring. This makes the interview process faster and more consistent. The system can easily be added to existing workflows with minimal training.

6.2 Technical Deployment Readiness

The system has been built with real-time performance in mind. It runs smoothly on local servers and standard hardware. The backend is developed in Django. The frontend uses React. Each module is independent and connects through APIs. These APIs are well-documented and easy to manage. The system supports both audio and visual inputs. All models have been optimized for speed and memory usage. The speech recognition, text processing, and facial analysis work in near real-time. The system can be hosted on cloud services if needed. It is ready for deployment in small and medium-sized companies. With minor scaling, it can support larger setups too. Regular updates and logging features are already included. This helps monitor the system after deployment. Overall, the system is technically complete and deployment-ready.

6.3 Licensing and IP Considerations

This project is academic and was developed for research purposes. There is no commercial license or registered intellectual property at this stage. However, the code and models can be shared under academic or open-source licenses. Future versions may require legal review if the system is commercialized. Institutions or startups interested in using this system can discuss licensing later. Intellectual property rights may be handled by the university if needed. Any reuse or modification of this system should follow fair-use policies. The current focus remains on academic contribution and proof of concept. Licensing and IP will be addressed based on future collaboration needs.

6.4 Integration Challenges and Solutions

We faced several challenges during integration. The biggest issue was combining multiple input types. Audio, text, and video had to be processed in sync. This required careful time handling. In early tests, response delays were high. To solve this, we used lightweight models and optimized code. We also faced problems with facial detection. Poor lighting caused inaccurate results. We

improved this by preprocessing the frames. Another issue was lack of diverse data for confidence prediction. The training set was limited. We handled this using data augmentation. This improved accuracy without extra data collection. Integration across modules also caused some bugs. These were resolved with proper testing and API checks. Each problem taught us more about real-world deployment needs.

6.5 Feedback from Industry Mentors

We shared the system details with academic and industry mentors. They gave helpful suggestions. Mentors appreciated the modular design. They found the recruiter dashboard useful. The automatic evaluation was seen as a strong feature. It reduces manual effort in shortlisting. However, some concerns were raised. One issue was domain dependency. The system works best in computer science topics. For other fields, models may need retraining. Another point was handling unclear answers. The system struggles with vague or off-topic responses. Mentors also mentioned the need for more diverse testing. We noted all feedback carefully. Based on their input, we made small improvements. We added clarification handling and improved model thresholds. The feedback was useful in shaping the final version.

Chapter 7

Discussion

This section presents a review of what the project aimed to achieve and what actually happened. It explains how the system performed and where it did not meet ideal conditions. The discussion reflects on why the project matters and what could be improved in the future. The main goal was to build an AI-based interview system. It was designed to evaluate candidates in a fair and smart way. We wanted to assess both technical answers and confidence levels. The aim was to reduce manual work and make hiring more efficient. We focused on academic interviews in computer science. Our system was expected to match human scoring and run in real time. The system performed well in most cases. It generated questions correctly. It evaluated candidate answers using NLP models. It predicted confidence from facial expressions. Scores from the system were close to human evaluation. The correlation was 0.79, which is a strong match. Confidence prediction reached 81% accuracy. System speed was also acceptable. Next questions were asked quickly. Dashboard updates were made within two minutes. These results show that the system worked as planned in core areas.

These outcomes are useful. They show that automated interviews can support human recruiters. The system helps in shortlisting based on clear metrics. It reduces the time needed for interviews. It also adds objectivity to evaluation. This can be useful in academic hiring and entry-level screenings. The results match industry expectations like accuracy, fairness, and quick feedback. Still, there were some limits. The system was tested only on computer science topics. It may not work as well in other fields. The models were trained on limited data. Especially in the confidence module, the dataset was small. This affects generalization. Lighting and camera quality also impacted face detection. Some candidate responses were unclear. The system struggled to handle vague or noisy answers. These problems reduced accuracy in a few cases.

There were things we could not do. We did not test the system in real job interviews. We had no access to large industrial datasets. No feedback was collected from actual recruiters or candidates. These steps could have improved our evaluation. But time and resource limits made this hard. We also could not support multiple languages. The system only works in English for now. Despite these issues, the project achieved its main goals. It showed that AI can assist in structured interviews. It provided a working prototype with key features. It stayed close to industry standards. The system may not be perfect, but it is a strong base for further work. This matters because hiring is a slow and costly process. A smart interview system can help save time and improve decisions. With more training data and real-world testing, the system can be improved further. It may also be adapted for other domains and more diverse users.

Chapter 8

Conclusions

This chapter concludes the project. It gives a short summary of the problem, our approach, and the final results. It reflects on what was achieved and how the system meets the original goals. The main problem was the slow and biased process of academic interviews. Traditional methods take time. They also rely heavily on human judgment. This makes the process hard to scale and sometimes unfair. We aimed to build an AI-based system to solve this. The system should conduct interviews, evaluate answers, and predict confidence. It should work in real time and help recruiters make faster and fair decisions. To solve this, we used several methods. We generated questions using LLMs. Speech was converted to text using recognition tools. NLP models handled intent detection and entity recognition. Semantic similarity was used to score answers. Facial expressions were analyzed using CNN to predict confidence. All modules worked together in a single system. Each part was tested separately and as a whole. Evaluation showed good results. Scoring had a strong correlation of 0.79 with human judgment.

Confidence prediction reached 81% accuracy. The system worked fast. It generated next questions in under 2 seconds. It updated the recruiter panel within 2 minutes. These results prove that the system can support interview processes. It meets the original objectives set at the start. The system had some limits. It was tested only in the computer science domain. It used a limited dataset. Real job interviews were not conducted. Feedback from recruiters or candidates was not collected. Still, the project achieved its core goals. In the end, the system addresses the problem stated in the introduction. It makes the interview process faster and more structured. It reduces manual work and brings more fairness. It provides a working base for future development in AI-driven recruitment and academic evaluations.

Chapter 9

Future Work

This project achieved its main objectives, but there are still areas that can be improved and extended. Some important parts were not covered due to limited time, data, and scope. Future versions can address these gaps and turn the system into a more robust and flexible solution. One major limitation was that the system was developed and tested only in the computer science domain. It did not support other academic fields such as business, engineering, or healthcare. These domains have different types of questions and expectations. To support them, the system would need domain-specific datasets and training. The question generation, scoring models, and entity recognition modules would need fine-tuning based on subject knowledge. This can make the system more versatile and helpful in a wider range of academic or industry settings.

The system also supports only English language interviews. It cannot understand or evaluate other languages. This limits its use in regions where English is not the primary language. Future work can focus on adding multilingual support. This would include training separate NLP models, speech recognition systems, and entity recognition tools for each target language. It may also require the inclusion of cultural context and local terms in the question and scoring systems. Supporting more languages will allow the system to be used globally, especially in multilingual institutions and companies. Another area of improvement is the confidence prediction module. It used a limited dataset and basic facial features. The model worked well in many cases but had trouble in low lighting and varied camera angles. To improve it, a larger and more diverse training dataset is needed. This should include people of different age groups, skin tones, and environments. Advanced face-tracking tools and better lighting control can also help. If connected with hardware like external cameras or ring lights, the system can capture more reliable visual data. This would lead to more accurate and fair confidence scoring.

The system was tested only in a controlled lab environment. It has not yet been used in actual recruitment or academic interviews. Field testing should be the next step. This means using the system during real interviews and collecting feedback from recruiters and candidates. Such testing will reveal real-world issues and user needs that were not visible during internal testing. It will also help validate system performance in noisy, unpredictable environments. Feedback collected from real users can be used to improve user interface, system flow, and model behavior. There is also potential for expanding the system's capabilities. At present, the interview flow is linear. The system asks a fixed set of questions and follows a basic path. In future versions, the flow can be made dynamic. This means generating follow-up questions based on the candidate's previous response. It will make the interview more natural and adaptive. Doing this requires improvements in dialogue management and response understanding. A better context management system can help the system ask smarter questions and keep the conversation on track.

On the product side, this system can be turned into a complete software platform. With a proper frontend, backend, and admin tools, it can serve as an interview-as-a-service solution. Schools, universities, and small companies can use it to screen candidates easily. This would require licensing models, user management features, and proper data security. Support for GDPR or other

local data privacy laws would also be important. Collaborations with industry partners can help shape the system into a deployable product. They can provide domain data, field testing environments, and even funding for further development. From a research point of view, the project can explore additional features. These include bias detection in AI scoring, fairness analysis, and support for candidates with disabilities. Ethical concerns in automated hiring and AI assessments are also important. Further studies can be done to see how the system performs across genders, cultures, and education levels. These research directions will make the system more inclusive and reliable for real-world use. In summary, the system provides a good foundation. However, there are many ways to improve it. Future work should focus on expanding domain support, improving model accuracy, adding language options, and testing in real scenarios. It should also consider deployment models, licensing, and user feedback. These steps will help turn this project from a research prototype into a practical and trusted tool.

References

[1] [Online]. Available: https://www.hirevue.com/. [Accessed 24 10 2024].

[2] A. a. A. A. a. D. H. a. D. M. a. A. M. Naeiji, "Question Generation Using Sequence-to-Sequence Model with Semantic Role Labels".

[3] S. {. F. a. A. R. a. Adiwijaya, "Analysis of LLMs for educational question classification and generation," *Computers and Education: Artificial Intelligence.*

[4] Naeji, "Question generation using sequence-to-sequence model with semantic role labels.".

[5] M. F. a. A. H. a. J. A. R. a. K. N. a. B. S. S. Bashir, "Subjective Answers Evaluation Using Machine Learning and Natural Language Processing," *IEEE Access,* pp. 158972-158983.

[6] N. A. a. B. M. a. S. H. M. Alnajem, "Siamese Neural Networks Method for Semantic Requirements Similarity Detection," *IEEE Access,* pp. 140932-140947, 2024.

[7] J. S. J. a. A. G. Guerra, "Educational Evaluation with Large Language Models (LLMs): ChatGPT-4 in Recalling and Evaluating Students' Written Responses," *J. Inf. Technol. Educ. Innov. Pract.,* p. 2.

[8] L. M. a. R. C. K. Chandrapati, "Descriptive Answers Evaluation Using Natural Language Processing Approaches," *IEEE Access.*

[9] D. J. L. a. M. K. a. S. S. a. M. A. Walker, "Automatic Optimization of Dialogue Management".

[10] S. L. a. D. L. a. D. D. a. K. J. M. a. F. R. a. L. D. Harper, "An Architecture for Dialogue Management, Context Tracking, and Pragmatic Adaptation in Spoken Dialogue Systems".

[11] R. M. R. &. V. R. E. (. Ponnaboyina, "Smart recruitment system using deep learning with natural language processing. In Intelligent Systems and Sustainable Computing: Proceedings of ICISSC 2021,Singapore: Springe".

[12] [Online]. Available: https://www.linkedin.com/pulse/artificial-intelligence-hiring-how-tech-reshaping-recruitment/. [Accessed 11 09 2024].

[13] "Planning First, Question Second: An {LLM}-Guided Method for Controllable Question Generation".

[14] [Online]. Available: https://www.zippia.com/advice/job-interview-statistics.. [Accessed 10 01 2025].

[15] [Online]. Available: https://standout-cv.com/usa/job-interview-statistics-us. [Accessed 19 01 2025. [Accessed 11 12 2025].

[16] K. &. N. H. T. Taghipour, "A neural approach to automated essay scoring. In Proceedings of the 2016 conference on empirical methods in natural language processing.".

[17] M.-W. C. K. L. K. T. J. Devlin, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.".

[18] Y.-X. W. a. M. Hebert, "Model recommendation: Generating object detectors from few samples,"2015".