2023 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)

Resume Building Application based on LLM (Large Language Model)

Rafael Jace Sunico
SFSU
San Francisco, USA
rsunico@mail.sfsu.edu

Shubh Pachchigar
SFSU
San Francisco, USA

Vinay Kumar
SFSU
San Francisco, USA

Ishika Shah
SFSU
San Francisco, USA

Jingyi Wang
SFSU
San Francisco, USA

Isabel Song
SFSU
San Francisco, USA

***Abstract*—Amid the highly competitive job market, creating an effective resume is vital but often difficult, particularly for students from underprivileged backgrounds with limited career development support. To mitigate this, we introduce a novel resume building application that employs the Large Language Model (LLM) to aid students in composing their first resumes. The application comprises three modules: Resume Generation, Resume Assessment, and User I/O. The Resume Generation module utilizes prompt engineering to produce resume bullet points, while the Resume Assessment module evaluates these bullet points for potential enhancements. The User I/O module simplifies user interaction by accepting free-style plain English as input and displaying the generated bullet points as suggestions. We have developed a prototype application that demonstrates the effectiveness of these functionalities while emphasizing ease-of-use. We have also confirmed that the content generated for resumes adheres to the benchmarks of high-quality standards. As future work, we aim to carry out usability testing with real students to further evaluate the application's utility in educational environments.**

Keywords—resume building; prompt engineering;LLM (Large Language Model)

I. INTRODUCTION (HEADING 1)

Crafting a compelling resume is a fundamental yet daunting task for job applicants, especially for university seniors preparing to enter the permanent workforce for the first time. Regrettably, most public schools in the US, which serve a substantial number of students from underprivileged communities, face a dearth of resources to offer adequate career services to these students. These institutions often fail to provide sufficient career counseling, an indispensable element of professional growth. For instance, San Francisco State University (SFSU), a member of the CSU (California State University) system, which is the country's largest public school system, only has a few full-time career counselors for its community of 25,000 students. Due to the scant career support from their schools, these students often have to invest an unnecessary amount of time and effort into crafting their resumes. Or this often results in poorly-prepared resumes for their desired roles, leading to unsuccessful job searches. Our resume building application steps in to bridge this gap between students and the scarce career counseling support available, providing a much-needed boost, particularly to underrepresented groups. The application could potentially enhance their educational prospects and foster a marked improvement in their societal standing.

Previous work on online recruitment has investigated various methodologies to enhance job application and selection processes. These studies have primarily revolved around resume analysis, job category classification, and the generation of interview questions and evaluation of their answers. However, they haven't addressed the specific needs related to assist resume building. Furthermore, none of these solutions have explored the use of the Large Language Models (LLMs) [1] which are swiftly and profoundly transforming numerous sectors, including the field of education.

In this paper, we present a novel application that leverages the LLM to facilitate the resume building process. The application takes straightforward natural language input from users, reflecting their past experiences and skills, and transforms these inputs into suggested bullet points for their resume. It also offers users the ability to further modify these generated suggestions, allowing for necessary improvements. By involving users in the resume building process, our application ensures that the final resume genuinely reflects the users' work history and capabilities, thus preventing any misrepresentation of the users' skills. This engagement also helps prevent any hallucination problems [1] that might occur with the LLM, where the model might generate incorrect responses.

The main design obstacle of our application is to produce high-quality resume bullet points from user-friendly inputs like free-style plain English, ensuring they comply with resume guidelines [2][3][4]. These guidelines stress the appropriate use of soft skills to truly represent users' skill sets, and the articulation of resume bullets in a succinct STAR format which encompasses situation, task, action, and results. This is essential as many students, especially those from STEM (science, technology, engineering, and math) disciplines, often concentrate on hard skills, such as specific engineering abilities

ISBN: 979-8-3503-0611-8/23/$31.00 ©2023 IEEE

like JavaScript, while undervaluing equally vital soft skills, like communication and teamwork, which employers highly value. Ensuring the generation of high-quality resume bullet points from LLM, which effectively depict soft skills in the STAR format, heavily relies on the precise design of prompts [1]. These prompts serve as input to LLM, playing a crucial role in guiding the generation of optimal outputs from the LLM.

The rest of the paper is organized as follows. In Section II, we delve into previous work on online recruitment solutions and existing prompt engineering methods of LLM. In Section III, we describe the software architecture of our application, encompassing resume generation, resume assessment and user I/O, along with our prompt designing strategy that facilitates the generation of high-quality resume bullet points. Section IV demonstrates our prototype application, serving as proof of concept, and evaluates the quality of the generated resume bullet points. Finally, Section V concludes the paper by discussing potential future research directions to extend this work in aiding students in real academic environments.

II. Related Work

A. Online Recruitment Solutions

A variety of online recruitment solutions are at hand to enhance the process of job application and hiring. These tools cover aspects like screening resumes, recommending candidates for a specific job category, preparing interview questions, and evaluating responses, all designed to handle an extensive pool of applicants efficiently in digital spaces.

Luo [5] introduced ResumeNet, a neural network-based approach for resume screening. Each subnetwork within ResumeNet focuses on a different section of a resume, assessing its content and examining the consistency between multiple sections of resume. The resume's final quality is determined by the scores of these individual sections as well as their collective consistency. Luo demonstrated the effectiveness of ResumeNet by showcasing its ability to accurately analyze resumes with accuracy rate of up to 82%.

Pant [6] developed a method to pull information from resumes of varying formats, styles, and fonts. This method capitalized on the location of certain types of information in resumes and their associated keywords. As an example, skill sets usually appear at the end of a resume with the label "Skills". Pant demonstrated his machine learning model successfully extract the appropriate skill sets from five distinct resumes. In a similar vein, Sajid [7] created a system to parse resumes across different formats, extract vital information, and compile it into structured output. His method utilized Naïve Bayes networks [8] to identify multiple text blocks in resumes and Named Entity Recognition (NER) [9] to highlight crucial labels like "Education", "Experiences", and "Skills".

Ramaraj [10] devised a system for classifying resumes based on LinkedIn [11] applicants' profiles. He gathered data from real-time LinkedIn profiles, extracted features for each profile, and then trained a Convolutional Neural Network (CNN) [12]. He confirmed that this model could accurately sort each profile into one of 27 job categories with an accuracy rate close to 70%. Pant, Sajid and Ramaraj 's efforts in gleaning key facts from resumes and LinkedIn profiles significantly aid subsequent processes like candidate ranking and recommendations for a specific job category.

Purohit [13] put forward an innovative software framework, Jaro, a chatbot designed for conducting interviews. Jaro scrutinizes resumes and CVs, formulating an appropriate set of interview questions based on this analysis. The chatbot conducts the interview, evaluates the sentiment of the interviewee's responses, and compiles a report which is then saved in a database. A proof-of-concept demonstration validated that Jaro could effectively reduce the duration of the recruitment process. In a parallel development, Pandey [14] presented another interview chatbot framework capable of screening resumes, carrying out interviews, and evaluating interview outcomes. This chatbot employs Natural Language Processing (NLP) to extract keywords from resumes and allocate fitting job roles based on the match of these keywords. To enrich the interview experience, it automatically crafts questions derived from the applicant's skills, experiences, and potentially from their ongoing interview responses. The chatbot also acts as a proctor during the interview, discouraging any potential cheating. In the end, it calculates a score derived from the correlation between the job description and the candidate's performance during the interview. The chatbot framework demonstrated its capacity to efficiently narrow down candidates, which markedly lessens the time required for interviewing and eliminates any biases from the interview procedure.

The increase in online job applications has led to a flood of submissions, making the use of online recruitment technologies indispensable in the human resources field. Even though LLM hold considerable potential for a range of NLP tasks, which are vital for this domain, we haven't found any existing work which employs LLMs, especially in the area of facilitating resume generation.

B. Prompt Engineering

While traditional NLP tasks often need a deep learning architecture that's fine-tuned for a specific task like NER, summarizing, or semantic parsing, emerging LLMs like GPT [1], Llama [15], Bard [16], and many other open source LLMs from Hugging Face community [17] employ a uniform deep learning architecture and they are trained on a vast amount of data to acquire general language features [1] [15] [16] [17]. These pre-trained LLMs can then be customized for a variety of downstream tasks, such as NER, by using prompts. These prompts, which can be in the form of a question or a template, guide the LLM to generate responses optimal for each specific downstream task. The formulation of these prompts for each particular task is critical in driving optimal performance.

Designing effective prompt templates involves several considerations, including the type of LLM used, the choice between manual and automatic prompt designs, and the decision between single and multiple prompts [18]. In a manual design, a human intuitively engineers a prompt template for each specific task, such as NER. Conversely, in an automatic design, the prompt template is determined automatically by mining the relationship between the prompt and the anticipated output. While the manual design method is straightforward,

automatic design can help reduce human errors, although it may add to the complexity of mining and computational demands.

A single-prompt method only needs one prompt for the LLM whereas a multi-prompt method calls for several prompts [18]. These are assembled by merging individual templates using various techniques. For example, the prompt ensemble technique [19] creates a collective of diverse basic single prompt templates and combines the results from these different prompts, possibly through methods like averaging or majority voting. By employing an ensemble technique, this approach capitalizes on the combined advantages of multiple prompts, thereby alleviating the challenge encountered by the single-prompt method in identifying the most effective prompt. This task can often be intricate and complex.

Another multi-prompt approach, known as prompt composition [20], divides a task into several sub-tasks, with each sub-task having its own engineered sub-prompt. Explain-based prompting employs the idea of chain-of-thought in its prompt composition [21]. By incorporating a series of reasoning steps into the prompts, it enhances the LLMs' reasoning capabilities. Wei [21] found that this chain-of-thought prompting approach significantly improved the performance on NLP tasks such as arithmetic, symbolic, and commonsense reasoning. To further simulate human thought processes, Chang [22] proposed the Socratic prompting method. This method is grounded in Socratic reasoning [23], which is a form of cooperative argumentative dialogue between individuals, designed to stimulate critical thinking. In the same way that Socratic reasoning employs multiple techniques such as definition, examples, elenchus, dialectic, maieutics, generalization, and counterfactual reasoning, the Socratic prompting method integrates each of these techniques into its prompt. Chang demonstrated the effectiveness of Socratic prompting for the Critical Reading (CRIT) task which evaluates the quality and credibility of written materials [22].

These prompting strategies have been developed for a diverse range of tasks. However, no previous work has been specifically devoted to designing prompts for resume creation. In the next section, we will describe the software architecture of our resume-building application and explain our strategy for designing an effective prompt for this purpose.

III. Our Application

In this section, we introduce our app, an application created for university students. We will detail the software architecture and prompt design strategy we have employed in the application.

A. Software architecutre

Our architecture was formulated by pinpointing the essential requirements needed to construct high-quality resume bullet points. TABLE 1 outlines the key requirements to consider to generate high-quality resume bullet points. These requirements were guided by many career centers at universities including [2]. To fulfill these stringent standards, our application is split into three distinct modules: 1) Resume Generation, 2) Resume Assessment, and 3) User I/O. Each of these modules is specifically designed to ensure modularity of the application.

The Resume Generation module is responsible for creating high-quality resume bullet points, facilitated by a sophisticated prompting technique that we will detail in the next subsection. Acknowledging the rapid advancement of LLMs, we've implemented a separate interface layer inside the Generation module, as depicted in Fig. 1, that interacts with the LLM independently from the rest of the module. This configuration enables easy interchangeability among different LLMs as they progress. Currently, we employ GPT-3.5 Turbo [1], a potent and cost-effective LLM, as of writing this paper. The Generation module uses the free-style natural language input of experience and skills captured from the User I/O module and generates corresponding resume bullet points.

For quality enhancement of the generated resume bullet points, we have integrated a Resume Assessment module into our design as shown in Fig. 1. This module reviews the generated resume bullet points considering various factors such as content quality, usage of action verbs, and filler words and length, as guided by [2][3][4] and outlined in TABLE 1. Given the proven effectiveness of LLMs in classification and assessment tasks in numerous previous studies [1][15][16], we employ an LLM for this through the use of prompting. The detailed prompting method will be explained in the next subsection. The module also maintains a separate interface layer with LLM to ensure seamless interchangeability. Based on the assessment feedback, the Generation module may, if necessary, re-prompt the LLM to refine the bullet points, thereby enhancing the overall resume quality.

The User I/O module has the role of capturing the user's inputs concerning skills and experiences and presenting the generated resume bullet points. The user input is relayed to the Generation module, which in turn creates up to 10 bullet points. These bullet points are then displayed to the users through the I/O module. Users can then choose bullets that best depict their background and conveniently modify and enhance their resume as needed. By isolating the User I/O module, we can easily extend our application to different I/O formats such as various display layouts encompassing PC apps, mobile apps, or even chat-bot user interfaces, aiming to actively engage with the younger generation of students. Presently, we have opted for a PC app format for productivity of younger generation.

TABLE II presents a sequential representation of our application's functionality through pseudocode.

TABLE I. REQUIREMENTS OF A RESUME

Id	Name	Description
R1	Content	Each bullet point should adhere to the STAR format, where 'S.T.' signifies Situation or Task, 'A.' represents Actions, and 'R.' denotes Results [4].
R2		The collection of all bullet points should reflect each of the soft skills such as communication, leadership, teamwork, and critical thinking [2][3].
R3	Action verbs	Each bullet point should commence with an action verb, for instance, achieved, attained, etc [2].
R4	Filler words	Avoid the use of filler words such as 'Responsible for', 'Best-of-breed', etc [24].

Id	Name	Description
R5	Length	The length of a resume bullet point should be between 1~2 lines [25].

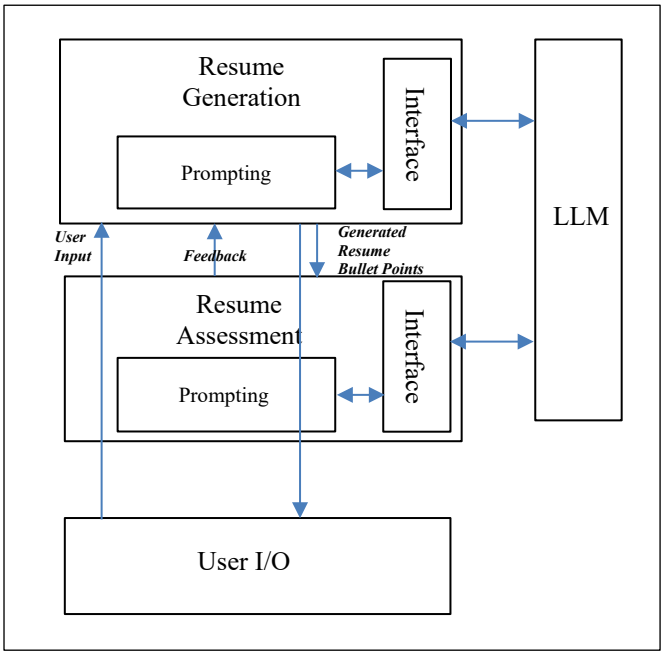

Fig. 1. High-level architecture of the resume building application

TABLE II. PSEUDOCODE

Step	
	Input : Experience and skillset articulated as free-style input; ***Output*** : Resume bullet points
#1	*User IO module*: Take user input related to his/her experiences and skillset.
#2	*Resume Generation module*: Issue a prompt to LLM which incorporates the user's input and the requirements shown in TABLE 1.
#3	*LLM*: The LLM creates a resume bullet point.
#4	*Resume Assessment module*: Issue a prompt to LLM to assess the quality of the generated resume bullet.
#5	*LLM*: The LLM assesses the quality against the criteria detailed in TABLE 1.
#6	*Resume Assessment module*: If improvement is needed, signal Resume Generation module to reissues the prompt. Otherwise, go to step #8
#7	*Resume Generation module*: issue a prompt to enhance the resume
#8	*LLM* : The LLM regenerates a resume bullet point.
#9	*User IO module* : Present the resume bullet to the users.

B. Prompt Design

Our strategy for generating resume bullet points is anchored in simplicity and high-quality. We choose a manual hand-engineered prompt design method tailored for this task. This approach is intuitive, and requires minimal computational resources.

For generating high-quality resume bullet points, we employ a multi-prompt approach which partitions the task into three sub-tasks. The first sub-task is to create bullet points that satisfy the criteria of R1 and R2 as detailed in TABLE 1. The following sub-tasks handle post-processing of the initially generated bullet point to verify if it meets all the criteria of R1~R5 shown in TABLE 1 and enhance it, if necessary.

We aim to mimic the cognitive processes of human experts. To this end, we incorporate three types of Socratic reasoning methods that are critical in resume creation: definition, examples, and cross-examination. To fulfil requirements R1 and R2, Resume Generation module issues a prompt incorporating clear definitions of the STAR format and essential soft skills, namely communication, leadership, teamwork, and critical thinking. P1 prompt for this was shown in TABLE III where these definitions are denoted as {skill_definitions_str} and {STAR_definition_str}. Additionally, we incorporate practical examples of the STAR format within the P1 prompt, denoted as {STAR_example_str} to better illustrate to LLM.

To ensure the quality of the initially generated bullet point, the Resume Assessment module conducts a cross-examination using P2-1 to P2-5 prompts as illustrated in TABLE III. This step verifies that the generated content meets each of requirements R1~R5. If it fails to meet any requirement, the Resume Assessment module triggers the Resume Generation module which then issues any of P3-1~P3-5 prompts, depending on the assessment result.

TABLE III lists all the prompts which we design for resume building application. TABLE IV shows five examples of generated bullet points with a given set of user input.

TABLE III. PROMPT DESIGN FOR RESUME BUILDING APPLICATION

Prompt	Dialogue	Used method
P1	You are program that reads student submitted input of {input_in_free_format}. Understand the following skill definitions: {skill_definitions_str}. Generate a resume bullet point adhering to the following structure: {STAR_definition_str}. Here are some examples for your reference: {STAR_example_str}.	definitions, examples
P2-1	Examine the generated bullet point to see if they follow STAR format.	cross-examination, definition
P2-2	Examine the generated bullet point to see if they have at least one of skills in {skill_definitions_str}.	
P2-3	Examine if the generated bullet point has an action verb defined in {strong_verbs_str_str }.	
P2-4	Examine if the generated bullet point has any of {filler words_str}.	
P2-5	Examine if the generated bullet point is {right_length}.	
P3-1	Rephrase a resume bullet point adhering to the structure of {STAR_definition_str}.	definition
P3-2	Rephrase a resume bullet point with following skills of {skill_definitions_str}.	
P3-3	Replace its verb for strong action verb synonym found in {strong_verbs_str_str}.	
P3-4	Remove any words found in {filler words_str}.	
P3-5	Make a bullet length into {right_length}.	

TABLE IV. EXAMPLES OF GENERATED BULLET POINTS

Role	Dialogue
Input	In my class, I put together a trip advisor app using React.js. I used Figma for the UI design. Following that design, I worked on the front-end development of the app. I ran into a snag with the app taking too long to launch. After some profiling, I found the bottleneck and managed to cut down the launch time by 30%. I was teaming up with two other students, and we talked a lot. It was a blast learning from each other. They've been my best buds ever since!
Generated Resume Bullets	1. Identified and resolved performance issue causing slow launch time of trip advisor app developed with React.js, resulting in a 30% improvement in app launch speed. 2. Collaborated closely with two fellow students, fostering a strong and supportive team environment that facilitated efficient learning and problem-solving. 3. Utilized Figma for UI design of trip advisor app, ensuring visually appealing and user-friendly interface. 4. Communicated effectively with team members to address challenges and discuss solutions, ensuring a cohesive approach to app development and successful problem-solving. 5. Showed leadership by taking charge of the app's front-end development, utilizing React.js and Figma to design a user-friendly interface that positively impacted the app's overall usability and user experience.

IV. EVALUATION

In this section, we demonstrate our prototype application and explain how it effectively encompasses all the functionalities detailed in Section III. We also highlight its ease of use and ensure that the resume bullet points produced by our application comply with desired resume quality standards.

A. Proof-of-concept application

We've designed our application with a user-friendly interface that simplifies the resume building process, requiring only natural language inputs from the user, as depicted in Fig. 2. Utilizing these, the application suggests suitable bullet points for the resume. Fig. 2 shows an input box to show input and a text editing box on the same page, enabling users to easily modify or re-enter input for fresh suggestions. Moreover, users can tailor the recommended bullet points to better reflect their personal backgrounds, as shown in Fig. 2.

Considering that crafting an initial resume might take up to a month for students without professional guidance such as career counseling, our prototype application presents an efficient solution for constructing a resume.

B. Quality of resume bullet points

In this sub-section, we explore the quality of resume bullet points generated, focusing on requirements R1 and R2 from TABLE 1. These are instrumental in shaping resume content. Other factors like R3~R5 can be easily achieved, and their compliance is easily verified.

For the evaluation, we employ GPT 3.5 turbo to classify and examine whether the generated bullet points align with the

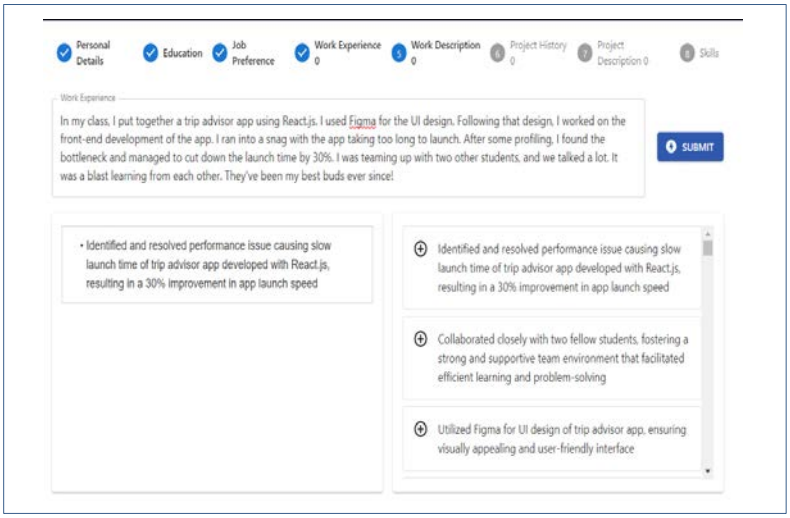

Fig 2. Prototype Application

STAR format and incorporate the required soft skills. Although we cross-examine the bullet point in its building process, this does not guarantee the desired outcome. Therefore, we evaluate the final results here.

We conduct our tests using five distinct sets of input, we generate 5 unique resume bullets, as showcased in TABLE IV. In total, we assess 100 (=5*4*5) resume bullet points to validate their compliance with the STAR format, as shown in TABLE VI. Impressively, our analysis revealed that no less than 93% of the bullet points conformed to the STAR format.

We further probed into the content to examine the number of resume bullets that succeeded in fulfilling each of the soft skills, a critical aspect of any professional resume. This examination is laid out in TABLE VII. Note that a single bullet point may meet several soft skill requirements. Our findings were reassuring, with each of the soft skills being well represented in the generated content. These test results underscore the effectiveness of our approach in generating high-quality resume content.

TABLE V. DIFFERENT USER INPUTS

Input #1	In my class, I put together a trip advisor app using React.js. I used Figma for the UI design. Following that design, I worked on the front-end development of the app. I ran into a snag with the app taking too long to launch. After some profiling, I found the bottleneck and managed to cut down the launch time by 30%. I was teaming up with two other students, and we talked a lot. It was a blast learning from each other. They've been my best buds ever since!
Input #2	Oh man, working on the backend was a trip! I got to mess around with Flask and connected it to a MongoDB database, even used MongoDB Atlas for easier management. Had to roll up my sleeves for some serious database optimization before deploying. And you won't believe it, even nailed the CI integration to keep things smooth for updates and changes. First time knocking out a whole project from start to finish.
Input #3	Teaming up with two classmates to work on a mentoring app was a blast. We tackled the front-end using React.js and got the Video Chat rolling with WebRTC. We didn't skimp on cybersecurity either, making sure to encrypt sessions, use a VPN, and throw in some anti-malware for good measure.

	Teamwork made the dream work! We enjoyed the pizza every night.
Input #4	Dug into data mining to predict traffic volume around campus, like figuring out the busiest times at the main intersection. Leveraged ML and honed in on classification with XGBoost and SVM to get the job done, making it easy to see patterns and trends. But it wasn't just about spitting out predictions. Used Explainable AI and SHAP analysis to make sure the results were clear and made sense to everyone, like showing why 2 p.m. on Tuesdays was a traffic nightmare. Turned the tricky stuff into a cakewalk, and the project turned out super solid, helping folks avoid traffic jams around campus!
Input #5	Dove into the Agile world, taking on some project management and even rocking the scrum master hat. Set up a GitHub Project to keep everything organized, making it a breeze to track our progress. Threw in some team-building activities to keep the vibes positive and everyone on the same page. Even organized a virtual Game Day, where we all played online games together. Made managing the project not just smooth, but a whole lot of fun too!

TABLE VI. CONFORMANCE OF STAR FORMAT

	Total bullet points	The number of bullet points in STAR format	Conformance Ratio
Input #1	20	19	0.95
Input #2	20	19	0.95
Input #3	20	20	1
Input #4	20	18	0.9
Input #5	20	18	0.9

TABLE VII. CONFORMANCE SOFT SKILLS

	Total bullet points	The number of bullet points conforming soft skills			
		comm.,	Leadership	teamwork	critical thinking
Input #1	20	9	8	9	11
Input #2	20	5	6	9	10
Input #3	20	6	7	9	8
Input #4	20	7	7	6	8
Input #5	20	14	11	11	7

V. CONCLUSION

This paper presents the development of a proof-of-the concept resume building application targeted at young professionals who have limited access to career resources. The application employs a simple user interface and generates resume bullet points. Three key modules—Resume Generation, Resume Assessment, and User I/O—have been designed with modularity, enabling easy expansion with additional upcoming LLM or I/O interfaces. By incorporating sophisticated prompting technology, the application harnesses the full potential of LLM, producing high-quality resume content. For future work, we intend to accept more unstructured input from students, including social networking sites and any pre-existing documentation. This approach aims to accurately represent their experience and reduce the effort required to compose a resume. We also we plan to test the application to students to assess its impact on their job search journey. This step will provide valuable insights into the application's helpfulness and its potential to support students in their career pursuits and academic endeavors.

REFERENCES

[1] OpenAI, "OpenAI," Apr. 25, 2019. https://openai.com/.

[2] "Berkeley Career Engagement | Career Readiness Workbook," https://issuu.com/calcareercenter/docs/cr_workbook_19-20__2_?fr=xKAE9_zU1NQ (accessed Sep. 26, 2023)

[3] D. Hua *et al.*, "Enhancing Student Career Readiness through Skills Infusion," Information Systems Education Journal, vol. 20, no. 5, Dec. 2022.

[4] "How To Create a STAR Method Resume (With Examples)," Indeed Career Guide. https://www.indeed.com/career-advice/resumes-cover-letters/star-method-resume.

[5] Y. Luo and H. Zhang, "ResumeNet: A Learning-based Framework for Automatic Resume Quality Assessment," IEEE International Conference on Data Mining, 2018.

[6] D. Pant, D. Pokhrel and P. Poudyal, "Automatic Software Engineering Position Resume Screening using Natural Language Processing, Word Matching, Character Positioning, and Regex," International Conference on Advanced Systems and Emergent Technologies, 2022.

[7] H. Sajid *et al.*, "Resume Parsing Framework for E-recruitment," International Conference on Ubiquitous Information Management Conference, 2022.

[8] Wikipedia Contributors, "Naive Bayes classifier," Jun. 17, 2019. https://en.wikipedia.org/wiki/Naive_Bayes_classifier.

[9] Wikipedia Contributors, "Named-entity recognition," Feb. 08, 2019. https://en.wikipedia.org/wiki/Named-entity_recognition.

[10] S. Ramaraj, V. Sivakumar and G. Kaushik Ramnath, "Real-Time Resume Classification System Using LinkedIn Profile Descriptions", IEEE CISPSSE, 2020.

[11] LinkedIn, "LinkedIn," 2023. https://www.linkedin.com/.

[12] Wikipedia Contributors, "CNN," Feb. 2019. https://en.wikipedia.org/wiki/CNN.

[13] J. Purohit *et al.*, "Natural Language Processing based Jaro-The Interviewing Chatbot", Third International Conference on Computing Methodologies and Communication, 2019.

[14] R. Pandey *et al.*, "Interview Bot with Automatic Question Generation," 9th International Conference on Advanced Computing and Communication Systems, 2023.

[15] Meta, "Llama 2 - Meta AI," 2022. https://ai.meta.com/llama.

[16] Google, "Bard," 2023. https://bard.google.com/

[17] Huggingface, "Hugging Face – On a mission to solve NLP, one commit at a time.,". https://huggingface.co/models.

[18] P. Liu *et al.*, "Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing", ACM Computing Surveys, 2023.

[19] Weizhe Yuan, Graham Neubig, and Pengfei Liu, "BARTScore: Evaluating generated text as text generation," NeurIPS, 2021.

[20] X. Han *et al.*,"PTR: Prompt Tuning with Rules for Text Classification". https://arxiv.org/abs/2105.11259.

[21] Jason Wei *et al.*, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," NeurIPS, 2022.

[22] E. Chang, "Prompting Large Language Models With the Socratic Method", IEEE CCWC, 2023.

[23] A. Coumoundouros, "Internet Encyclopedia of Philosophy,". https://iep.utm.edu/republic/.

[24] Grammarly, "26 Words and Phrases to Never Use in a Résumé," Aug. 07, 2017. https://www.grammarly.com/blog/resume-words/.

[25] Enhancv, "How Many Bullet Points Should I Have Per Job on a Resume," https://enhancv.com/blog/how-many-bullet-points-should-i-have-per-job-title-on-resume