Robust Visual Question-Answering using Generative Vision Language Models

Thesis submitted in partial fulfillment
of the requirements for the degree of

*Master of Science in **Computer Science and Engineering** by Research*

by

Rahul Mehta
2020900039
rahul.mehta@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
July 2024

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "**Robust Visual Question-Answering using Generative Vision Language Models**" by **Rahul Mehta**, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Adviser: Prof. Vasudeva Varma
Date	Co-Adviser: Prof. Manish Gupta

To *my family for everything*.

Acknowledgments

The past 3.5 years spent at IIIT Hyderabad was a such memorable experience for me. This thesis is a culmination of the invaluable research experience gained here. I will be forever grateful to the college and its community that helped me pursue a masters degree in research in Computer Science. IIIT Hyderabad provided me with access to a leading research community in India and to pursue my research while working as an industry professional.

First of all, I would like to thank Professor Vasudeva Varma who believed in me and gave me the opportunity to become part of the iREL lab. Under his guidance and mentorship, I got to work with some of the brightest research students and cutting edge research projects. Sir provided a collaborative research environment and guided me on my research on a regular basis. I will be forever indebted to sir for this opportunity.

I am also really thankful to my co-advisor Dr. Manish Gupta who guided me on a regular basis for my thesis. I learned a lot through his initial discussions to the final step till submitting my papers. Sir's mentorship with constructive feedback and encouragement have inspired me to become a better researcher and moreover a better person.

I am grateful to my wife who provided me with constant support throughout. She always motivated me through difficult times and it's mostly because of her support that I am able to complete this journey.

I would also like to thank my peers Aditya and Bhavyajeet with whom I got to work in research. It was great working with them in the past 1.5 years. I would like to thank the iREL peers Dhaval,Pavan,Savita maam and all the other lab members who provided a very collaborative environment in the lab.

I once again thank everyone for their guidance and support. It was an honor to be part of the IIIT family and I look forward to stay connected with the college and its community in the future.

Abstract

Visual Question Answering (VQA) represents a long standing challenge of combining computer vision and natural language processing, where machines has to answer questions about visual content such as images or videos. The challenge lies in not only recognizing objects, scenes, and relationships within the visual input but also comprehending the context of the questions posed in natural language. VQA systems are designed to understand the semantics of both the visual and textual modalities, requiring sophisticated algorithms to extract meaningful features from images or videos and integrate them with linguistic cues to generate accurate responses.

We release a visual question answering (VQA) system for electrical circuit images that could be useful as a quiz generator, design and verification assistant or an electrical diagnosis tool. Although there exists a vast literature on VQA, to the best of our knowledge, there is no existing work on VQA for electrical circuit images. To this end, we curate a new dataset, circuitVQA, of 115K+ questions on 5725 electrical images with $\sim$70 circuit symbols. The dataset contains schematic as well as hand-drawn images. The questions span various categories like counting, value, junction and position based questions. To be effective, models must demonstrate skills like object detection, text recognition, spatial understanding, question intent understanding and answer generation. We experiment with multiple foundational visio-linguistic models for this task and find that a fine-tuned BLIP model with component descriptions as additional input provides best results.

Hallucination in vision language models and their language model part is also a challenging area of research that directly affects a model's performance. We systematically study this phenomena and work on quantifying hallucination in a Vision Question Answering system. We also work on detecting hallucinations in large language models using ensemble of classifier models. Finally, we attempt to mitigate the hallucination problem by utilizing reinforcement learning based rewards to have better text generation capability for these language models.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The following thesis focuses on the problem of visual question answering. This chapter provides an overview of the VQA challenge by discussing the history of the challenge and formalizing the problem. It also lists many challenges associated with the VQA problem. Additionally, we discusses the rationale to pursue such a challenge. Towards the end of the chapter, we outline the key research contributions made in this thesis for each of the subsequent chapters.

1.1 The VQA Challenge

In 1950, Alan Turing designed the Turing test, which tests a machine's intelligence by having a text based dialog with it. Recently in 2015, [14] constructed a Visual Turing test where a machine's intelligence is measured by asking a set of questions on a given image and generating an answer. This led to the Visual Question Answering (VQA) challenge which aims at answering a text question in the context of an image [4]. The VQA task can be formalized as answering an image-question pair (I, q) given an image I and a natural language question q. [4]

Figure 1.1 provides a set of example pairs of images and their corresponding questions from the VQA dataset.

Application of such systems can have a very broad impact. It can be used to assist visually impaired persons in answering their questions. It can also be utilized in ecommerce, where it can be used to ask questions related to the product and to quickly recommend an image or a video given an input text query in search related settings. Additionally, it can also be utilized in embodied environments - like having a conversation with a robot or robots talking to each other.

For a machine learning model to be successful in the task of open-ended VQA, it needs a list of abilities to correctly handle the image-question input like fine-grained image recognition, object detection, spatial awareness, action recognition and knowledge-based reasoning. Fine-grained image recognition entails the ability to discern intricate details within an image, such as textures, patterns, and subtle visual cues. Object detection involves identifying and localizing specific objects within the image, regardless of their size, orientation, or occlusion. Spatial awareness is essential for understanding the spatial

relationships between objects within the scene, enabling the model to interpret questions that involve spatial reasoning. Action recognition allows the model to detect and understand dynamic elements within the image, such as movements or interactions between objects. Finally, knowledge-based reasoning involves leveraging external knowledge sources to infer answers to questions that require contextual understanding beyond what is directly observable in the image.

Given the progress in the zero-shot capabilities of extensive multimodal models, it's imperative to employ a meticulously crafted dataset to thoroughly evaluate the capabilities of these models.

Here we list few of the open research challenges while designing a VQA system

1.1.1 Understanding Context

One of the primary roadblock in VQA is to comprehend the contextual details within images. Images contain information in terms of objects, scenes, and relationships between these objects. Extracting this contextual understanding requires sophisticated image processing techniques that can capture both semantics (such as object categories and spatial relations), low level features (details of object categories) and also the complex relationship that exist between these objects.

1.1.2 Ambiguity and Subjectivity

Ambiguity arises in VQA tasks due to the inherent richness and diversity of visual content, coupled with the nuanced nuances of human language. The linguistic ambiguity adds another layer of complexity to VQA challenges. Natural language questions can be phrased in different ways, leading to variations in meaning and interpretation. For instance, the question "What color is the sky?" could refer to the color of the sky in the image, or it could be interpreted more abstractly, prompting answers like "blue" or "cloudy." Subjectivity can manifest in various aspects of VQA, including question formulation, answer selection, and evaluation criteria. For example, the perceived relevance of an answer to a given question may vary depending on individual preferences and perspectives. Similarly, the interpretation of visual concepts such as "beauty" or "happiness" may differ significantly among individuals, leading to subjective judgments in VQA tasks.Addressing ambiguity and subjectivity in VQA challenges requires a multi-faceted approach that incorporates both algorithmic techniques and human supervision. One strategy is to leverage contextual information from images and questions to disambiguate ambiguous queries. Contextual cues such as object relationships, scene semantics, and spatial arrangements can help narrow down the range of possible interpretations and guide the model towards more accurate answers.

1.1.3 Multimodality Fusion

VQA tasks involve integrating information from both visual and textual modalities. Effectively fusing these modalities while preserving the relevance and context of each input is non-trivial. Researchers have explored various techniques, including attention mechanisms, graph-based models, and neural fusion

architectures to fuse the information from both modalities. Newer transformer based architectures are usally encoder-decoder based and comprise mostly of an image encoder and a text decoder to generate an answer.

1.1.4 Complex Reasoning

Complex reasoning in VQA encompasses a wide range of cognitive processes, including deductive reasoning, spatial reasoning, temporal reasoning, relational reasoning, and commonsense reasoning. These forms of reasoning enable machines to go beyond mere recognition and generate meaningful answers based on understanding the content and context of images and questions.

Deductive reasoning involves drawing logical conclusions from given premises and is crucial for answering questions that require inference or logical deduction. For example, to answer a question like ”What is the color of the apple on the table?” the model needs to infer that the object in question is an apple based on its visual features and then deduce its color from the image.

Spatial reasoning is essential for understanding the spatial relationships between objects within an image and interpreting questions that involve spatial concepts such as position, orientation, and size. Questions like ”What is to the left of the chair?” or ”How many objects are above the table?” require the model to reason about spatial configurations within the image.

Temporal reasoning comes into play when dealing with dynamic scenes or sequences of events captured in images or videos. Models must understand the temporal order of events and infer causality and relationships between actions. For instance, answering a question like ”What happened before the cat jumped off the shelf?” requires temporal reasoning to identify the sequence of actions depicted in the image.

Relational reasoning involves understanding the semantic relationships between objects and entities within the scene and answering questions that require relational understanding. For example, to answer a question like ”What is the man holding?” the model needs to identify the person in the image, recognize the object in their hand, and understand the semantic relationship between them.

Commonsense reasoning is perhaps the most challenging aspect of complex reasoning in VQA, as it requires models to possess a broad understanding of the world and make inferences based on common sense and background knowledge. Questions like ”What would you use to eat soup?” or ”Why is the person wearing a coat?” necessitate commonsense reasoning to generate plausible answers.

Addressing complex reasoning in VQA tasks requires a combination of algorithmic techniques, architectural design choices, and dataset curation strategies. Architectures such as attention mechanisms, graph networks, and memory-augmented models have been proposed to enable models to capture and reason about complex relationships within images and questions.

1.1.5 Scalability,Efficiency and Privacy

Due to multi-modal nature of VQA, the systems based on that require processing both images and text, which becomes computationally challenging. Therefore, the current state of the art systems has few seconds of latency. Additional issue with deploying these systems is that they require storing images on cloud, leading to reduced privacy for the user. [7] is one such effort in making private and fast VQA system. Its a system designed to operate on user's mobile and that processes images within the device, and not to any cloud. The system also operates in millisecond latency compared to seconds ,leading to improved customer experience.

1.1.6 Dataset Bias

Current VQA datasets exhibit substantial bias, with studies indicating that models rely more on the formulation of the question rather than the content of the image. [54]. The wording of a question strongly influences the resulting answer, which poses significant challenges in evaluating VQA algorithms. Furthermore, questions that necessitate the interpretation of image content are relatively straightforward, often focusing on the presence of objects or scene characteristics, tasks that Convolutional Neural Networks (CNNs) can address effectively. Additionally, these datasets display notable language biases, with questions beginning with "Why" being less common and more challenging to answer. This disparity in question types could have significant implications for performance evaluation.

1.1.7 Societal Biases

Biases encoded in training data of VQA system can lead to societal biases and stereotypes , leading to unfair or discriminatory outcomes. [17] analyzed 5 famous VQA datasets. They found a stark difference between distribution of answers for questions about women and men. They also found samples with detrimental gender based stereotypical examples. Also, some races related attributes were found to be underrepresentated. Therefore, ensuring fairness, transparency, and accountability in VQA systems is paramount for deploying these systems in the society.

1.1.8 Open-Ended vs. Closed-Ended Questions

VQA tasks can be categorized into open-ended and closed-ended questions. Open-ended questions allow for a wide range of diverse answers, requiring models to generate free-form responses. Closed-ended questions, on the other hand, have a predefined set of answer choices, resembling a classification task. Developing a single model that excel at both types of questions still remains a big challenge.

1.1.9 Evaluation Metrics

Assessing the performance of VQA models necessitates appropriate evaluation metrics. Traditional metrics like accuracy may not capture the nuances of model performance, especially for open-ended questions where multiple valid answers exist. Designing evaluation metrics that align with human and accounts for societal values still remains an open area of research.

Figure 1.1: An example of VQA task.

1.2 Motivation

1.2.1 Need for domain specific VQA datasets

Several researchers have proposed multiple datasets for VQA. Most of these datasets focus more on real world objects. There are some scientific VQA datasets released like ScienceQA but they are limited to school based textbooks. Domain specific datasets provide an additional challenge to a machine learning system as they are usually trained on generic datasets. So there is a need to provide domain specific datasets like - medical,finance etc so that the ML systems can be thoroughly tested before being deployed in the corresponding industry.

Secondly, most of the images available on the web are digital and we have a dearth of hand-drawn based images. Hand-drawn images can possess additional challenges as images with similar content can look different because of different author styles. This can provide an additional challenge to a machine learning system. To the best of our knowledge, there is not much work done on handwritten based VQA systems.

1.2.2 Testing generative models on domain specific VQA datasets

Most methods for VQA use either basic multimodal fusion of language and image embeddings [27], attention-based multimodal fusion [67] or neural module networks [3, 19].

With the advent of transformers [64], various vision language models [32, 65, 30], have been proposed that have showcased remarkable capabilities for VQA datasets. These vision models contain an image encoder and a language model decoder to generate an answer. Usually, these encoders and decoders are pre-trained language models(PLMs) which are trained on general non domain specific datasets. Consequently, such models do not directly generalize to out-of-domain samples especially when there is large variety. Domain-specific variations can happen in terms of image characteristics, object categories, and language conventions to hinder a model's performance. Therefore, there is a need to test the capabilities of these systems in order to better understand their generalization capabilities.

Also, often these models have difficulty recognizing hand-drawn or hand-written text. In CLIP, the author showcased that even though the model has state of the art performance in many task related to retrieval, it cannot even beat a simple logistic regression models in hand-drawn MNIST dataset.

1.2.3 Limited study on hallucinations for VQA

Frequently, the vision language models generate nonsensical answers which are not reflective of the domain of the question. This phenomenon, known as hallucination, has been extensively studied. There are many metrics proposed to measure hallucination for vision language tasks like image captioning. However,the domain of Visual Question Answering (VQA) lacks a specific metric tailored to assess hallucination. Current,evaluation frameworks mostly focus on reporting the correctness of such systems like accuracy and neglect the hallucination part.

The utilization of large language models (LLMs) in VQA systems often contributes to the hallucination problem. This can be attributed to the usage of pre-trained Language models(PLMs) with the visual components. These PLM's are usually trained on generic datasets and when asked about domain specific questions, they can hallucinate and generate information from their pre-training data rather than providing information pertaining to a specific question.

1.3 Thesis Contribution

The primary contributions of this thesis is as follows -

1. We construct a novel and diverse circuit-based VQA dataset, CIRCUITVQA, for electrical domain, with 115K+ questions.The dataset contains carefully designed questions across 5 types which can be used test multiple visio-linguistic skills of multi-modal models.

2. We conduct a holistic evaluation of state-of-the-art vision language models. We perform (a) fine-tuning based evaluation of BLIP [37], GIT [69] and Pix2Struct [35] on train part of CIRCUITVQA, (b) zero-shot evaluation of instruction-tuned models like LLaVA [41], InstructBLIP [11] and OFA

3. We conduct extensive experimentation by combining external modules like Optical Character Recognition (OCR), Object detection and supplying detailed description of electrical components to improve and understand the capabilities of VQA task

4. We also work upon Hallucinations detection, measurement and mitigation for generative models

    - **For Hallucination Measurement**, we propose a novel quantifiable hallucination score called HVQA that can be widely applied to any Visual Question Answering(VQA) system
    - **For Hallucination Detection**, We devise a state of the art hallucination detection method for text generation models that can detect hallucinations across text generation task like - definition modelling, paraphrase generation and machine translation
    - **For Hallucination Mitigation**, we design a Reinforcement learning based rewards to improve the generation quality of text generation models like T5 for the specific task of fact2text generation

1.4 Thesis Workflow

The thesis is structured into seven chapters, and a brief overview of each chapter is provided as follows

- **Chapter 1** Introduces the VQA problem and open research challenges in tackling the problems. We also list down the motivation to create new domain specific VQA datasets and to test the generalisation power of various Vision Language models(VLMs) on these datasets. Finally, we describe the need to tackle the hallucination problem in these VLMs and LLMs.

- **Chapter 2** Describes the relevant work related to the problem of Visual Question Answering. It discusses the similar datasets and progress of Machine Learning in the field of electrical engineering. We also discuss the current state of the hallucination problem in language and vision-language models.

- **Chapter 3** Describes the process of domain specific dataset creation for electrical images which we call - CicrcuitVQA. The

- **Chapter 4** Studies the performance of various generative Vision Language models on CircuitVQA. We also describe in detail adding additional external knowledge to these systems.

- **Chapter 5** Studies the measurement,detection and mitigation of generative models for VLM and LLMs on various tasks like - VQA,Fact2Text Generation, Machine Translation,Paraphrase Generation and Definition Modelling

- **Chapter 6** Concludes with a summary of the work done in this thesis. It also discusses potential future direction for Visual Question Answering based on the insights and ideas drawn from the current thesis

  In the end, **Appendix A** provides a fine-grained details of the circuit components of the circuitVQA dataset. It list the units of measurements used in these components and also provide the textual descriptions generated by ChatGPT which was utilized in our experiments.

Chapter 2

Related work

This chapter commences by examining the research conducted on existing datasets utilized in visual question answering. Additionally, it delves into the advancements made in machine learning within the realm of electrical circuits. Lastly, a concise overview is provided regarding the definition of hallucination and the current methods used to measure hallucinations in both text-only language models and vision-language models.

2.1 VQA for Science

Unlike general VQA [4, 39, 51, 15, 23, 28] which focuses on natural images, VQA for Science is a subfield of VQA that focuses on answering questions about scientific images, such as diagrams, graphs, charts, and illustrations. ScienceQA [38] is a dataset containing 21k multimodal multiple-choice questions with a diverse set of science topics and annotations of their answers with corresponding lectures and expla- nations. AI2D [26] is a dataset of diagrams with constituent and relationship annotations for 5K+ diagrams and 15K+ questions and answers. ChartQA [41] covers 9.6K human- written questions as well as 23.1K questions generated from human-written chart sum- maries. FigureQA[25] contains synthetic and scientific-style figures from five classes: line plots, dot-line plots, vertical and horizontal bar graphs, and pie charts. DVQA [24] is a synthetic question-answering dataset on images of bar-charts. PlotQA [43] contains millions of QA pairs grounded over 224K+ plots with large variability in data labels, real-valued data, and complex reasoning questions. LeafQA [8]contains densely anno- tated figures/charts, constructed from real-world open data sources and is significantly more complex than FigureQA and DVQA. BizGraphQA [6] is a dataset for VQA over graph-structured diagrams from business domains. Although these datasets contain diverse range of diagrams, they do not particularly contain any questions related to electrical circuits.

2.2 ML for electrical circuits

With the down-scaling of CMOS technology, the design complexity of very large-scale integration (VLSI) is increasing. The increasing complexity of electronic design automation (EDA) tasks has aroused large interest in incorporating ML to solve EDA tasks [20] and electronic circuit design. Roy et al. [53] propose a method for recogni- tion of hand-drawn electrical and electronic circuit components, with both analog and digital components included. Recently, CNNs and YOLOv5 [49] have been been used to detect the components [47]. Some effort has been made to share such a dataset of the images containing electrical circuits [58]. ML has also been applied for fault diagnosis of analog circuits [21]. In this work, we extend this line of work by introducing the task of VQA for circuit images.

2.3 Hand-drawn or Text based Visual Question Answering Datasets

Historically, one of the first hand-drawn images datasets shared can be traced back to the famous handwritten MNIST database comprising of 60,0000 images of handwritten digits in black and white color. Later datasets include a dataset of human sketches comprising of images of everyday objects [13] . The focus of both the datasets was limited to classification of digits and sketches respectively.

Specific to visual question answering with handdrawn images, one such dataset is DocVQA [42] [45]. DocVQA contains a specific category of handdrawn images. This datasets contained images from different indsutries from the n UCSF Industry Documents Library.

Notable Text based datasets include OCR-VQA and TextVQA [56]. These datasets have questions which ask about the textutal content of the image. The OCR-VQA dataset contains images specific to books containing front and back pages with textual content of book title,author name and publisher names. On the other hand, TextVQA dataset contains more diversed set of real-world images like images of an airplane with text on its logo, text mentioned on measuring cup and so-on.

2.4 Hallucinations in Language models

One of the most widely adopted definition of hallucination in large language models (LLMs) is the amount of the generated content that is nonsensical or unfaithful to the provided source content

2.4.1 Measuring Hallucinations in Vision Language Models (VLMs)

For VLMs, this definition can be extended to contradictions between the visual input (taken as ‘fact’) and the textual output of a VLM [52, 33]. Ji et al. [22] provide a great summary of metrics to detect hallucination and methods to avoid the same. Current research on evaluation of hallucinations in VLMs are focused on two types: non-hallucinatory generation and hallucinatory discrimination. In the first category, a quantitative analysis of the hallucinatory elements in the model’s response and their proportion

is performed. Notable metrics include CHAIR [52] and model based methods that utilize LLMs like GPT4 as an evaluator [36] . CHAIR evaluates object hallucinations in image captioning by quantifying differences of objects between model generation and ground-truth captions. Hallucinatory discrimination requires a binary answer of whether the response contain any hallucinatory element. In this line of work, POPE [38] formulates a binary - yes or no questions about the object presence in the images such as "Is there a person in the image?". CIEM [18] is another method similar to POPE, that automates object selection by prompting ChatGPT. Since no specific hallucination evaluation metrics have been proposed for the VQA task specifically, we fill that gap by proposing a new metric,HVQA, in this paper.

2.4.2 Hallucination Detection in Large Language Models(LLMs)

Broadly, detecting hallucination can be categorized in the following ways

2.4.2.1 Factuality Hallucination Detection

1. **Retrieve External Facts** The model-generated content can be compare with reliable knowledge sources to identify inaccuracies. Recent approaches prioritize real-world evidence from uncurated web sources and automate the process with claim decomposition, document retrieval, summarization, and veracity classification. Other techniques include retrieving evidence from authoritative sources, fine-grained factual metrics, and enhancing evidence retrieval for hallucination detection through query expansion.

2. **Uncertainty Estimation** Hallucinations can also be linked to the uncertainty of the LLMs' generated content. By estimating this uncertainty, hallucinations can be detected. Two main approaches to uncertainty estimation can be a) Based on internal states of the model and b) those relying solely on observable behaviors.

2.4.2.2 Model aware Detection

These methods require access to model weights and their logits [62]. For machine translation task, [16] showcased that sequence log-probability performs quite well compared to reference based methods. For article generation task, [63] uncertainty estimation techniques [5, 60] were used to detect hallucination in ChatGPT. Other methods to detect hallucinations include Retrieval Augmented Generation [55] and Chain of Verification based techniques [31].

2.4.2.3 Black box Detection

With the prevalence of closed source models, there has been recent work on black-box hallucination detection methods which doesn't require the model inputs, only the generated text. For example, a

recently proposed system SelfCheckGPT [40] utilizes a sampling-based technique based on the idea that sampled responses for hallucinated sentences will contradict each other.

2.4.3 Hallucination Mitigation using Reinforcement Learning in Large Language Models(LLMs)

Deep reinforcement learning can be utilized to assign task-specific reward functions which motivate the model to generate outputs which are syntactically aligned to ground truth output. [29] Introduced Reinforcement learning based rewards to improve the text generation quality of BART. They used Style Classification Reward and BLEU score reward to target style and content and further improve the BART model.

One of the recently popular method is Reinforcement learning from Human Feedback(RLHF).In this method, we collect data where humans rank different model-generated responses based on their qualitative responses. Once these ranked responses are collected, the model is then fine-tuned to better align with human preferences and expectations. One of the drawback of this approach is if the human feedback is of poor quality, the model degrades in its performance.

Another proposed method is Reinforcement Learning from Knowledge Feedback (RLKF) [34]. This method utilizes human annotated ranked knowledge-based Q & A data and then train a reward model using Proximal Policy Optimisation (PPO) algorithm.

Our focus is on utilizing task-specific reward function instead of collecting feedback from human preferences which is time consuming and a costly exercise.

Chapter 3

CircuitVQA - Dataset Creation and Analysis

3.1 Dataset Creation

In this section, we introduce the novel CIRCUITVQA dataset for VQA on electrical circuit images. Particularly, we discuss three aspects of dataset construction: (a) collecting circuit images from various sources, (b) generating questions automatically, and (c) generating answers utilising either human annotations or automatically using available metadata.

3.1.1 Collection of circuit Images

We gather the images in our CIRCUITVQA dataset from five datasets available on public platforms like Roboflow2 and Kaggle3.The original source of many of these datasets can be traced back to the Handwritten Circuit Diagram Images (CGHD) [59]. These images are of two types: schematic and handwritten. Besides the images, the dataset contains metadata like human annotated bounding boxes and the corresponding symbol classes like resistor, ammeter etc. Table 3.1 shows details of the five source datasets: Roboflow Circuit recognition (D1)[1], Kaggle CGHD (D2)[2], Roboflow CGHD-Supplement (D3)[3], Roboflow Circuit Recognition Electronics (D4)[4] and Roboflow CGHD-Full Supplement (D5)[5]. Note that D1 and D4 are schematic datasets while others are hand-drawn. The datasets also differ in terms of the kind of electrical components. Some datasets like D1 have just 7 object classes while others like D2 have as many as 59 object classes labeled. We aggregate the data across these five datasets leading to a collection of 7027 images. Next, we identify potential duplicate images using perceptual hashing [73]. We then keep only one copy of these images by deleting similar ones with a Hamming distance greater than 3. This leads to our final unified dataset of 5725 images of which 3175 are hand-drawn and 2550 are schematic. We make the dataset publicly available.

[1] `https://universe.roboflow.com/rp-project/circuit-recognition`

[2] `https://www.kaggle.com/datasets/johannesbayer/cghd1152`

[3] `https://universe.roboflow.com/development-tohnm/cghd-supplement-g34fl`

[4] `https://universe.roboflow.com/rp-project/circuit-recognition-electronics/`

[5] `https://universe.roboflow.com/development-tohnm/cghd-full-supplemented`

	Type	# Images	Description	Frequent Object Classes
D1	Schematic	1284	Electrical circuits with 7141 annotations for object detection across 7 classes.	resistor, current-source, inductor, capacitor, voltage-ac, voltage-dc, arrow
D2	Hand-drawn	2304	Hand-drawn electrical circuit diagram images as well as 212K bounding box annotations across 59 object classes, and segmentation ground-truth files. Also has junction, cross-over and text annotations.	resistor, terminal input, diode, transistor, GND, LED, voltage, thyristor, switch, inductor, VSS, speaker, AND, NOT, varistor
D3	Hand-drawn	487	Electrical circuits with 8353 annotations for object detection across 14 classes.	junction, text, resistor, current-source, inductor, capacitor-unpolarized, voltage-dc, voltage-dc_ac, multi-cell-battery, gnd, diode, terminal, single-cell-battery, crossover
D4	Schematic	1273	Digital circuit images with 2398 annotations for object detection across 7 classes.	and, nand, not, or, xor, nor, xnor
D5	Hand-drawn	1679	Electrical circuits with 58K annotations for object detection across 45 classes.	junction, text, resistor, terminal, diode, capacitor-unpolarized, crossover, transistor, gnd, inductor, voltage-dc, thyristor, switch

Table 3.1: Details of source datasets for CIRCUITVQA

3.1.2 Generation of Question Answer Pairs

We generate five categories of questions: Simple Counting, Spatial counting, Position based, Value Based and Junction based. Table 3.2 shows example question-answer pairs for each question type for a sample circuit image. To generate these questions, we utilize the metadata associated with the images like the associated components and their bounding boxes. For each type, we obtain question templates using ChatGPT [46] and then instantiate questions using these templates. A full list of generated question templates is mentioned in Table 3.2. In the following we discuss the question-answer generation process for each question type. Table 3.3 summarizes the answer type for every type of question.

Fig. 3.1 illustrates our question-answer pairs generation pipeline.

1. Simple Counting Questions Given an image, in a simple counting question, we ask for the count of each component type in the image. We prompt ChatGPT with this prompt: "Paraphrase the following text in 20 ways - How many X does the circuit have?" This leads to 20 different paraphrases which are used as question templates to generate simple counting questions in CIRCUITVQA. For every image, we randomly sample a question template and replace the placeholder X with the actual component name to get an instantiated question. This can be done because each image has the component names and their

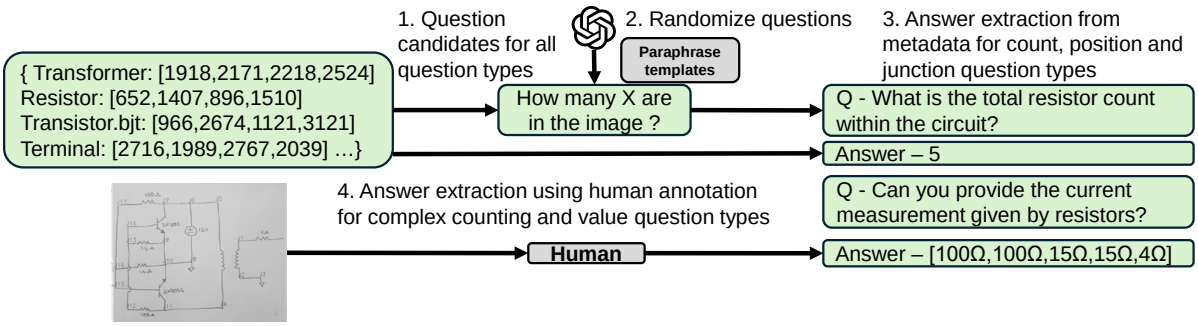

Figure 3.1: CIRCUITVQA: Question-answer pairs generation pipeline

counts as associated metadata. The metadata is also used to obtain the actual answer. Answering such questions requires a model to possess object recognition and counting skills.

2. Spatial Counting Questions Given an image, in a spatial counting question, we ask how many components of a certain type are connected directly to the left, right, top or bottom of the given component. Thus, for datasets D1, D2, D3 and D5, we use this question template "How many Y are connected directly to the ⟨direction⟩ of X?" where direction can be any of left, right, top or bottom. For dataset D4 which is based on digital gates, we use the following question templates: "How many gates are providing an input to X?", "How many gates are connected to the right of X?", "How many Y gates are connected to the right of X?", and "How many Y gates are connected to the left of X?" For every image in these datasets, we randomly sample a question template and replace the placeholders X and Y (from the set of components mentioned in metadata) with the actual component name to get an instantiated question. Since there is no automated way of generating an answer using associated metadata, we perform human annotation to annotate answers. The first author performed manual annotations for this objective and well-defined labeling task. Answering these questions requires the model to have an understanding of the way components are connected to each other spatially, i.e., object detection and localization skills.

3. Value Based Questions Given an image, in a value based question, we ask what is the value associated with a particular electrical component. We prompt ChatGPT with this prompt: "Paraphrase the following sentence in 20 ways. What is the reading on X?" This leads to 20 different paraphrases[6] which are used as question templates to generate value based questions in CIRCUITVQA. Again, we instantiate these templates to generate questions. If there are multiple components of type X in the image, the system is expected to provide a list of all of their values as the answer. Answering such questions requires a model to possess the optical character recognition skills, object recognition skills, and also the capability to link text labels with components.

Image metadata does not contain values associated with components. But the values are mentioned in the image. To generate answers automatically we used Google Vision APIs to perform OCR. The

[6]On manual inspection, we removed a few templates which did not make sense.

Question Type	Question Templates
Simple Counting	How many Xs are there in the specified circuit? What number of X are included in the given circuit? What is the total count of Xs in the circuit? Can you determine the number of Xs in the circuit? How numerous are the Xs in the circuit? What is the quantity of Xs present in the circuit? Are there multiple Xs in the circuit? What is the total X count within the circuit? Could you provide the number of Xs in the circuit? How many components are there in the circuit that function as Xs? What is the X tally in the circuit? Can you ascertain the number of Xs in the circuit? Could you indicate the quantity of Xs present in the circuit? How many X devices are there in the circuit? What is the total X count in the given circuit? Do you know how many Xs are present in the circuit? Can you determine the number of X components in the circuit? Could you specify the quantity of Xs in the circuit? Could you provide the count of Xs included in the circuit? What is the tally of components offering X in the circuit?
Spatial Counting	How many Y are connected directly to the left of X? How many Y are connected directly to the right of X? How many Y are connected directly to the top of X? How many Y are connected directly to the bottom of X? How many gates are providing an input to X? How many gates are connected to the right of X? How many Y gates are connected to the right of X? How many Y gates are connected to the left of X?
Value Based	What are the current reading displayed by the XX? Please provide the values displayed on the XX. What does the XX show in terms of reading? What numerical value is being shown on the XX? What reading does the XX display? What are the value depicted on the XX? Can you provide the current measurement given by the XX? What are the current value indicated on the XX? What does the XX read at the moment? What are the present reading on the XX? Could you share the current reading that the XX shows?
Junction based	Does a X exist between junction Y and junction Z? Is there a X present from junction Y to junction Z? Does a X occupy the space between junction Y and junction Z? Is there a X connecting junction Y to junction Z? Can a X be found between junction Y and junction Z? Does junction Y have a X leading to junction Z? Is there a X in the path from junction Y to junction Z? Can we observe a X between junction Y and junction Z? Does the circuit between junction Y and junction Z contain a X? Is a X situated between junction Y and junction Z? Is there impedance in the connection between junction Y and junction Z? Can you confirm the presence of a X between junction Y and junction Z? Is there any resistance between junction Y and junction Z? Does the circuit at junction Y involve a X leading to junction Z? Is a X located along the path from junction Y to junction Z? Can you verify if there is a X between junction Y and junction Z? Is a X part of the circuit between junction Y and junction Z? Is there a X linking junction Y to junction Z? Is there a X bridging the gap between junction Y and junction Z? Does junction Y connect to junction Z through a X? Is there any resistance encountered from junction Y to junction Z? Is a X placed in the line connecting junction Y and junction Z?
Position based	Which circuit symbol is on the far X? Identify the circuit symbol that is at the extreme X. What is the circuit symbol located on the Xmost side? Tell me the circuit symbol positioned at the Xmost end. Point out the circuit symbol that is furthest to the X. Which circuit symbol is on the very X-hand side? Please indicate the circuit symbol situated all the way to the X. What is the name of the circuit symbol at the Xmost position? Which circuit symbol is on the extreme X? Find the circuit symbol that is farthest to the X. Determine the circuit symbol on the Xmost side. Locate the circuit symbol positioned at the very X. Can you tell me which circuit symbol is at the Xmost position? Which circuit symbol is placed at the extreme X end? Point me to the circuit symbol on the Xmost side. What is the circuit symbol's name that appears on the Xmost? Show me the circuit symbol that is on the Xmost edge. Tell me the circuit symbol positioned to the far X. Among the circuit symbols which one is at the Xmost position?

Table 3.2: Question Templates for various question templates

Question Type	Answer Type
Simple Counting	Count (number)
Spatial Counting	Count (number)
Junction based	Binary
Position based	Component Name
Value based	List of values with units

Table 3.3: Answer types for every question type

Question Type	Training	Test	Val	Total
Simple Counting	16249	4776	2332	23357
Spatial Counting	624	170	236	1030
Junction based	45948	13998	6640	66586
Position based	14904	4232	2151	21287
Value based	2823	137	362	3322
Total	80548	23313	11721	115582

Table 3.4: # question-answer pairs per question type

value text label is then linked with the closest bounding box (from associated metadata), and hence to a relevant component. However, on manual inspection, we found that this led to poor results because (i) OCR quality is bad especially for hand-drawn images, and (ii) closest bounding box heuristic often fails. Hence, finally we resorted to manual answer labeling done by the first author.

4. Junction based Questions Given an image, in a junction based question, we would like to know whether a component exists between two junctions. Thus, these are binary questions. Datasets D2, D3 and D5 also have labeled bounding boxes for junctions. We prompt ChatGPT with this prompt: "Paraphrase the following text in 20 ways - Does a X exist between junction Y and junction Z?" The generated paraphrases are used as question templates to generate junction based questions. To instantiate these templates for a positive answer (i.e., answer="yes"), we need valid triples of component X, junction Y and junction Z. First, we randomly choose a junction Y. Next, based on its Euclidean distance with other junctions (computed using centers of their bounding boxes), we choose a junction Z which is closest to Y. Lastly for every component in the image, we find its distance to every junction, and choose a component X such that its sum of distances to junctions Y and Z is minimum compared to any other pair of junctions. Such a $\langle$X, Y, Z$\rangle$ triple helps generate a question with answer="yes". Next, we randomly sample a component X' from the image metadata, of a different type from X. Such a $\langle$X', Y, Z$\rangle$ triple helps generate a question with answer="no". Answering junction-based questions requires a model to possess object detection and localization, as well as spatial reasoning skills.

5. Position based Questions Given an image, in a position based question, we want to know the component at the left-most, right-most, top-most or bottom-most of the image. We prompt ChatGPT with this prompt: "Paraphrase the following in 20 ways - Which is the Xmost circuit symbol?" The resultant paraphrases are used as question templates to generate position based questions. For every image, we randomly sample a question template and replace the placeholder X with one of left, right, top or bottom to get an instantiated question. To get the answer, we utilize the bounding boxes of the components present in the image and find their minimum and maximum X and Y coordinates to decide the left-most, right-most, top-most or bottom-most components in the image. If there is no unique answer, we eliminate those questions. Answering such questions requires the model to possess object detection and localization skills.

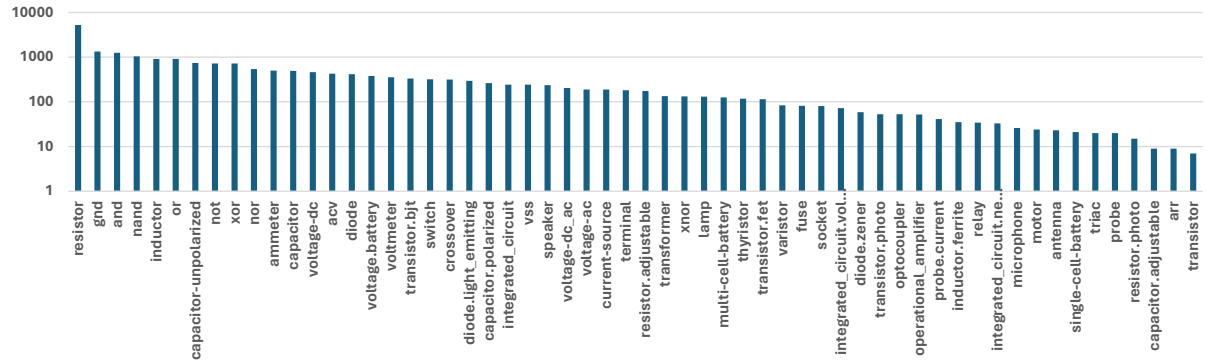

Figure 3.2: Frequency distribution of value-based questions across component names.

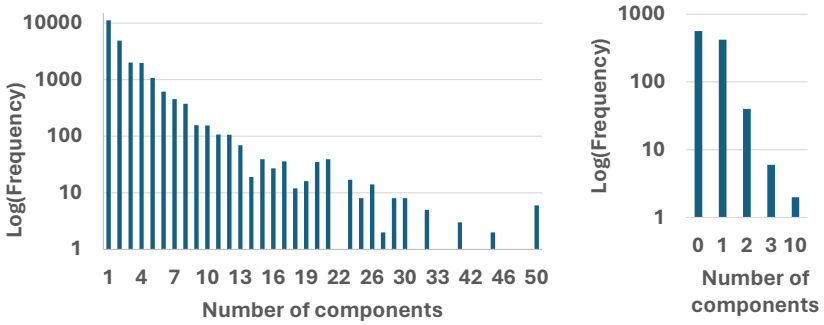

Figure 3.3: Frequency distribution of count-based questions across number of components of a particular type in images in CIRCUITVQA. Left: Simple Counting, Right: Spatial Counting.

3.2 CIRCUITVQA Dataset Analysis

We split the images into 70%, 20% and 10% split for training, testing and validation sets. Table 3.4 provides the count of questions by question type for train, test and validation splits. Fig. 3.2 shows the frequency distribution of value-based questions across component names in CIRCUITVQA. Components like "resistor", "gnd", "and gate", "nand gate", and "inductor" are the most frequent in value-based questions. Fig. 3.3 shows the frequency distribution of count-based questions across number of components of a particular type in images in CIRCUITVQA. The left plot is for simple counting questions while the right plot is for spatial counting. For simple counting questions, although several questions have count as 1, $\sim$52% questions have the answer count greater than 1. similarly, there is good variety in answers for spatial counting questions.

3.3 Summary and Conclusion

In this chapter, we explain the complete dataset creation process for CircuitVQA dataset. First, we share how we collect a large dataset of both hand drawn and schematic images. Subsequently, we detail out the automated Question generation and semi-automatic answer generation process,leveraging the metadata of the images collected. We provide a comprehensive breakdown of the step-by-step process involved in generating questions and answers across five distinct categories: Counting, Spatial Counting, Junction, Position, and Value-based Questions. Additionally, we present detailed analysis of the circuitVQA dataset detailing the distribution of components by questions and frequency distribution of value based questions.

In conclusion, we introduce the CircuitVQA dataset, which can serve as a valuable resource for the research community seeking to assess the generalization and domain-specific performance of state-of-the-art Vision-Language models. This chapter sets the groundwork for testing these models across various scenarios. The subsequent chapter utilizes the CircuitVQA dataset for performing the VQA task.

Chapter 4

CircuitVQA - Methods,Experiments and Results

4.1 Methods for CIRCUITVQA

To solve the CIRCUITVQA problem, we leverage two kinds of multimodal large language models as discussed in the following and detailed in Table 4.1.

4.1.1 Generative Models

BLIP [32] BLIP (Bootstrapping language-image pre-training) is a multimodal mixture of encoder-decoder which operates with unimodal encoders for image and text. The model comprises of an image-grounded text encoder, image-grounded text decoder based on BERT [11] and image encoder based on vision transformers (ViT) [12]. In Visual Question Answering setting, we follow the same methodology described in the paper for finetuning on train part of CIRCUITVQA. Specifically, we provide a circuit image-question pair to the image and text encoders separately, then compute the multimodal embeddings and provide it to the final text decoder (along with shifted outputs). The VQA model is fine tuned with Language modelling loss which utilizes ground-truth answer as the target labels.

GIT [65] GIT (Generative Image-to-text Transformer) is a decoder-only transformer that leverages CLIP [48] as a vision backbone. We fine-tune GIT on our task. Specifically, we concatenate the question and ground-truth answer as a special caption and apply the language modelling loss to the answer and [EOS] token.

Pix2Struct [30] Pix2Struct is a generative model for visual understanding that converts image to text. It has an image encoder and a text decoder. We provide the images and the questions to the input image encoder. The model renders the questions on top of the image. It scales images up or down to extract maximal patches that fit within the sequence length parameter. For fair comparison with other models, we evaluate its performance using input images that have been resized to 384×384 dimensions.

Model	Architecture Initialization			Pretraining Objective	Size (Parameters)
	Text Encoder	Image Encoder	Text Decoder		
BLIP-Base	BERT-base	ViT-B/16	BERT-base	Image captioning, image-text contrastive (ITC), image-text matching (ITM)	129M
GIT-Base	No text encoder	ViT-B/16	BERT	Image captioning	129M
Pix2Struct-Base	No text encoder	ViT	BERT	Screenshot parsing	282M
LLaVA	No text encoder	ViT-L/14	LLaMA	Auto-regressive loss for Conversation, detailed description, complex reasoning	6.76B
InstructBLIP	No text encoder	ViT + QFormer	Vicuna-7B	Language modeling on 26 datasets	7.91B

Table 4.1: Details of generative and instruction-tuned models that we experiment with for the CIRCUITVQA task.

4.1.2 Instruction Tuned Models

LLaVA [37] LLaVA (Large Language and Vision Assistant) is an end-to-end trained large multimodal model trained to follow human intent to complete visual tasks. It connects a vision encoder (ViT) with massive LLM based on LLaMA [61] or Vicuna [9]. At finetune time, the visual encoder weights are frozen but both the pre-trained weights of the projection layer and LLM are updated.

InstructBLIP [10] InstructBLIP is the instruction fine-tuned version of BLIP2. Just like BLIP2, it is pretrained in two stages. Instruction-aware Q-former module takes in the instruction text tokens as additional input. While performing instruction tuning, the image encoder and the LLM are frozen. Tuning is done using 26 publicly available diverse datasets.

4.1.3 Language Modelling Loss

We used the following loss functions for our experiments.

1. **Cross entropy loss** The loss function is the language modelling loss computed via Cross Entropy loss for the text decoder.

   For each image-text pair, for an image I, where $y_i, i \in \{1, \ldots, N\}$ be the text tokens, y_0 be the [BOS] token and $y_N + 1$ be the [EOS] token, p is the predicted probability and CE represents the cross entropy loss with label smoothing of 0.1, we define LM loss as

$$\mathcal{L} = \sum_{i=1}^{N} CE(y_i, p(y_i|I, \{y_j, j = 0, \ldots, i-1\}))\frac{1}{N} \tag{4.1}$$

2. **Weighted cross entropy Loss** Because of the imbalanced nature of our dataset, we provide class weights for each of the classes. Here each class is represented as a group of tokens in the loss function. Specifically, we calculate weight a class at a token level as a inverse count of that token in the dataset.

$$\mathcal{L} = \sum_{i=1}^{N} w_c * CE(y_i, p(y_i|I, \{y_j, j = 0, \ldots, i-1\}))\frac{1}{N} \tag{4.2}$$

4.1.4 Input Representations

In the base variant of our experiments, we pass the original image and text as input to various models discussed in the previous subsection. Further, we also experiment with passing other forms of input representations as input. These include OCR text, bounding box information from object detection, and visual description of components. Table 4.2 shows how such information is included as part of the input prompt to instruction-tuned models.

OCR text: Since some questions relate to actual text labels in the image, the models may benefit from outputs of an external OCR module. Therefore, we conduct an experiment to provide the OCR extracted tokens as an input to the vision-language models. We utilize Google Vision API[1] to collect the OCR outputs from the circuit image. Then, we append the OCR output as a prefix to the question separated by an [OCR] token for fine-tuning the generative models. We also experiment with passing filtered OCR text as input by keeping only the numbers and units typically expected by electrical measurements. In this setting, we retain any OCR output tokens that contain any of the symbols in [‘Ω’, ‘H’, ‘A’, ‘F’, ‘V’, ‘W’, ‘k’, ‘K’, ‘.’, ‘κ’, ‘M’] or a combination of these symbols with a digit or only digits.

Bounding box information: Bounding boxes identified using object detection methods help in attending to the relevant local parts of the image [35], and their usage has been shown to improve the performance in transformers [2]. Therefore to increase the spatial awareness of the components in images, we utilize an object detection module.

Metadata in CIRCUITVQA contains human annotated bounding boxes for various components in electrical circuit images. We use this dataset to (a) fine-tune the YOLOv8 [50] object detection model, and (b) use them in our fine tuning experiments of vision-language models.

We fine-tune the pretrained YOLOv8 model for 300 epochs on image size of 384. The batch size was kept at 16 and patience (early stopping criterion) was set at 50 epochs. The learning rate was determined automatically and set at 0.01 and SGD optimizer was used with momentum 0.9. On validation set, YOLOv8 finetuned Objection detection leads to precision of 78.1, recall of 63.9, mAP50

[1] https://cloud.google.com/vision

	Variant	Prompt
LLaVA	Base	Given the image, answer the following question: Q
	Desc	The question is about the circuit component ⟨Component-Name⟩. Its definition is as follows: ⟨ChatGPT-description⟩. Now, given the image, answer the following question: Q
	OCR	Here is the OCR information ⟨OCR⟩. You can use it to answer the following question. Now, given the image, answer the following question: Q
	BBox	Here are the bounding box coordinates of each component in the given image in the format of a pair of component name and coordinates. ⟨Bounding-box-coordinates⟩. Now, given the image, answer the following question: Q
	BBox +Segments	Here are the bounding box coordinates and segment of each component in the given image in the format of a triple of component name, coordinates, and segment name. ⟨Bounding-box-segments⟩. Now, given the image, answer the following question: Q
Instruct BLIP	Base	Q
	Desc	The question is about the circuit component ⟨Component-Name⟩. Its definition is as follows: ⟨ChatGPT-description⟩. Q
	OCR	Here is the OCR information ⟨OCR⟩. Q
	BBox	Here are the bounding box coordinates of each component in the given image in the format of a pair of component name and coordinates. ⟨Bounding-box-coordinates⟩. Q
	BBox +Segments	Here are the bounding box coordinates and segment of each component in the given image in the format of a triple of component name, coordinates, and segment name. ⟨Bounding-box-segments⟩. Q
GPT4V	Base	Q
	Desc	Use the following description of the electrical component to answer the question: ⟨ChatGPT-description⟩. Now, respond to this question: Q
	OCR	Use the following OCR output to answer the question: ⟨OCR⟩. Now, respond to this question: Q
	BBox	Use the following bounding box output comprising of the components and their coordinates in the image: ⟨Bounding-box-coordinates⟩. Now, respond to this question: Q
	BBox +Segments	Use the following bounding box output comprising of the components and their corresponding positions in the image : ⟨Bounding-box-segments⟩. Now, respond to this question: Q

Table 4.2: Input Prompt Templates for Instruction-based Models

of 69.8 and mAP(50-95) of 51.3. Fig. 4.1 shows a few object detection examples. The figure shows that our fine-tuned model is able to identify electrical components from circuit images effectively. We fine-tuned YOLOv8 for these classes: __background__, acv, ammeter, and, antenna, arr, block, capacitor, capacitor-unpolarized, capacitor.adjustable, capacitor-polarized, crossover, crystal, current-source, diac, diode, diode.light_emitting, diode.thyrector, fuse, gnd, diode.zener, inductor, inductor.coupled, inductor.ferrite, inductor2, integrated_circuit, integrated_circuit.ne555, integrated_circuit.voltage_regulator, junction, lamp, magnetic, mechanical, microphone, motor, multi-cell-battery, nand, nor, not, operational_amplifier, operational_amplifier.schmitt_trigger, optical, optocoupler, or, probe, probe.current,

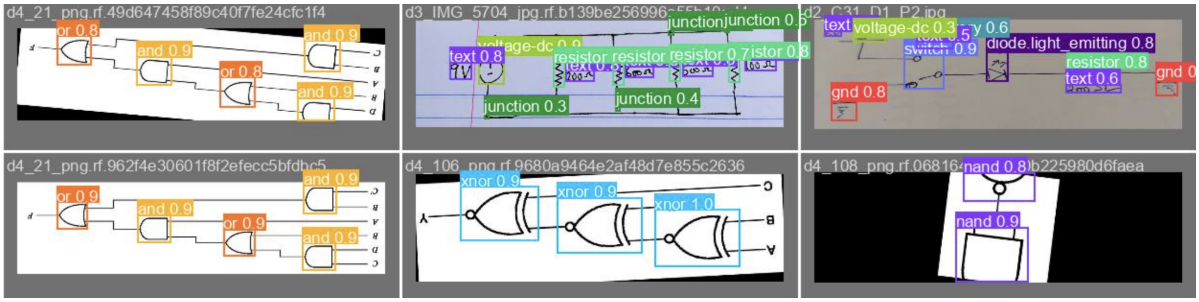

Figure 4.1: Examples of Object detection results using our finetuned YOLOv8.

relay, resistor, probe.voltage, resistor.adjustable, resistor.photo, single-cell-battery, socket, speaker, switch, terminal, text, thyristor, transformer, transistor, transistor-photo, transistor.bjt, transistor.fet, triac, unknown, varistor, voltage-ac, voltage-dc, voltage-dc_ac, voltage.battery, voltmeter, vss, xnor, and xor.

We experiment with two ways of providing the bounding box information as input to our vision-language models. In the BBox method, for each detected component, along with the component name, we pass bounding boxes in the $\langle x, y, w, h \rangle$ format where x, y are box center, w and h indicate width and height. The model may not be able to process the numerical information; hence we abstract out this information by assigning each bounding box to one of the 9 segments depending on its position in the image "upper left", "upper middle", "upper right", "left", "middle", "right", "lower left", "lower middle", "lower right". Based on this segment assignment, in the BBox+Segment method, for each detected component, along with the component name, we pass bounding boxes in the $\langle x, y, w, h \rangle$ format as well as the segment name.

Visual description of components: For every electrical component in our CIRCUITVQA dataset, we first obtain a short description using ChatGPT [46] with the following prompt "Describe the electrical component ⟨component⟩ in 50 words". In the Desc method, we pass the component description of relevant circuit component as a prefix to the question with a special token [DESC] separator. For example, description for capacitor is "Capacitor: Symbolized by two parallel lines with a gap, it stores and releases electrical energy, acting as a temporary energy reservoir in a circuit."

4.2 Experiments and Results

4.2.1 Experimental Setup

Generative models: For GIT, the learning rates are set to 1e-5 and 2e-5 for the image encoder and the text decoder respectively. Rest of the hyper-parameters are set to default values. For BLIP and Pix2Struct, learning rate for the text decoder is set to 2e-5. For all models, we use cosine learning rate scheduler. We use AdamW optimizer with a weight decay of 0.05. For Pix2Struct, we use default patch size of 16×16 and sequence length of 4096. For the text decoder of all models, we used hidden size of 768.

All models are trained for 10 epochs. The batch size is set to 4 for all the experiments. For fine-tuning and inference, we used a machine with 8 NVIDIA 32GB V100s. The computation time was 20-40 hours for various models. All models are trained to optimize for cross-entropy loss (with label smoothing of 0.1) except for Pix2Struct where we found weighted cross-entropy loss to perform better. Also, we perform all experiments using an input image size of 384×384.

Instruction-based models: To utilize InstructBLIP in zero shot settings, we set the number of beams to 5 and min length of the sequence to be generated to 1 and max length to 256. We keep the probability value for top p sampling to 0.9.Also, we set the temperature to 1 after trying out few different temperature settings. For LLaVA, we set the number of beams for beam search to 1. And provide the max length of the tokens to be generated to 512. Finally, we set the temperature to 0.

4.2.2 Metrics

For every model, we measure exact-match accuracy and hallucination score as the two metrics. A good model should not just generate accurate answers but also not hallucinate. Hallucinations for visual question answering deserve specific definitions. Hence, we discuss these definitions and propose a new metric HVQA in the following.

Hallucination in VQA systems could be in terms of predictions of non-existing in-domain objects, over-counting of existing objects, or predictions with out-of-domain objects. Accordingly, we define Hallucination Score for Visual Question Answering (HVQA) as average of three scores: (a) $\text{HVQA}_{\text{count}}$ (captures over-counting of existing objects), (b) $\text{HVQA}_{\text{in-domain}}$ (captures predictions of non-existing in-domain objects), and (c) $\text{HVQA}_{\text{out-domain}}$ (captures predictions with out-of-domain objects). Each of these are fractions with total number of predicted objects as the denominator. Since we perform object detection on the input image as part of generating the answer, we can directly use the object detection outputs to compute the above scores. $\text{HVQA}_{\text{count}}$ is applicable for simple counting, spatial counting and value based questions. $\text{HVQA}_{\text{in-domain}}$ and $\text{HVQA}_{\text{out-domain}}$ are both applicable for position-based questions. HVQA is a general metric applicable to any VQA task.

4.2.3 Results

Main Results: Table 4.3 shows our main results where we compare various methods under different input representations on the CIRCUITVQA test set with respect to Accuracy (Acc) and hallucination score (HVQA). BLIP provides the best accuracy while LLaVa and GPT4V provide the lowest hallucination scores. Our best model is a fine-tuned BLIP model with an accuracy of 91.7, when it is paired with prompts of visual description of the component (we call it as BLIP-Desc). It also maintains one of the lowest HVQA scores among all models. When an external OCR output is provided to these models, we observe a drop in their respective performances. This could be due to a lot of noise in the output of the OCR module. However, after postprocessing of the OCR output, there was a significant improvement in GIT (accuracy 71.2 vs 68.4) and InstructBLIP (accuracy 13.2 vs 12.5) when compared to the OCR output

Table 4.3: Main Results on CIRCUITVQA test set. H=HVQA. Acc (↑), HVQA (↓).

	Model	Base		OCR		OCR-Post		Desc		BBox		BBox+Seg	
		Acc	H	Acc	H	Acc	H	Acc	H	Acc	H	Acc	H
Fine-Tuned	BLIP	**84.4**	5.9	**81.8**	5.8	**80.8**	5.9	**91.7**	5.5	**75.6**	6.4	**74.0**	6.2
	GIT	72.5	6.3	68.4	6.2	71.2	5.9	55.3	6.7	40.2	7.6	48.7	6.1
	Pix2Struct	71.2	6.3	69.1	6.2	41.9	6.7	70.3	6.1	44.2	4.1	36.6	4.5
Zero-Shot	LLaVA	35.6	**3.8**	35.4	5.2	35.4	5.4	35.6	**3.8**	42.9	**2.8**	44.6	3.8
	InstructBLIP	6.8	19.2	12.5	14.3	13.2	13.7	35.0	5.5	6.8	19.2	6.8	19.2
	GPT4V	34.5	4.8	41.2	**2.2**	34.1	**4.0**	33.7	5.9	32.1	3.0	32.3	**3.7**

Table 4.4: Results per question type for the Desc variants of the models on CIRCUITVQA test set.

Model	Simple Counting	Spatial Counting	Junction based	Position based	Value based
BLIP	83.5	57.6	97.9	84.1	18.2
GIT	46.5	34.7	66.9	29.4	0.7
Pix2Struct	48.2	44.7	90.1	32.6	11.7
LLaVA	18.8	0.6	7.8	50.6	0.7
InstructBLIP	35.8	14.1	0	0.9	0
GPT4V	12.5	10.0	50.6	6.4	0.65

Table 4.5: Hallucination scores. A=count, B=in-domain, C=out-domain.

Model	A	B	C
BLIP	0.9	15.6	0
GIT	0.6	19.6	0
Pix2Struct	0.3	18.2	0
LLaVA	5.8	0	5.5
InstructBLIP	16.6	0	0
GPT4V	0.07	12.3	5.4

used directly. Also, when bounding boxes with their coordinates for each component were provided, we observe a drop in performance of the fine-tuned smaller models. However, the larger LLAVA zero-shot model can utilize that information and shows significant accuracy gains (42.9 vs 35.6 for the base model). Notably, the accuracy further increases to 44.6 with BBox+Seg.

Results per Question Type: For our best model (model that uses description), we analyze the results per question type in Table 4.4. Table 4.5 shows hallucination scores across various models on the CIRCUITVQA test set, where A=$\text{HVQA}_{\text{count}}$, B=$\text{HVQA}_{\text{in-domain}}$, C=$\text{HVQA}_{\text{out-domain}}$. We observe that BLIP-Desc outperforms all other models for each question type. It also hallucinates less on in-domain objects compared to its fine-tuned counterparts GIT and Pix2Struct. Also, fine tuning broadly ensures that the models (BLIP, GIT and Pix2Struct) do not hallucinate out-of-domain objects. On the other hand, instruction-tuned models like LLaVA and GPT4V have a significantly higher $\text{HVQA}_{\text{out-domain}}$. LLaVA predicts out-of-domain objects like 'circle', 'square', 'A', 'B', 'D', 'F', 'triangle', 'carlin', 'nano', 'peizo-keeper', 'trigger', 'Snake Snake Detector'. InstructBLIP is very cautious and has neither out-of-domain nor in-domain hallucinations, possibly because of its failure to understand position-based or value-based questions.

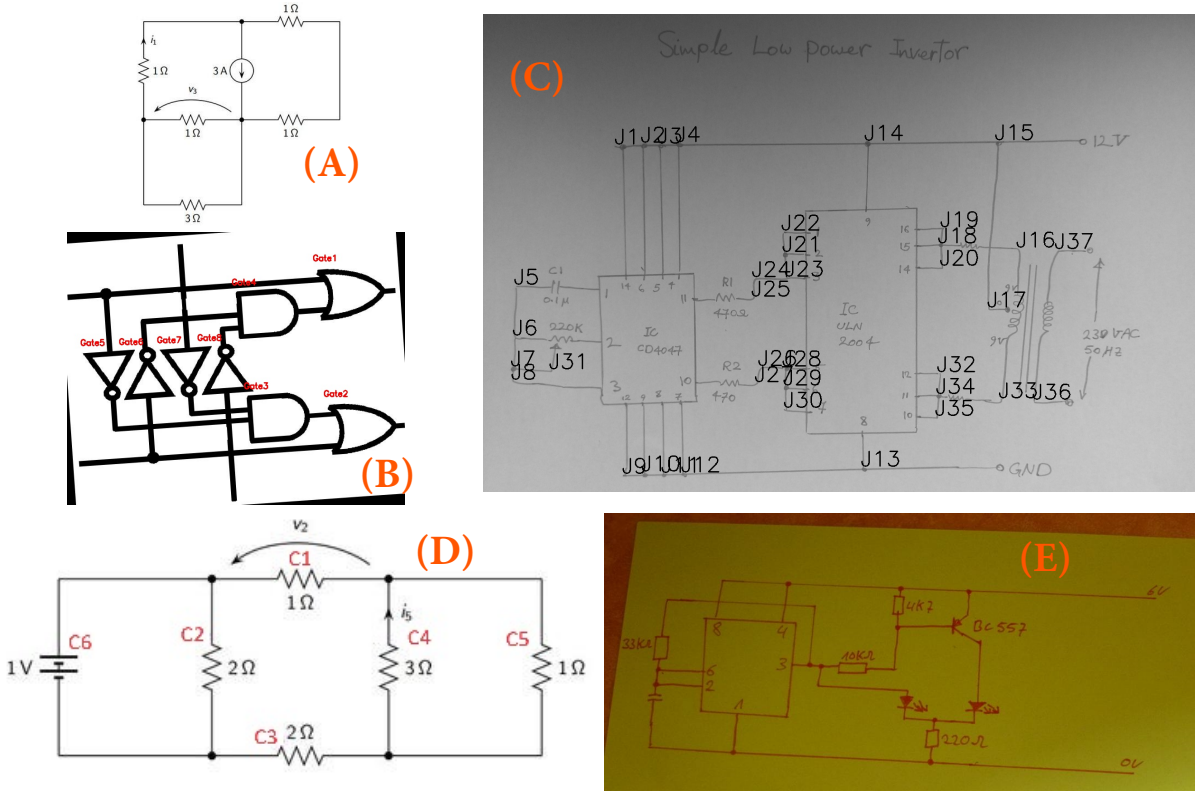

Figure 4.2: Examples of images for our best model.

For counting of objects, Pix2Struct hallucinates the least ($HVQA_{count}$ of 0.3), while our best model BLIP-Desc hallucinates a little more, but is twice accurate compared to Pix2Struct. Among all visual description based models, InstructBLIP hallucinates the most on counting ($HVQA_{count}$ of 16.6).

Case Studies: Table 4.6 show examples of questions and predicted answers associated with a few circuit images from the test set. For value based question, we can see that the model is able to accurately extract various values associated with the respective component. For junction question types, the model can correctly answer the respective question about two junctions even when there are more than 40 junctions in the image. We also observe that the model can correctly answer spatial counting questions by understanding the id associated with each component and then reasoning over the image to answer the question. Similarly the model can easily count values between 1 to 5, as shown in the examples.

Error Analysis: We manually analyzed 100 test cases where our system leads to an error, 20 for each question type. Among the 20 value-based questions, 4 errors can be attributed to incorrect units, 5 were a result of both units and values being wrong, and the majority (11 errors), were due to incorrect values. For 20 junction-based questions, 12 errors were for images with $\geq$40 junctions and 8 for images with $<$40 junctions. Broadly, we observe that accuracy drops with increase in number of junctions in input image. For position-based questions, for 9 samples, the predicted component was physically the second closest to the correct answer component; remaining 11 predictions were far from the actual answer. In simple

counting questions, we identified 11 over-counting errors, all within a range of 1 to 5, while there were 9 instances of under-counting. Spatial counting questions had 4 cases of over-counting and 16 examples of under-counting. Table 4.7 and Figure 4.2 shows a few error predictions and their corresponding images.

4.3 Summary and Conclusion

In this chapter, we introduce the challenge of visual question answering for electrical circuit images. We curated a dataset, CIRCUITVQA, for the task with five question types. We hope that this dataset will help the VQA community to focus on the critical problem of VQA for circuit images. We conducted thorough evaluation of various state-of-the-art vision language models. We also experimented with different forms of input representation including OCR text, bounding boxes based on object detection and detailed description of relevant circuit components. Our experiments reveal that the BLIP model with text description of components provide the highest VQA accuracy across most question types, and the lowest hallucination score.

Image	Question Type	Question	Prediction
(A)	Simple Counting	Could you indicate the quantity of resistors present in the circuit?	5
	Position-based	What is the circuit symbol's name that appears on the rightmost?	resistor
	Value-based	What is the value depicted on the ammeter?	['3a']
	Position-based	Which circuit symbol is on the extreme bottom?	resistor
	Value-based	What do the resistors read at the moment?	['1ohm', '1ohm', '1ohm', '1ohm', '3ohm']
(B)	Spatial Counting	How many gates are providing an input to Gate4	2
	Spatial Counting	How many or gates are connected to the right of Gate3 ?	1
	Simple Counting	Could you indicate the quantity of ors present in the circuit?	2
	Spatial Counting	How many and gates are connected to the left of Gate1 ?	1
	Position-based	Can you tell me which circuit symbol is at the leftmost position?	not
	Position-based	Which circuit symbol is placed at the extreme bottom end?	or
(C)	Junction-based	Is a integrated_circuit placed in the line connecting junction 4 and junction 12 ?	yes
	Simple Counting	How numerous are the capacitor-unpolarized in the circuit?	1
	Simple Counting	What number of transformer are included in the given circuit	1
	Simple Counting	How many components are there in the circuit that function as integrated_circuits?	2
	Junction-based	Is there a transformer interposed between junction 37 and junction 36 ?	yes

Table 4.6: Examples of Predictions from our best model.

Image	Question Type	Question	Answer	Prediction	Error Category
(C)	Value-based	What does the resistor.adjustable read at the moment?	['220kohm']	['100kohm']	Wrong values
	Junction-based	Is there a capacitor between junction 18 and junction 16?	no	yes	-
(D)	Spatial Counting	How many voltmeter are connected directly to the right of C4?	0	1	Over-counting
	Position-based	Which circuit symbol is placed at the extreme left end?	voltage.battery	resistor	Near miss
(E)	Simple Counting	Could you provide the count of resistors included in the circuit?	4	2	Under-counting

Table 4.7: Examples of error cases from our best model

Chapter 5

Hallucinations in Vision(Language) models - Measurement,Detection and Mitigation

5.1 Hallucination Measurement for VQA task for VLMs

Hallucination in VQA systems could be in terms of predictions of non-existing indomain objects, over-counting of existing objects, or predictions with out-of-domain objects. Accordingly, we define Hallucination Score for Visual Question Answering (HVQA) as average of three scores: (a) HVQAcount (captures over-counting of existing objects), (b) HVQAin-domain (captures predictions of non-existing in-domain objects), and (c) HVQAout-domain (captures predictions with out-of-domain objects). Each of these are fractions with total number of predicted objects as the denominator. Since we perform object detection on the input image as part of generating the answer, we can directly use the object detection outputs to compute the above scores. HVQA is a general metric applicable to any VQA task.

We describe calculating this metric using the following example - For a set of 4 questions for a given image containing 4 resistors and 5 capacitors only.

Question	Prediction	Actual	Hallucination Category
Q1. Which symbol is on the leftmost position ?	Voltage	Resistor	In-domain
Q2. Which symbol is on rightmost position ?	cat	resistor	Out-domain
Q3. Count the number of resistors in the image ?	13	4	Undercounting
Q4. Count the number of capacitors in the image ?	3	5	Overcounting

Table 5.1: Sample Example Q & A set

For table 5.1, we calculate HVQA as

$$HVQAin-domain = (Q1(1/1) + Q2(0/1) + Q3(0) + Q4(0))\frac{1}{4} = 1/4 \tag{5.1}$$

$$HVQAout - domain = (Q1(0/1) + Q2(1/1) + Q3(0) + Q4(0))\frac{1}{4} = 1/4 \tag{5.2}$$

$$HVQAcount = (Q1(0/1) + Q2(0/1) + Q3(9/13) + Q4(0/3))\frac{1}{4} = 9/52 \tag{5.3}$$

$$HVQA = ((0.25 + 0.25 + 9/52))\frac{1}{3} = 0.224 \tag{5.4}$$

5.2 Hallucination Detection in LLMs using ensemble models

5.2.1 Task

As large language models are often the answer generation component of vision-language models and the core reason for hallucination, our next step was to develop a hallucination detection system specifically for LLMs. To address this subproblem, we chooose to focus on the recently released shared task of Semeval 2024 Task 6, , **SHROOM : A Shared-task on Hallucinations and Related Observable Overgeneration Mistakes** [44]. The organizers of SHROOM propose a binary classification task wherein participants are tasked with predicting whether a machine-generated sentence constitutes a hallucination or not. The task encompasses three types of text generation tasks: Definition Modelling, Machine Translation, and Paraphrase Generation. Additionally, the shared task is divided into two tracks: model agnostic and model aware. In the followings sections, we describe the dataset details, the baseline model provided by the organizer and how we built a state-of-the-art classifier system using an ensemble of classifiers and compare the results with the baseline.

5.2.2 Dataset

Both the datasets consists of 1500 samples each for model aware and model agnostic track. Table 5.2 and Table 5.3 provides the dataset sample splits by each task for both the model aware and agnostic track. Each task contains examples from subtasks of definition modelling, paraphrase generation and machine translation. Figure 5.1 provide sample examples for each subtask of defnition modelling, paraphrase generation and machine translation.

5.2.3 Baseline system

The given baseline system is based on a simple prompt retrieval approach, derived from SelfCheck-GPT [40]. It uses an open-source Mistral instruction-finetuned model as its core component. The scores of the baselines system are being mentioned in Table 5.4

	Model Aware Track		
Task	Train	Dev	Test
Definition Modeling	10000	188	562
Machine Translation	10000	188	563
Paraphrase Generation	10000	125	375
Total	30000	501	1500

Table 5.2: Dataset Statistics for the Model Aware Track

	Model Agnostic Track		
Task	Train	Dev	Test
Definition Modeling	10000	187	562
Machine Translation	10000	187	563
Paraphrase Generation	10000	125	375
Total	30000	499	1500

Table 5.3: Dataset statistics for the Model Agnostic Track

5.2.4 Proposed approach

Our approach is centered around building a meta-model for hallucination detection, with the hypothesis that the quality of prediction from underlying base models is highly correlated with the meta-model's predictive power. Given a set of base models $M = \{m_1, m_2, ..., m_n\}$ and actual labels $L = \{l_1, l_2, ..., l_n\}$ in the dataset, the Spearman correlation between predicted hallucination scores H and actual labels is given by:

$$\rho_s(H, L) = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

where d_i is the difference between the ranks of corresponding elements in H and L.

Our overall process was to identify the meta-model that minimized this mean absolute error (MAE) function ϵ, where

$$\epsilon = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)$$

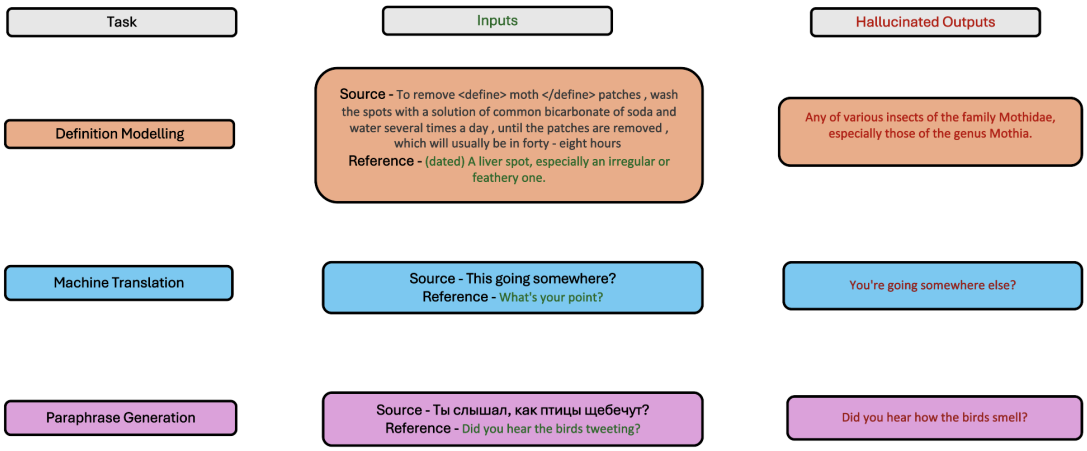

Figure 5.1: Sample examples of hallucinations for each subtask

because Spearman correlation was one of the secondary metrics for Task 6 evaluation. Here, Y_i represents the actual Spearman correlation values for hallucination and $\hat{Y}_i$ represents the predicted values. Our overall process is captured in Algorithm 1

Algorithm 1 Meta-Model Training/Evaluation

Input : Base models M, Meta-models N, Threshold x

Outputs : Top performing meta-model

for *each base model* m *in* M **do**
    $score_m \leftarrow$ Evaluate m (MAE)

$FilteredMs \leftarrow Models$.filter(MAE $< x$)

for *each meta-model* n *in* N **do**
    Train n with $FilteredMs$
    $metaScore_n \leftarrow$ Spearman MAE

$TopMeta \leftarrow$ Meta-model in N with lowest Spearman MAE

The algorithm follow a unified framework, initiating with the setup of training data and labels, with the ultimate aim of fine-tuning a meta-regressor model. A meta-search cross-validation approach was used to conduct a hyperparameter space for each model's architecture. The process involves iterating over the defined hyperparameter space for each algorithm, fitting the meta-regressor with the training data, and concluding with the identification and preparation of the highest-performing model for deployment. Also,MSE, MAE, and R-squared were used as additional proxies in meta-model evaluation.

Because this problem was assessed with binary classification accuracy, data was classified based on the Spearman correlation coefficient according to:

$$\text{Class} = \begin{cases} \text{'Hallucination'}, & \rho_s > 0.5 \\ \text{'Not Hallucination'}, & \text{otherwise} \end{cases}$$

to convert our regression problem into a binary classification task, simplifying the analysis and interpretation of results. Once converted to a classification problem, the primary metric used for evaluation was accuracy. Precision, Recall, F1 Score, and a confusion matrix were used for secondary evaluation.

5.2.5 Results

The results are shown in 5.4 . We achieve 1st place in model aware and 2nd place in model agnostic track and achieves an absolute improvement of 10 percent accuracy in model aware and 20 percent accuracy in model agnostic track.

Model Type	Track	Accuracy	Rho	Rank
Baseline	Model Aware	70.6	0.46	NA
Ours	Model Aware	80.6	0.71	1/46
Baseline	Model Agnostic	64.9	0.38	NA
Ours	Model Agnostic	84.7	0.77	2/49

Table 5.4: Final Modeling results on the test set

5.3 Hallucination Mitigation of LLMs using Reinforcement Learning

5.3.1 Task

To mitigate hallucinations in text generation, we worked on a task we proposed which we call Fact to long text generation [57]. This task involves taking as input, all facts about a particular entity and the output is a paragraph in another target language which is expected to capture all the semantic information in English facts without hallucination. We benchmark a reinforcement learning based reward method against a baseline T5 model and showcase the efficacy of the our approach.

5.3.2 Dataset

We derive our dataset, XLAlign, from the existing dataset, XAlignV2 (which is a revised version of XAlign [1]). In total, the XLAlign dataset contains 125,106 paragraphs across12 different languages. We split the dataset into train:validation:test in the ratio 75:15:10.

5.3.3 Proposed approach

We obtain enhanced output quality is through deep reinforcement learning, employing reward mechanisms tailored to specific tasks. These incentives drive the model to produce outputs that not only align syntactically with the desired output but also maintain semantic coherence with the input English data.

The concept of **Source Entailment Reward (R_{SE})** is introduced, which evaluates the semantic congruence between the generated text and the source English information. Given an input instance represented as $A(t_k)$ with reference text t_k, the RSE quantifies the semantic similarity between the generated text and the corresponding English facts $A(t_k)$. However, due to the inherent differences between English fact tokens and those in the generated target language, a method is devised to establish an equivalence termed entailment probability. This probability is based on the likelihood that the presence of ngrams in the generated text aligns with the associated English facts, thereby posing a significant challenge in language understanding. Let y_k represent the generated sentence text, and y_k^n denote the collection of all ngrams of order n within y_k. Let b denote one such ngram, and let w be any token within b. The entailment probability of token w being entailed by the source is computed as the maximum probability among its potential entailments by each lexical item (subject, relation, object, or qualifier) v within a fact in the source.

$$P(w \impliedby A(t_k)) = \max_{v \in A(t_K)} P(w \impliedby v) \tag{5.5}$$

where $P(w \impliedby v)$ is estimated by using similarity scores from MuRIL embeddings of the token w and lexical item v. Using this, we compute the entailment probability of ngram b being entailed as the geometric average of entailment probabilities of each of the constituent tokens as follows.

$$P(b \impliedby A(t_k)) = \left(\prod_{w \in b} P(w \impliedby A(t_k)) \right)^{1/|b|} \tag{5.6}$$

where $|b|$ is the order of the ngram b. Lastly, entailment score of generated sentence y_k for ngrams of order n with respect to the aligned ground truth facts is obtained by taking mean of entailment probabilities of each of the constituent ngrams as follows.

$$ES^n(y_k, A(t_k)) = \frac{\sum_{b \in y_k^n} (P(nbyA(t_k)))}{|y_k^n|} \tag{5.7}$$

where $|y_k^n|$ denotes the number of ngrams in y_k^n. Lastly, we obtain entailment score $ES(y_k, A(t_k))$ of generated sentence y_k with respect to the aligned ground truth facts by taking geometric mean of $ES^n(y_k, A(t_k))$ across all orders. Then the source entailment reward is given by $R_{SE} = \lambda_{SE} \times ES(y_k, A(t_k))$ where λ_{SE} is a tunable hyperparameter that controls the reward in the overall objective to be optimized.

Target Similarity Reward (RTS): This metric evaluates the resemblance in structure between the generated text y_k and the reference text t_k, quantified using the BLEU metric. Hence, $R_{TS} = \lambda_{TS} \times$

$BLEU(\mathrm{y}_k, t_k), where \lambda_{TS}$ represents a configurable hyper-parameter governing the significance of this reward within the broader optimization objective.

These rewards serve as guides for policy learning. We employ the policy gradient algorithm [66] to maximize the expected reward—either source entailment (R_{SE}) or target similarity (R_{SE})—for the generated sequence y_k. The gradient with respect to the parameters ϕ of the neural network model is estimated through sampling as follows:

$$\Delta_\phi J(\phi) = E[R.\Delta_\phi \log(P(y_k|x;\phi))] \tag{5.8}$$

where R denotes either the R_{SE} reward or the R_{TS} reward, y_k is sampled from the distribution of model outputs at each decoding time step, x (comprising $A(t_k)$, language ID l_i, and the coverage prompt) serves as the model input, and ϕ represents the parameters of the long text generation model. The overarching objectives for ϕ encompass both the loss of the base model L_{TG} and the policy gradient stemming from the various rewards.

5.3.4 Results

Table 5.5 shows the result of the XLFT task. We report the highest scores on BLUE,chrF++,XPARENT when the above mentioned Reinforcement learning rewards were used in training.

	All Test Instances			Test Instances with ≥2 sentences		
	BLEU	chrF++	XPARENT	BLEU	chrF++	XPARENT
Single-Sentence XFST [1]	15.515	45.410	42.202	14.059	44.171	40.301
Multi-Sentence XFST	18.660	37.621	50.338	15.873	37.067	50.327
Fact Organizer+Single-Sentence XFST	20.395	44.136	52.679	18.227	43.366	52.628
Fact Organizer+CP+RL	**22.663**	**49.532**	**55.328**	**18.760**	**48.717**	**54.966**

Table 5.5: Performance Comparison of various methods for XFLT task.

5.4 Summary and Conclusion

To summarize, this chapter addresses the issue of hallucination in vision and language based models. Initially, our focus is on quantifying hallucination in vision language models for the Visual Question Answering(VQA) task. We introduce a novel HVQA hallucination score that captures the hallucination in terms of overcounting and in-domain and out-domain specific object hallucinations. Subsequently, we examine the problem of hallucination detection in large language models.Here, we propose an ensemble classifier by utilizing multiple classifiers and their probabilities of hallucination. Finally, we improve the text generation capabilities of transformer models by integrating reinforcement learning based rewards in the language models. This led to improved text generation scores such as BLEU. In conclusion, this

chapter presents various strategies for measuring, detecting, and mitigating hallucination, with the aim of addressing this significant issue in vision and large language models. We anticipate that these strategies will contribute to ongoing efforts to combat hallucination effectively in vision and large language models.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this thesis, we introduced the novel problem of visual question answering(VQA) on electrical circuits. Initially, we introduced a domain specific VQA dataset of electrical circuits. This dataset can serve multiple purposes, first it can be used to advance the field of machine learning for electrical images. It can also be used to test the generalization capabilities of deep learning models. We extensively perform testing of numerous state of the art Vision Language models(VLMs) to measure the generalization ability on our CircuitVQA dataset. We also showcase the impact on VLMs when external modules like OCR,bounding boxes and textual descriptions are provided as an additional information to these VLMs.

In the second phase of the thesis, we specifically tackle the problem of hallucinations in Vision and Large Language models. We focus our efforts on quantification,detection and mitigation of hallucinations. We proposed a novel method to measure hallucination in a VQA system. For hallucination detection in LLMs, we designed a state of the art ensemble clasification system. Finally, for hallucination mitigation, we designed reinforcement learning based rewards like target similarity reward and source entailment reward that improved the text generation capabilities like BLEU score for fact2text generation task.

Chapter-1 Addresses the problem of visual question answering and the numerous challenges associated with it. It also highlights the motivation to pursue such problem providing issues like limited study of VLMs in out of domain settings and the pressing need for VQA datasets to tackle such problem.

Chapter-2 Provides an overview of related work done in VQA datasets and identifies the key gaps in the literature which our current work improves upon. It also shares the progress of machine learning in the field of electrical domain. Finally, in this chapter, an overview of the hallucination problem in Vision Language models and Large Language models is provided. It also highlights the current limitation in terms of quantification,detection and mitigation of the hallucinations in such models.

Chapter-3 The chapter delves deeper into the data creation process of Circuit-VQA dataset that consists of electrical images base visual question answering. We discuss the various types of question categories that are formulated - counting,spatial counting, position,junction and value based. Each of them tackle different visio-linguistic skills. Other information related to the prompt templates using

ChatGPT to generate diverse linguistic questions is also provided. We also looked at dataset statistics by different question types and also at an overall level.

Chapter-4 The chapter focusses on comprehensive experimentation on the earlier created CircuitVQA dataset using state of the art generative Vision Language models. Various state of the art generative models like BLIP,PIX2STRUCT and GIT and also Instruction-tuned models like LLAVA,GPT-4V were utilized. We further perform various ablation studies with external modules like OCR, visual text description, bounding boxes in coordinate format and segment formats. Finally, after all experiments, we concluded that the BLIP model with textual description provide us with maximum accuracy and lowest VQA based hallucination scores.

Chapter-5 Focuses on our efforts for hallucination measurement, detection and mitigation. To measure hallucination in VQA models, we propose a novel VQA score which we call Hallucination Score for Visual Question Answering (HVQA). It comprises HVQAcount,HVQAin-domain and HVQAout-domain. Next, we tackle the problem of hallucination detection in LLMs for various substasks of machine translation,definition modeling and paraphrase generation. There, we utilize an ensemble of classifiers built on top of the output predicted probabilities of multiple classifiers like ChatGPT,Vectra etc. Lastly, we work on the problem of mitigating hallucination. In this setting, we develop reinforcement learning based rewards - Source entailment reward(SER) and target similarity reward(TSR). The source entailment reward measures the semantic similarity between the generated text and source English facts. The target similarity reward evaluates the resemblance in structure between the generated text and the reference text. This metric evaluates the similarity in structure between the text which is generated and the available reference text. We showcase the usage of these two rewards that improves the text generation scores like BLEU.

6.2 Future Work

Although we have made significant strides towards releasing domain specific VQA dataset and testing SOTA architectures,these models have a long way to go. We think there are few other research directions that can be worked upon to improve the abilities of these models and to make them more robust against hallucinations and more efficient in training and inference.

- **Better Vision Language Model architectures** For building the model, we utilized various state of the art models. Based on the current limitations of the current architecture we conclude that newer architectures can help on Vision Question Answering task a.Dynamic resolution of images - We observed that for better performaces b. Pre Training vision-language models with handwritten images can help improve the model in their generalization capabilities

- **Hallucination mitigation in VQA systems** Expanding the exploration of hallucination mitigation in Vision Question Answering (VQA) systems presents a novel challenge. While existing research has made significant strides in quantifying hallucinations within VQA frameworks, there remains a

noticeable gap in actively addressing and mitigating these phenomena. Our study sheds light on the potential of reinforcement learning (RL) techniques to mitigate hallucinations, particularly in the context of large language models.

By leveraging RL-based reward mechanisms, we demonstrated promising avenues that improves the robustness and reliability of VQA systems against hallucinatory responses. These methods hold the potential to incentivize the generation of more accurate and grounded answers, thereby enhancing the overall quality of VQA outputs.

However, the application of RL techniques to alleviate hallucinations in VQA systems represents a new and challenging area. Future research endeavors could delve deeper into the design and implementation of RL-based frameworks tailored specifically to address hallucination phenomena within VQA architectures. This could involve exploring novel reward functions that effectively penalize hallucinatory responses and incentivize the generation of answers grounded in genuine visual and contextual cues.

Moreover, the scalability and generalizability of RL-based hallucination mitigation strategies required careful attention, especially in the context of deploying VQA systems in real-world scenarios with diverse and dynamic visual environments. Addressing these challenges may involve designing adaptive RL algorithms capable of continuously learning and that adapts to to evolving patterns and contexts in visual data.

- **Extending from image to Video Modality** The current scope of the thesis was on images, a natural extension can be question answering applied to video or also called VideoQA. We observe that it's quite GPU intensive to work with consumer GPUs for Visual Question Answering. The sheer volume of frames within a single video—potentially numbering in the thousands—underscores the necessity for developing highly efficient neural network architectures tailored to handle such intensive computational tasks.

  To address this challenge, we can explore innovative approaches to streamline video-based VQA algorithms, optimizing them for performance on both hardware and software fronts. This could involve devising novel network architectures that are specifically designed to extract relevant information from video streams in a computationally efficient manner.

  Additionally, techniques like as temporal attention mechanisms and recurrent neural networks (RNNs) can be leveraged to effectively capture the temporal dependencies while minimizing computational overhead.

  Moreover, advancements in hardware acceleration technologies, such as specialized video processing units (VPUs) or distributed computing frameworks, may offer promising avenues for improving the efficiency of video-based VQA systems. By harnessing the parallel processing capabilities of these hardware accelerators, researchers can potentially alleviate the computational burden associated with analyzing large volumes of video data in real-time.

By innovating in both algorithmic design and hardware optimization, we can unlock the full potential of video-based VQA systems across a diverse range of applications and domains..

- **Enriching VQA models with Knowledge Graphs** A relatively less explored area is of adding external knowledge to VQA systems. By leveraging domain-specific information, like scientific knowledge and principles and electrical engineering concepts, these systems can potentially achieve more nuanced and accurate responses to visual queries.This could involve techniques such as knowledge graph embeddings, semantic parsing, or attention mechanisms tailored to incorporate domain-specific knowledge sources. Moreover, considering the dynamic nature of scientific knowledge , continual updates of external knowledge sources pose additional challenges and new sets of opportunities. We seed strategies for maintaining the relevance and accuracy of external knowledge repositories in VQA systems over time.

Overall, this thesis has made a core contribution in releasing domain specific VQA dataset that can serve as a benchmark to test wide variety of visual understanding based skill of visio-language foundation models. Also, we have made contributions in the direction of measuring,detecting and mitigating hallucinations in LLMs and VLMs. Finally, we have listed out many open-ended research directions that can be pursued towards more robust and efficient multimodal system for designing visual question answering (VQA) systems.

Appendix A

CircuitVQA - Component Details

In this appendix, we describe some of the fine-grained and important details with respect to the circuitVQA dataset. First, the list of electrical units being used in the components are discussed. This was helpful for designing the OCR based experiments. We also describe the component description extracted from ChatGPT based on the prompt to describe that component. This was helpful to design the visual description experiment.

A.1 Units for Electrical Measurements

The units of electrical component are useful while designing the experiment of OCR with postprocessing of units.In these experiments, we pass filtered OCR text as input, we retain any OCR output tokens that contain any of the symbols in the unit list or a combination of these units with a digit or only digits.

Table A.1 contains the above mentioned the list of units.

Unit	Component
‘Ω’	Resistor
’H’	Inductor
‘A’	Ammeter
’F’	Capacitor
‘V’	Voltmeter

Table A.1: Common electrical component with their electrical units

A.2 Component Descriptions

Here we describe all descriptions generated by ChatGPT based on an input prompt ”Please describe X in 50 words” where we limit the description to 50 words. We use this description as a visual cue in textual format to assist the vision-language model in generating better answers. Also, we purposefully keep the description short to fit it in the context length of 512 in the transformers.

- AND: AND gates are digital logic gates with two or more inputs and one output. They produce a high output (1) only when all inputs are high (1), and a low output (0) otherwise. The symbol resembles an intersection of two input lines leading to one output line.
- OR: OR gates are digital logic gates with two or more inputs and one output. They produce a high output (1) if at least one input is high (1), and a low output (0) only when all inputs are low (0). The symbol resembles a curved figure with two or more input lines leading to one output line.
- NAND: The NAND gate is an AND gate followed by a NOT gate. It is symbolized by the AND gate symbol with a circle at its output. It produces a low output only when both inputs are high.
- NOR: The NOR gate is an OR gate followed by a NOT gate. It is depicted by a curved shape with a circle at its output. It produces a high output only when both inputs are low.
- NOT: The NOT gate, also called an inverter, is represented by a triangle with a small circle at its input. It produces the logical complement of its input, i.e., a high input becomes low and vice versa.
- XNOR: The XNOR gate is an XOR gate followed by a NOT gate. It is depicted by the XOR gate symbol with a circle at its output. It produces a high output when both inputs are either high or low.
- XOR: The XOR gate, or exclusive OR gate, is represented by a curved shape with a plus (+) sign at the intersection of two inputs. It produces a high output when the number of high inputs is odd, and a low output when the number of high inputs is even.
- TERMINAL: In an electrical circuit, a terminal refers to a point where an external component, such as a resistor or a power supply, is connected. It acts as an interface for the circuit, enabling the transfer of electrical signals or power between the circuit and the connected device.
- Capacitor: Symbolized by two parallel lines with a gap, it stores and releases electrical energy, acting as a temporary energy reservoir in a circuit.
- Crossover: Two lines crossing without touching, indicating the crossover connection of two or more electrical signals, typically used in audio systems to separate frequencies for speakers.
- Current-Source: Represented by a circle with an arrow inside, it provides a constant current in a circuit, ensuring a steady flow of electrical charges.

- Diode: Shown as a triangle with a horizontal line, it allows current flow in one direction while blocking it in the opposite direction, acting as a one-way valve for electricity.
- GND: Depicted by a horizontal line with three vertical lines branching downwards, it represents the ground or common reference point in a circuit, used as a voltage reference for other components.
- Inductor: Represented by a coil symbol, it stores energy in a magnetic field when current flows through it, resisting changes in current and acting as a component in filters and transformers.
- Integrated_Circuit: Symbolized by various shapes representing interconnected electronic components, it represents a miniaturized circuit that combines multiple functions onto a single chip, such as microprocessors or memory chips.
- Lamp: Shown as a circle with a cross inside, it represents a light-emitting component, typically an incandescent bulb or an LED, indicating the presence of a light source in a circuit.
- Multi-Cell-Battery: Symbolized by multiple stacked rectangles, it represents a battery composed of multiple cells connected in series or parallel, providing a higher voltage or increased capacity.
- Operational_Amplifier: Depicted by a triangular shape with two inputs and an output, it represents an electronic amplifier used for signal processing and amplification in various circuits, offering high gain and versatile functionality.
- Optocoupler: Shown as a line with an arrow intersecting a broken line, it consists of an LED and a photodetector coupled together, providing electrical isolation while transmitting signals using light, commonly used for noise reduction and electrical isolation in circuits.
- Resistor: Symbolized by a zigzag line, it represents a passive electronic component that limits the flow of electrical current, regulating voltage levels and offering resistance to the passage of electricity. It is commonly used for signal attenuation and current control in circuits.
- Single-Cell-Battery: Represented by a single rectangle, it symbolizes a battery consisting of a single cell, providing a specific voltage and capacity for powering electronic devices.
- Socket: Depicted as an opening with lines indicating contact points, it represents an electrical socket or connector where another component can be inserted or connected, allowing for the transfer of electrical signals or power.
- Speaker: Shown as a cone or a sound wave symbol, it represents a transducer that converts electrical signals into sound waves, producing audio output in devices such as radios, televisions, and audio systems.
- Switch: Symbolized by a simple line or a line with a gap, it represents a device that can interrupt or establish an electrical connection in a circuit, enabling control over the flow of current or signals.

- Terminal: Depicted as a point where two or more lines connect, it represents an interface or connection point in a circuit where external components or wires can be connected, facilitating the transfer of electrical signals or power.
- Thyristor: Symbolized by a triangle with an additional line or arrow, it represents a semiconductor device used for controlling large currents, typically in switching applications or as a solid-state relay.
- Transformer: Shown as two coils or windings with lines connecting them, it represents a device used to transfer electrical energy between circuits through electromagnetic induction, altering voltage and current levels.
- Transistor: Depicted by various symbols such as a triangle or rectangles, it represents a semiconductor device used for amplification, switching, and signal processing in electronic circuits, playing a crucial role in modern electronics and digital systems.
- Triac: Symbolized by a combination of two triangular shapes in opposite directions, it represents a semiconductor device capable of controlling AC power by regulating the flow of current in both directions, commonly used in dimmer switches and motor control applications.
- Varistor: Shown as a symbol resembling two back-to-back diodes, it represents a voltage-dependent resistor that protects electronic circuits from voltage surges or transients by rapidly changing its resistance to divert excessive voltage and protect sensitive components.
- Voltage-AC: Represented by a wavy line, it denotes an alternating current (AC) voltage source in a circuit, where the direction and magnitude of the voltage periodically change over time, commonly used in household electricity supply.
- Voltage-DC: Depicted by a straight line, it signifies a direct current (DC) voltage source in a circuit, where the voltage remains constant in magnitude and direction over time, commonly provided by batteries or power supplies.
- VSS: Shown as a horizontal line with three vertical lines branching downwards, it represents the negative power supply or ground connection in a circuit, serving as a reference point for voltage measurements and providing a common reference for other components.
- Capacitor-Unpolarized: Symbolized by two parallel lines of equal length, it represents an unpolarized capacitor that can be connected in any direction in a circuit. It stores and releases electrical energy, acting as a temporary energy reservoir without a specific polarity requirement.
- Junction: Depicted by a dot where three or more lines intersect, it represents a junction point in a circuit where multiple conductors meet. It indicates the connection of different wires or components without specifying any particular electrical behavior.

- Voltage-DC_AC: Shown as a combination of straight and wavy lines, it denotes a voltage source that can provide both direct current (DC) and alternating current (AC) output. It signifies a power source capable of delivering both constant and periodically changing voltage, often found in specialized electronic systems.
- ACV: Symbolized by a wavy line with a letter V above it, it represents an AC voltage source or measurement point in a circuit. It indicates the presence or measurement of alternating current voltage, commonly used in electrical systems powered by AC sources.
- ARR: Shown as a circle with an arrow that curves back, it represents an array or set of components connected together. It signifies a grouping or arrangement of multiple elements or devices in a circuit.
- Ammeter: Depicted as a circle with a letter A inside, it represents an ammeter used to measure electric current in a circuit. It indicates a device capable of measuring and displaying the magnitude of electrical current passing through a specific point.
- Voltmeter: Symbolized by a circle with a letter V inside, it represents a voltmeter used to measure voltage in a circuit. It signifies a device capable of measuring and displaying the magnitude of electrical potential difference between two points in a circuit.
- Voltage.Battery: Symbolized by a series of stacked rectangles with a longer line on top, it represents a voltage source such as a battery that provides a constant potential difference in a circuit, supplying electrical energy to other components.
- Resistor.Adjustable: Shown as a rectangle with an arrow inside, it represents an adjustable resistor or potentiometer. It allows the user to vary the resistance manually, controlling the flow of current and adjusting the voltage levels in a circuit.
- Resistor.Photo: Depicted as a rectangle with a circle and an arrow inside, it represents a photoresistor or light-dependent resistor (LDR). Its resistance changes with the intensity of light, allowing it to be used in light-sensing applications.
- Capacitor.Polarized: Symbolized by two parallel lines with a curved line at one end, it represents a polarized capacitor. It stores and releases electrical energy with a specific polarity requirement, with the curved line indicating the positive terminal.
- Capacitor.Adjustable: Shown as two parallel lines with an arrow inside, it represents an adjustable capacitor or variable capacitor. It allows for manual adjustment of capacitance, altering the ability to store and release electrical energy in a circuit.
- Inductor.Ferrite: Depicted as a coil symbol with a solid core, it represents an inductor with a ferrite core. It stores energy in a magnetic field using a ferrite material, providing inductance and impedance in electronic circuit.

- Inductor.Coupled: Symbolized by two or more coil symbols interconnected, it represents coupled inductors. They are used to transfer energy between different parts of a circuit, achieving mutual inductance and coupling effects.
- Diode.Light_Emitting: Shown as a triangle with two arrows pointing away, it represents a light-emitting diode (LED). It emits light when current passes through it, commonly used as indicators or light sources in electronic devices.
- Diode.Thyrector: Symbolized by a diode symbol with a vertical line extending from its cathode, it represents a thyrector or transient voltage suppression diode (TVS diode). It protects electronic circuits from voltage spikes and transients by diverting excessive voltage.
- Diode.Zener: Depicted as a diode symbol with a tilted Z inside, it represents a Zener diode. It allows current to flow in reverse-bias direction when the voltage exceeds its breakdown voltage (Zener voltage), commonly used as voltage regulators or in voltage reference circuits.
- DIAC: Shown as two parallel lines with a diagonal line connecting them, it represents a DIAC (diode alternating current) or bidirectional diode. It is a two-terminal device used in triggering and controlling alternating current (AC) circuits, often used in dimmer switches and triggering circuits.
- Transistor.BJT: Symbolized by a triangle and two intersecting lines, it represents a bipolar junction transistor (BJT). It amplifies or switches electronic signals by controlling the flow of current between its terminals, commonly used in amplifiers and digital logic circuits.
- Transistor.FET: Shown as a line with an arrow and a vertical line connected, it represents a field-effect transistor (FET). It controls the flow of current using an electric field, offering high input impedance and low power consumption, commonly used in amplifiers and switching applications.
- Transistor.Photo: Depicted as a triangle with a circle and an arrow inside, it represents a phototransistor. It is a light-sensitive transistor that responds to the intensity of incident light, commonly used in light-sensing and optoelectronic applications.
- Operational_Amplifier.Schmitt_Trigger: Symbolized by a triangle with a hysteresis symbol, it represents an operational amplifier configured as a Schmitt trigger. It converts a varying input voltage into a binary output signal with hysteresis, used for noise rejection and signal shaping.
- Integrated_Circuit.NE555: Shown as a rectangle with pins, it represents the NE555 integrated circuit (IC). It is a versatile timer IC widely used in timing applications, pulse generation, and oscillator circuits, providing precision timing functions.
- Integrated_Circuit.Voltage_Regulator: Symbolized by a rectangle with pins and a horizontal line above, it represents a voltage regulator IC. It maintains a stable output voltage regardless of input voltage variations, commonly used to provide regulated power supply in electronic circuits, ensuring consistent voltage levels.

- Probe: A pointed symbol used for measurement and testing purposes, representing a probe that allows for electrical or signal connections to be made in a circuit.
- Probe.Current: Symbolized by a circle with an arrow passing through it, it represents a current probe used to measure or monitor electrical current in a circuit without interrupting the flow.
- Probe.Voltage: Shown as a circle with a plus (+) and minus (-) sign inside, it represents a voltage probe used to measure or monitor electrical voltage in a circuit, providing voltage readings without significant impact on the circuit.
- Relay: Depicted as a rectangle with a zigzag line inside, it represents an electromagnetic relay that controls the flow of current in one circuit using a signal from another circuit, commonly used for switching higher power loads.
- Fuse: Symbolized by a squiggly line, it represents a fuse that protects circuits by breaking the circuit when current exceeds a specified limit, preventing damage to other components in the event of a fault.
- Motor: Shown as a circle with a curved line inside, it represents an electric motor that converts electrical energy into mechanical energy, generating rotational motion to drive machinery or devices.
- Microphone: Symbolized by a circle with a triangle or lines inside, it represents a microphone that converts sound waves into electrical signals, used for audio recording and amplification.
- Antenna: Depicted as a straight or curved line with branches, it represents an antenna that receives or transmits electromagnetic signals, such as radio waves, used for wireless communication and reception.
- Crystal: Shown as a shape with symmetrical lines, it represents a crystal oscillator, a component used for generating precise and stable oscillating signals in electronic circuits, commonly used in timing and clock circuits.
- Mechanical: Symbolized by gears or mechanical components, it represents a mechanical device or component in a circuit, typically used for physical movement or mechanical functions.
- Magnetic: Depicted as a horseshoe magnet or a symbol with letter N and letter S poles, it represents a magnetic component or magnetic field in a circuit, indicating the presence or utilization of magnetic forces or materials.
- Optical: Symbolized by a lens or light beam, it represents an optical component or element in a circuit, indicating the utilization or interaction of light or optical signals.
- Block: Shown as a rectangle or square, it represents a block or functional unit in a circuit diagram, representing a specific component, module, or subsystem with internal circuitry and functionality.

Related Publications

- **Rahul Mehta**,Manish Gupta,Vasudeva Varma and Bhavyajeet Singh. **CIRCUITVQA: A Visual Question Answering Dataset for Electrical Circuit Images**. *Under Review at European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD),2024*

- **Rahul Mehta**,Andrew Hoblitzell,Hyeju Jang,Jack O'Keefe and Vasudeva Varma. **GroupCheck-GPT at SemEval-2024 Task 6: Multi-task Hallucination Detection Using LLM Uncertainty And Meta-models.** *The 18th International Workshop on Semantic Evaluation,2024,NAACL Workshop. [**1st Place**]*

- Bhavyajeet Singh **Rahul Mehta**,Manish Gupta,Vasudeva Varma,Aditya Hari and Tushar Abhishek. **XFLT : Exploring Techniques for Generating Cross Lingual Factually Grounded Long Text.** *European Conference on Artificial Intelligence(ECAI),2023.*

- Bhavyajeet Singh, Aditya Hari, **Rahul Mehta**, Manish Gupta, Vasudeva Varma **Cross- lingual Multi-Sentence Fact-to-Text Generation: Generating factually grounded Wikipedia Articles using Wikidata** *In Proceedings of The Wiki Workshop 2023*

Other Publications

- **Rahul Mehta** ,Vasudeva Varma **LLM-RM at SemEval-2023 Task 2: Multilingual Complex NER using XLM-RoBERTa** *The 17th International Workshop on Semantic Evaluation,2023,ACL Workshop.*

Bibliography

[1] T. Abhishek, S. Sagare, B. Singh, A. Sharma, M. Gupta, and V. Varma. Xalign: Cross-lingual fact-to-text alignment and generation for low-resource languages. In *The World Wide Web Conference*, pages 171–175, 2022.

[2] C. Alberti, J. Ling, M. Collins, and D. Reitter. Fusion of detected objects in text for visual question answering. In *EMNLP-IJCNLP*, pages 2131–2140, 2019.

[3] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Neural module networks. In *CVPR*, pages 39–48, 2016.

[4] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. Vqa: Visual question answering. In *ICCV*, pages 2425–2433, 2015.

[5] A. Azaria and T. Mitchell. The internal state of an llm knows when its lying. *arXiv preprint arXiv:2304.13734*, 2023.

[6] P. Babkin, W. Watson, Z. Ma, L. Cecchi, N. Raman, A. Nourbakhsh, and S. Shah. Bizgraphqa: A dataset for image-based inference over graph-structured diagrams from business domains. In *SIGIR*, pages 2691–2700, 2023.

[7] Q. Cao, P. Khanna, N. D. Lane, and A. Balasubramanian. Mobivqa: Efficient on-device visual question answering. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 6(2), jul 2022.

[8] R. Chaudhry, S. Shekhar, U. Gupta, P. Maneriker, P. Bansal, and A. Joshi. Leaf-qa: Locate, encode & attend for figure question answering. In *WACV*, pages 3512–3521, 2020.

[9] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, I. Stoica, and E. P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023.

[10] W. Dai, J. Li, D. Li, A. M. H. Tiong, J. Zhao, W. Wang, B. Li, P. Fung, and S. Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning, 2023.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, volume 1, page 2, 2019.

[12] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020.

[13] M. Eitz, J. Hays, and M. Alexa. How do humans sketch objects? *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4):44:1–44:10, 2012.

[14] D. Geman, S. Geman, N. Hallonquist, and L. Younes. Visual turing test for computer vision systems. *Proc. of the National Academy of Sciences*, 112(12):3618–3623, 2015.

[15] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *CVPR*, pages 6904–6913, 2017.

[16] N. M. Guerreiro, D. Alves, J. Waldendorf, B. Haddow, A. Birch, P. Colombo, and A. F. T. Martins. Hallucinations in large multilingual translation models, 2023.

[17] Y. Hirota, Y. Nakashima, and N. Garcia. Gender and racial bias in visual question answering datasets. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 1280–1292, 2022.

[18] H. Hu, J. Zhang, M. Zhao, and Z. Sun. Ciem: Contrastive instruction evaluation method for better instruction tuning. In *Workshop on Instruction Tuning and Instruction Following*, 2023.

[19] R. Hu, J. Andreas, M. Rohrbach, T. Darrell, and K. Saenko. Learning to reason: End-to-end module networks for visual question answering. In *ICCV*, pages 804–813, 2017.

[20] G. Huang, J. Hu, Y. He, J. Liu, M. Ma, Z. Shen, J. Wu, Y. Xu, H. Zhang, K. Zhong, et al. Machine learning for electronic design automation: A survey. *Trans. on Design Automation of Electronic Systems (TODAES)*, 26(5):1–46, 2021.

[21] K. Huang, H.-G. Stratigopoulos, and S. Mir. Fault diagnosis of analog circuits based on machine learning. In *2010 Design, Automation Test in Europe Conference Exhibition (DATE 2010)*, pages 1761–1766, 2010.

[22] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

[23] J. Johnson, B. Hariharan, L. Van Der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, pages 2901–2910, 2017.

[24] K. Kafle, B. Price, S. Cohen, and C. Kanan. Dvqa: Understanding data visualizations via question answering. In *CVPR*, pages 5648–5656, 2018.

[25] S. E. Kahou, V. Michalski, A. Atkinson, Á. Kádár, A. Trischler, and Y. Bengio. Figureqa: An annotated figure dataset for visual reasoning. *arXiv:1710.07300*, 2017.

[26] A. Kembhavi, M. Salvato, E. Kolve, M. Seo, H. Hajishirzi, and A. Farhadi. A diagram is worth a dozen images. In *ECCV*, pages 235–251, 2016.

[27] A. Kembhavi, M. Seo, D. Schwenk, J. Choi, A. Farhadi, and H. Hajishirzi. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *CVPR*, pages 4999–5007, 2017.

[28] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. S. Bernstein, and L. Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vision*, 123(1):32–73, may 2017.

[29] H. Lai, A. Toral, and M. Nissim. Thank you BART! rewarding pre-trained models improves formality style transfer. In C. Zong, F. Xia, W. Li, and R. Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 484–494, Online, Aug. 2021. Association for Computational Linguistics.

[30] K. Lee, M. Joshi, I. R. Turc, H. Hu, F. Liu, J. M. Eisenschlos, U. Khandelwal, P. Shaw, M.-W. Chang, and K. Toutanova. Pix2struct: Screenshot parsing as pretraining for visual language understanding. In *ICML*, pages 18893–18912, 2023.

[31] D. Lei, Y. Li, M. Hu, M. Wang, V. Yun, E. Ching, and E. Kamal. Chain of natural language inference for reducing large language model ungrounded hallucinations. *arXiv*, cs.CL(arXiv:2310.03951), 2023.

[32] J. Li, D. Li, C. Xiong, and S. Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, pages 12888–12900. PMLR, 2022.

[33] Y. Li, Y. Du, K. Zhou, J. Wang, W. X. Zhao, and J.-R. Wen. Evaluating object hallucination in large vision-language models. In *EMNLP*, pages 292–305, 2023.

[34] Y. Liang, Z. Song, H. Wang, and J. Zhang. Learning to trust your feelings: Leveraging self-awareness in llms for hallucination mitigation. *arXiv preprint arXiv:2401.15449*, 2024.

[35] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755, 2014.

[36] F. Liu, K. Lin, L. Li, J. Wang, Y. Yacoob, and L. Wang. Mitigating hallucination in large multi-modal models via robust instruction tuning. In *ICLR*, 2023.

[37] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. *NeuRIPS*, 36, 2024.

[38] P. Lu, S. Mishra, T. Xia, L. Qiu, K.-W. Chang, S.-C. Zhu, O. Tafjord, P. Clark, and A. Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *NeuRIPS*, 35:2507–2521, 2022.

[39] M. Malinowski, M. Rohrbach, and M. Fritz. Ask your neurons: A neural-based approach to answering questions about images. In *ICCV*, pages 1–9, 2015.

[40] P. Manakul, A. Liusie, and M. J. Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*, 2023.

[41] A. Masry, X. L. Do, J. Q. Tan, S. Joty, and E. Hoque. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In *ACL*, pages 2263–2279, 2022.

[42] M. Mathew, D. Karatzas, and C. Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209, 2021.

[43] N. Methani, P. Ganguly, M. M. Khapra, and P. Kumar. Plotqa: Reasoning over scientific plots. In *WACV*, pages 1527–1536, 2020.

[44] T. Mickus, E. Zosa, R. Vázquez, T. Vahtola, J. Tiedemann, V. Segonne, A. Raganato, and M. Apidianaki. Semeval-2024 shared task 6: Shroom, a shared-task on hallucinations and related observable overgeneration mistakes. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 1980–1994, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[45] A. Mishra, S. Shekhar, A. K. Singh, and A. Chakraborty. Ocr-vqa: Visual question answering by reading text in images. In *ICDAR*, pages 947–952, 2019.

[46] OpenAI. Chatgpt. `https://chat.openai.com/`.

[47] R. Rachala and M. Raveendranatha Panicker. Hand-drawn electrical circuit recognition using object detection and node recognition. *SN Computer Science*, 3, 05 2022.

[48] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PMLR, 2021.

[49] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[50] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016.

[51] M. Ren, R. Kiros, and R. S. Zemel. Image question answering: A visual semantic embedding model and a new dataset. *ArXiv*, abs/1505.02074, 2015.

[52] A. Rohrbach, L. A. Hendricks, K. Burns, T. Darrell, and K. Saenko. Object hallucination in image captioning. In *EMNLP*, pages 4035–4045, 2018.

[53] S. Roy, A. Bhattacharya, N. Sarkar, S. Malakar, and R. Sarkar. Offline hand-drawn circuit component recognition using texture and shape-based features. *Multimedia Tools and Applications*, 79, 11 2020.

[54] H. Sharma and A. S. Jalal. A survey of methods, datasets and evaluation metrics for visual question answering. *Image Vision Comput.*, 116(C), dec 2021.

[55] K. Shuster, S. Poff, M. Chen, D. Kiela, and J. Weston. Retrieval augmentation reduces hallucination in conversation. In M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics.

[56] A. Singh, V. Natarajan, M. Shah, Y. Jiang, X. Chen, D. Batra, D. Parikh, and M. Rohrbach. Towards vqa models that can read. In *CVPR*, pages 8317–8326, 2019.

[57] B. Singh, A. Hari, R. Mehta, T. Abhishek, M. Gupta, and V. Varma. XFLT: exploring techniques for generating cross lingual factually grounded long text. In K. Gal, A. Nowé, G. J. Nalepa, R. Fairstein, and R. Radulescu, editors, *ECAI 2023 - 26th European Conference on Artificial Intelligence, September 30 -*

October 4, 2023, Kraków, Poland - Including 12th Conference on Prestigious Applications of Intelligent Systems (PAIS 2023), volume 372 of *Frontiers in Artificial Intelligence and Applications*, pages 2162–2169. IOS Press, 2023.

[58] F. Thoma, J. Bayer, Y. Li, and A. Dengel. A public ground-truth dataset for handwritten circuit diagram images. In *Document Analysis and Recognition – ICDAR 2021 Workshops: Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part I*, page 20–27, Berlin, Heidelberg, 2021. Springer-Verlag.

[59] F. Thoma, J. Bayer, Y. Li, and A. Dengel. A public ground-truth dataset for handwritten circuit diagram images. In *ICDAR*, pages 20–27, 2021.

[60] K. Tian, E. Mitchell, A. Zhou, A. Sharma, R. Rafailov, H. Yao, C. Finn, and C. D. Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. `https://doi.org/10.48550/arXiv.2305.14975`, 2023. arXiv:2305.14975v2 [cs.CL].

[61] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv:2302.13971*, 2023.

[62] L. van der Poel, R. Cotterell, and C. Meister. Mutual information alleviates hallucinations in abstractive summarization, 2022.

[63] N. Varshney, W. Yao, H. Zhang, J. Chen, and D. Yu. A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation, 2023.

[64] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *NeuRIPS*, 30, 2017.

[65] J. Wang, Z. Yang, X. Hu, L. Li, K. Lin, Z. Gan, Z. Liu, C. Liu, and L. Wang. Git: A generative image-to-text transformer for vision and language. *TMLR*, 2022.

[66] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

[67] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola. Stacked attention networks for image question answering. In *CVPR*, pages 21–29, 2016.