Super Rapid Annotator - GSOC24

Vedant Deepak Mehra

April 1, 2024

Summary of the Proposal

I propose developing a multimodal vision tool that can analyze and comprehend images and videos, enabling it to answer a series of questions or queries presented by the user in JSON format. The user can also customize the appropriate JSON structure for the output. In the current method, I use a pipeline architecture, consisting of a vision model (responsible for the description of input media) and a language model (responsible for the actual question answering) wrapped inside a JSON parsing tool (responsible for the structured output). This tool would enable any user to simply upload any video or image along with a set of queries (CSV) about the input media, the tool will analyze the media and provide appropriate answers to the queries in a structured format in a downloadable CSV form.

Background

Presently, the majority of the vision models do not support video inputs which defeats our purpose. We could use an image model and send every frame of the video as a single image and combine the output however, that is not an efficient approach. Such an image model would miss out on the temporal semantics of the video and thus any queries that require temporal information might be wrongly answered. Example query: "*Is the boy running?*", "*Is the taxi moving?*" etc. Fine-tuning an image-language model for benchmark performance requires strong computing power and precise embedding alignment of text and image for appropriate integration. Out of the very few video-compatible models, *Video-LLaVA* is a lightweight and efficient model, hence suitable for our task. Video-LLaVA {*paper*} aligns the video and image vector space before feeding into the language vector space, hence the language model has a better understanding of formats.

The next matter of objective is the need for structured output. A model that could directly annotate videos and images on input queries in a specific structure would prove very helpful. The structured output facilitates data interchange, simplifies processing, and enhances interoperability across various applications and systems. Currently, the major method to structure the output in the required format is to use function calls from OpenAI API, however, they are paid hence limiting the input context and the depth of the JSON structure we provide. We could use Langchain library's *outputparser* but that is not compatible with all models especially not with our vision model. Hence I propose to use an external open-source JSON parsing tool i.e. *jsonformer*. Jsonformer is a fairly simple tool and provides much room to work with the input structure and the datatypes of the output. Currently, none of the external JSON parsers are directly compatible with the vision model including jsonformer. However, since the codebase of jsonformer is open-source and simpler compared to other tools like langchain, I could gain a thorough understanding of the codebase of the jsonformer and I also believe there might be a slight possibility to modify the codebase such that we can directly use it with our vision model. For now, I propose a robust and efficient pipeline architecture where I use an extra language model as a base for the jsonformer. In such a pipeline architecture, there might be issues with context hence developing functions to use the input query itself to generate a more specific prompt rather than a general prompt like "*Explain the video in detail*" proves efficient. This also helps us limit the tokens generated for the whole query and avoids hallucinations since the model has to just focus on the specific question like "*Is the person standing?*".

Currently, the Super Rapid Annotator involves manual annotation which is very useful but the user has to manually answer the questions via buttons. However, that can be automated using a multi-modal vision

model. By implementing a multi-modal model users will be able to upload videos or images as inputs with a set of annotation queries based on the input media and the tool will return a CSV file of the response to queries based on the vision analysis. In the proposed method, there is a pipeline consisting of a vision and a language model that takes in the input video/image and custom text query. The vision analysis output is stored in a temporary file structure. The next block is the language model and an external JSON parser wrapped along with it. We prepare an appropriate schema according to the user input schema and then use it to prompt the language model. The language analyzes the previously saved output and answers the queries based on it. The JSON parser enforces the required structure with the help of the schema prepared. Hence, we can directly read the annotation query from a CSV and even save the outputs there directly. The vision model I propose to use is *Video-LLaVA* and the JSON parser is *jsonformer* along with a suitable language model. The language model I propose to use is an open-source model from hugging face, *model*. The sole reason to use this model is that it meets our objective with accurate responses and aligns with the dependency versions of the vision model.

Goal and Objectives

The objective of this research is to develop a simple multimodal-vision tool that enables any user to annotate input media in a structured format with ease and efficiency even without any coding background. Considering the rapid advancements in vision models, I believe we may find superior models for our task. Our model-independent pipeline architecture enables us to easily replace the model without affecting any functions or models in any other section, this is very helpful for a period like today where there are new models with improved benchmarks every week. I'll continuously test new vision models while comparing their results with our current one. Concurrently, I'll focus on refining and implementing our existing approach, which already effectively fulfills our objective with precision and efficiency.

A breakdown of my tasks:

- Understanding the input structure (CSV sheet) more accurately and figuring out the most efficient way to pass it to the vision model.
- Developing the function to tune the prompt for the vision model and making it dynamic to the input.
- Analyzing the output and devising the best approach to store the output in a logical manner per input.
- Testing some relevant language models for an appropriate language model such that it must align with the dependencies of the vision model and use it if necessary. The current proposed model is efficient at its task and I intend to use it by default.
- Testing the relevant JSON parsing tools available at the time and making changes if necessary. Setting up the final JSON parser with the language model. I intend to use jsonformer by default.
- Our next objective is to define a custom JSON schema for the user. I shall do that by developing a function that analyzes the input structure and defines a new JSON schema compatible with jsonformer.
- Setting up the final combined environment for the JSON parser, the language model, and the input pipeline.
- Finally generate the output of the queries in the required format by optimizing it further using custom functions.
- Developing a function for optimizing and storing the output in a CSV format along with its corresponding video address/id and the input query.
- Building the Gradio app with input as a zip file or a download URL and outputting again a CSV file containing the annotations.
- Additionally I would work on building a command line version of this tool as it would enable people to run the model locally for their private data and inputs and use the pipeline elsewhere too. The command line version would take input parameters: the media and the path to CSV for the queries.

Methods

This section includes a deeper explanation of my approach and the steps I will follow to attain the objective. A high-level structure of the work is demonstrated in the flowchart at the end of the section. I will always ensure to draft proper documentation of each step and component as I implement them. I shall also write blogs about the process and issues faced and how I tackled them.

1. **Vision Analysis Section:**

- A CSV consisting of queries will be input along with the media for annotation, I will first work on developing a function to design prompts from the input questions. A rough version of the code and output can be found below.

```
# Example of input.json
 {
     "questions": [
         {
             "description": "Is the person in the image standup?",
             "value": "standup"
         },
         {
             "description": "Can you see the hands of the person?",
             "value": "hands"
         }
     ]
 }

import pandas as pd
global step
num_of_q=1
df=pd.read_json('/content/input.json')
Prompt=[]
prompt=""
i=0

# the function
for question in df.questions:
  prompt+=((question["description"])+" KEYWORD="+question["value"]+"\n")
  i+=1
  if(i==num_of_q):
    Prompt.append("explain your answer to all the questions in context of
    the KEYWORD\n"+prompt)
    prompt=""
    i=0

# final Prompt
 [
 'answer the question in context of the KEYWORD\nIs the person in the image
    standup?KEYWORD= standup\n',
  'answer the question in context of the KEYWORD\nCan you see the hands of
    the person? KEYWORD= hands\n'
 ]
```

Listing 1: Example Prompt Drafter

- Set up the vision model after testing and comparing it with some of the relevant vision models at the time. Video-LLaVA is the vision model by default.
- The next task would be to store the output from the vision model appropriately to pass it to the language model. The format and structure must be simple to read from, later by the schema generator function. An example of the output with the corresponding question:

```
Prompt: "Answer the question in context of the KEYWORD
            Is the person in the image standup? KEYWORD=standup"
Output: "Yes, the person in the image is standing up while talking about
            the weather."
```

Listing 2: Example Output

2. JSON Parser:

- After extensive testing of JSON parsers, I've found that jsonformer efficiently achieves our objective. While OpenAI's llm API offers structured output, it's a paid service, and langchain's parser can be unreliable. Jsonformer's compatibility with any open-source Hugging Face model ensures consistent output formatting, even when it cannot provide an answer based on the prompt provided.
- Develop a function that utilizes context/keywords from the previous output to prepare an input JSON schema for the JSON parser. An initial version of the function code can be found below. This schema is of a format that is compatible with jsonformer.

```
import pandas as pd

# Initialize schema skeleton
schema={
         "type": "object",
         "properties": {
             "querie":{
               "type": "object",
               "properties": {}
             }
         }
     }

# append the input questions to get a jsonformer compatible  scema
for question in df.questions:
  new_property = {
    str(question["description"]): {
        "type": "boolean"
    }
  }
  schema["properties"]["answers"]["properties"].update(new_property)

# store it
with open('/content/jsonschema.txt', 'w') as writefile:
    writefile.write(str(schema))

#schema looks like this
 {
     'type': 'object'
     'properties': {
         'queries': {
             'properties': {
                'Is the person in the image standup?': {'type': 'boolean'},
                'Can you see the hands of the person?': {'type': 'boolean'},
                 },
             'type': 'object'
             }
     },
  }
```

Listing 3: Example JSON Schema

3. Language Model + JSON Parser Section:

- Set up the final language model after testing the latest language models, if required, combined with the the JSON parser.
- Generate outputs and store this temporary output in a temporary file because the structure of the output of the language model might not be exactly what we require.
- Hence develop a simple function to extract the exact annotations and combine them to get the final output which is to be written in the final CSV format.

```
#rough version of the function
mapped_values = {}
for question in df["questions"]:
    question_value = question["value"]
```

```
    mapped_values[question_value] = answer["answers"][question["description"
    ]]
highlight_values(mapped_values)

# final output
{
  standup: True,
  hands: True
}
```

Listing 4: Example Final Output Achieved with the initial Implementaion

4. Gradio App

- Building the Gradio app consisting of the entire pipeline. This should not be a complicated task. A simple Gradio app interface would enable all kinds of users to use the tool without any kind of code involved. A rough sketch of how I vision it:

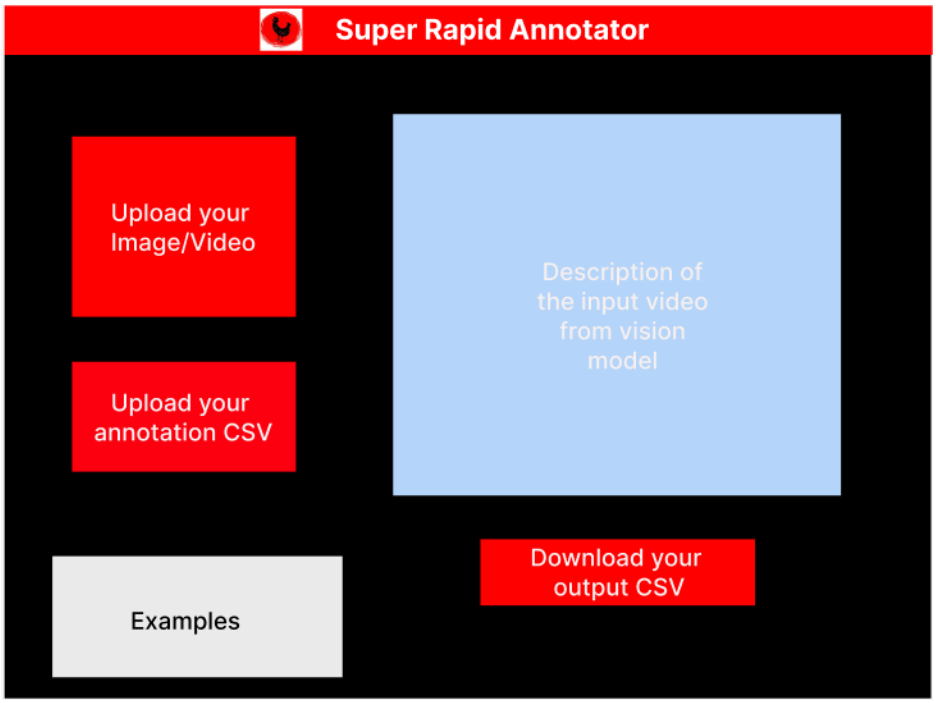


Figure 1: Gradio Interface

5. CLI Version

- Building the CLI version consisting of the entire pipeline to be able to run the tool locally. This again should not be a complicated task and would increase the impact of the project by being able to include it in other scripts.

Future Prospects:

- I believe there is a possibility to make changes to the jsonformer codebase such that it can be directly used to prompt the vision model, which it cannot currently. This way we can eliminate the intermediate language model. I have looked through the working of jsonformer and I believe this might be possible. If time persists I shall delve into that.
- We could build a modularized version of the architecture where users could choose between a set of models for the task with the help of a simple dropdown.

General Points:

- The final deliverable will be an automated end-to-end tool that takes in input media and a CSV of queries and returns a downloadable CSV of output consisting of the answers in the context of the media. I believe we can build this within the 175h deadline including some time to look into some of the future prospects after discussion with the mentors.
- I will ensure that I always update the mentors about my progress over emails or via presentations over meets. Also I shall push my code on Github regularly which is a crucial part of remote projects.
- I have mentioned that before implementing the models I would test out the newer relevant models, at the time of implementation, solely to develop a tool that meets the benchmark standards at the time of completion. This is important because of the fast pace of advancement of the language and vision models. I have a perfect set of robust working models to begin with, which meet our objective. Also, I have ensured that the compute resources required for the models are as low as possible.
- We need to standardize the amount of customizability provided to the user in terms of the JSON schema and the datatypes of the output. For example, would the user prefer a "standup: True" (boolean) or "standup: Yes the man is standing" (string)? jsonformer provides multiple options including boolean, string, number, and array.

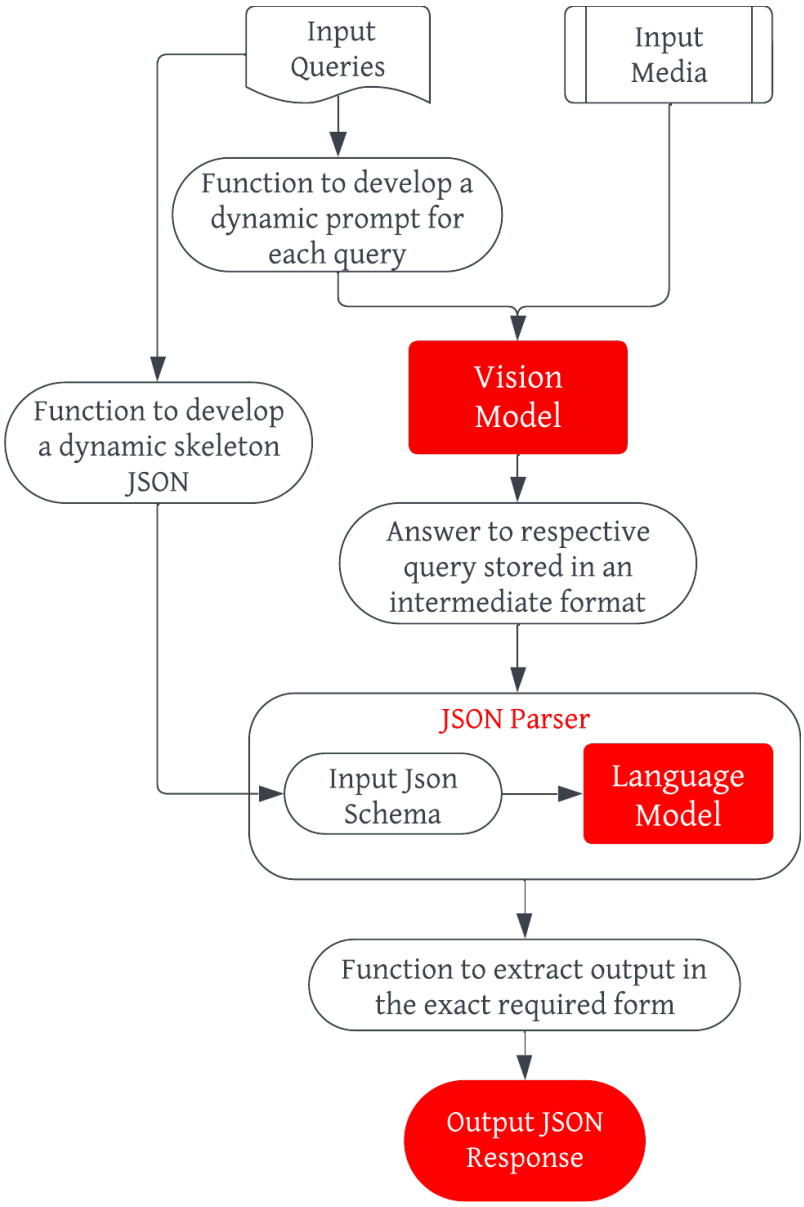
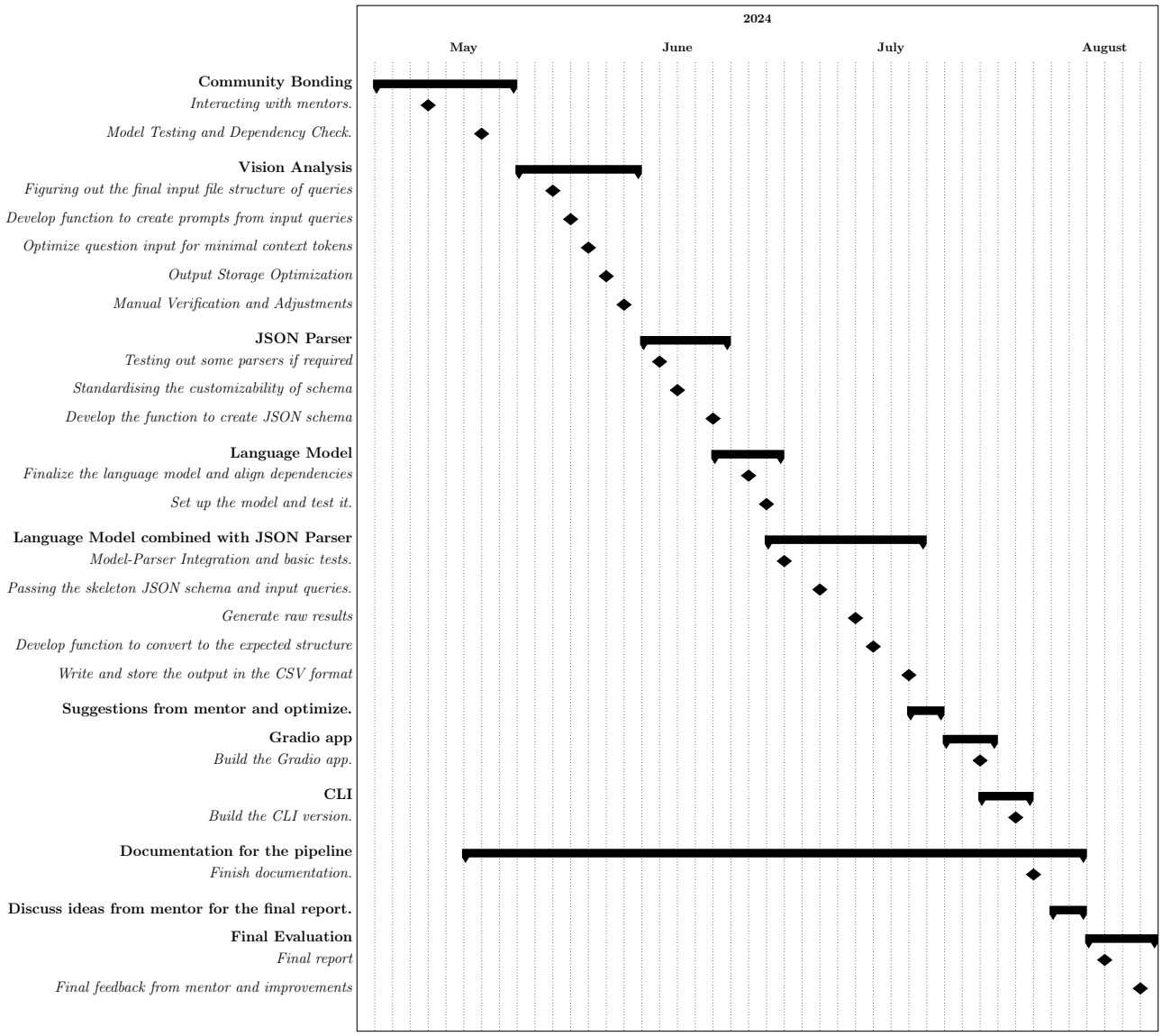


Figure 2: Workflow of Pipeline

Tentative Timeline

This section is my estimated timeline for the project. It aligns with the Medium (175h) Project size. After the community bonding period, the official coding period would begin on May 27. By the Midterm evaluation date in early July, I would have set up all the components and developed the majority of the functions. I aim to finish the remaining portion ie the final functions, Gradio app, and CLI version by early August leaving enough time before the Aug 26 deadline, to be able to make optimizations and document it, and then work on some of the future prospects.



Why me?

As a sophomore in computer science at VJTI, I've delved into computer vision with a focus on transformer-based models, which I find incredibly exciting. Starting with basic classifiers on CIFAR-10/MNIST datasets using PyTorch and TensorFlow, I gradually progressed to more complex projects like end-to-end video semantic segmentation from scratch. I built this remotely under the mentorship of my college senior, as a part of an open-source mentorship program for selective students, in the same structure as a GSOC project. My recent project involves developing a real-time chess position detector, analyzer, and move suggestions. I also participated in a recent hackathon where I built an AI-based legal document drafter and RAG apps for case studies and legal advice enhancing my NLP skills.

I've developed my skills in model building, fine-tuning, and dataset handling through this journey. Now, I'm keen on automating video and image annotations, a crucial task in computer vision. I've thoroughly researched the area and stay updated with the latest advancements. I have done immense testing including trying out various models, resolving dependency conflicts, and looking for the right parser, all with the

compute constraint of a Google Colab T4 GPU. With my experience in remote open-source projects and a commitment to regular updates, I believe I can tackle this project effectively.

I value high-level mentorship for project success, emphasizing clear communication and independence in my work, and won't nudge the mentors for help continuously. I understand the importance of clear and consistent communication, especially within a remote collaboration setting.

I understand the importance of documentation for anyone to use a tool and or even develop it further, hence I shall ensure that all my implementations are well documented. I would also write blogs about my updates, experiences, challenges, and how I solved them for community bonding.

My availability

- Once coding begins, I will work dedicatedly for at least 4-5 hours on weekdays(and 6-8 on weekends) until the completion of the project. After the submission of my proposal (on or before April 4), I will start researching and testing more models and look more into the code base of jsonformer.
- When the coding period begins (May 27), I have my end-semester examinations which will last until June 7. However, I promise to begin my work in the community bonding period as much as possible to be on track. After the examinations, I shall put in more time to make up for any lost time or lag.
- I have a 2-month long vacation after June 7 and no other commitments in hand, so I will be able to devote ample time to the project.

References

Video-LLaVA
Cog-VLM
jsonformer
Output Parser - Langchain
Output parser - OpenAI

VEDANT MEHRA

+91 8779564726 ◇ LinkedIN◇ vedantmehra20@gmail.com ◇ Github

EDUCATION

Veermata Jijabai Technological Institute, Mumbai 2022 - present
Bachelor of Technology in Computer Engineering (CGPI - 8.88)

Prakash College of Science 2020 - 2022
87.7%

St. Francis School (ICSE) 2010 - 2020
97.33%

Examinations
MHTCET 99.97% — JEE Mains 99.04% — JEE Advanced AIR 13796

RELEVANT COURSES

- Data Structures and Algorithms
- Linear Algebra
- Digital Logic Design
- Computer Organization and Architecture
- Python
- Basics of Deep Learning
- Convolutional Neural Networks
- Basics of Machine Learning

TECHNICAL SKILLS

Programming Languages	Python, C++, Javascript, HTML/CSS
Frameworks	React, PyTorch, Tensorflow
Tools	Git, VS Code, Google Colab
Libraries	NumPy, Matplotlib, OpenCV, Transformers

PROJECTS

Real Time Semantic Segmentation (Neural Networks, CV, PyTorch) Sep. 2023 - Nov. 2023

- Implemented Semantic Segmentation for images and videos using Deep Learning.
- Trained custom models with PyTorch and TensorFlow, focusing on UNet architecture, from scratch.
- Fine-tuned Semantic Segmentation models, achieving 85% accuracy on MobileNetV2 with a customized ADE20K dataset.

Interactive Music App (React, CV, Mediapipe) Dec.2023 - Jan.2024

- Developed a music streaming Web-App with an interactive UI and smart playback features using React.
- Implemented gaze-controlled music playback and gesture-controlled volume with OpenCV and MediaPipe.

End-to-End Chess Tracker (Python, OpenCV) Feb. 2024- Present

- Analyses a Chess Game on providing a video as input.
- Tracked the Chessboard contours to locate chess pieces and their positions.
- Tracking game progress, evaluating moves and providing a UI for the model.

POSITION OF RESPONSIBILITY

- **Department Head** at Pratibimb,VJTI Aug. 2023 - Present
  - Managing Cultural events and representing the entire department
- **Sector Executive** (Sponsorship) at Pratibimb,VJTI Jan. 2023 - Aug. 2023
  - Brand interactions and managing sponsors
- **Member** of ProjectX (Open source and DSA) at COC, VJTI Sep. 2023 - Present
  - Conducting workshops about open-source, Computer Vision, and DSA