



JOÃO PEDRO ALVES BORDALO

BSc in Computer Science

VISUAL DIALOGUE FOR OPEN TASKS

MASTER IN COMPUTER SCIENCE

NOVA University Lisbon
December, 2023



VISUAL DIALOGUE FOR OPEN TASKS

JOÃO PEDRO ALVES BORDALO

BSc in Computer Science

Adviser: João Miguel da Costa Magalhães

Full Professor, NOVA University of Lisbon

Examination Committee

Chair: Henrique João Lopes Domingos

Associate Professor, NOVA University of Lisbon

Rapporteur: Bruno Emanuel Da Graça Martins

Associate Professor, Instituto Superior Técnico

Adviser: João Miguel da Costa Magalhães

Full Professor, NOVA University of Lisbon

MASTER IN COMPUTER SCIENCE

NOVA University Lisbon
December, 2023

Visual Dialogue for Open Tasks

Copyright © João Pedro Alves Bordalo, NOVA School of Science and Technology, NOVA University Lisbon.

The NOVA School of Science and Technology and the NOVA University Lisbon have the right, perpetual and without geographical boundaries, to file and publish this dissertation through printed copies reproduced on paper or on digital form, or by any other means known or that may be invented, and to disseminate through scientific repositories and admit its copying and distribution for non-commercial, educational or research purposes, as long as credit is given to the author and editor.

ACKNOWLEDGEMENTS

I am most grateful to my adviser, Professor João Magalhães, who helped and guided me extensively over the course of this dissertation. With his guidance, I was able to design and validate innovative solutions, and contribute scientifically to our research field. I learnt what it meant to *do research* of the highest standard, being on the boundary of scientific knowledge.

A huge thanks to the research team at Google, with whom I met almost weekly to discuss results and come up with new ideas for our research problems. They provided invaluable insight throughout the course of this dissertation.

I would also like to express my thanks to the VisionBox Project (CC 04040101), for granting me a research scholarship for the past nine months.

I am very proud to have been part of team TWIZ, alongside bright people who always worked collectively towards an end goal. Together, we participated in a challenging competition, and together we won it. Every member was always ready to help another, no questions asked. Working on my dissertation in this environment was specially engaging, and I learnt many lessons along the way. I am happy to consider my former team members as my friends.

I must thank my friends most deeply, without whom this journey would have been much more difficult, academically and personally. I knew I could always count on them, and I am certain we grew together. We shared numerous memorable experiences over the last five years, and here is to hoping that won't change.

Finally, I thank my parents for allowing me to be here now, to get an education, and for always making sure I can pursue all my aspirations.

ABSTRACT

Visual Dialogue is a task requiring an AI agent to hold dialogue with humans in natural, conversational language about visual content. It is a challenging task, requiring a high level of understanding about both the visual world and natural language. The open nature of conversational agents further increases the complexity of this task. This task brings together the two main fields of AI and, being sufficiently detached from typical downstream tasks, serves as a general test of machine intelligence. In addition to the technical challenge, it is also an impactful application of AI, as it can help users when interacting with systems, improving their experience. In the context of this work, we propose to enrich the multimodal aspect of a task assistant, in two ways: 1) **Dialogue Video Moment Retrieval:** We will allow users to navigate through videos by voice. We will extract the video's most relevant frames, create useful data about these frames, and index the data, so it can later be retrieved; 2) **Task-Grounded Image Sequence Synthesis:** We will use Image Synthesis models to illustrate task steps, with an emphasis on sequence coherence.

Keywords: Visual Dialogue, Video Moment Retrieval, Image Synthesis, Multimodal Models

RESUMO

Visual Dialogue é uma tarefa que requer que um agente de IA mantenha diálogos, em linguagem natural, com humanos, sobre conteúdo visual. É uma tarefa desafiante, que requer um alto nível de conhecimento acerca do mundo visual e de linguagem natural. A natureza livre dos agentes conversacionais aumenta a complexidade desta tarefa. Esta tarefa une as duas vertentes mais promissoras da Inteligência Artificial. Sendo suficientemente desconectada das tarefas típicas, pode servir como um teste geral para a Inteligência Artificial. Além do desafio técnico, é uma aplicação importante da área de IA, podendo ajudar utilizadores quando interagem com sistemas, melhorando a sua experiência. No contexto deste trabalho, propomos enriquecer um agente conversacional em dois aspectos: 1) **Dialogue Video Moment Retrieval**: Permitir aos utilizadores navegar pelos vídeos através da voz. Extrairemos as *frames* mais relevantes dos vídeos, criar dados úteis sobre estas *frames*, e indexá-los, para mais tarde podermos recuperá-los; 2) **Task-Grounded Image Sequence Synthesis**: Usaremos modelos de geração de imagem para ilustrar os passos das tarefas, com um foco na coerência da sequência.

Palavras-chave: Visual Dialogue, Video Moment Retrieval, Geração de Imagem, Modelos Multimodais

CONTENTS

List of Figures	ix
List of Tables	xii
List of Listings	xiii
1 Introduction	1
1.1 Scope and Motivation	1
1.2 Guiding Users Through Complex Manual Tasks	3
1.3 The Visual Dimension of Conversational Task Assistants	4
1.3.1 Dialogue Video Moment Retrieval	4
1.3.2 Task-Grounded Image Sequence Synthesis	4
1.4 Objectives	6
1.5 Document Structure	7
2 Literature Review	8
2.1 Word Embeddings	8
2.2 Transformers	9
2.2.1 Positional Embeddings	10
2.2.2 Self-Attention	11
2.2.3 Encoder	12
2.2.4 Decoder	12
2.2.5 Encoder-Decoder	13
2.3 Large Language Models	13
2.3.1 Encoder Models	15
2.3.2 Decoder Models	17
2.3.3 Encoder-Decoder Models	18
2.3.4 Instruction Tuned Models	19
2.4 Vision Transformer Models	22
2.5 Vision and Language Encoder Models	24

2.5.1	Cross-Encoder—Early-Fusion	26
2.5.2	Dual-Encoder—Late-fusion	29
2.6	Vision and Language Decoder Models	31
2.6.1	BLIP	31
2.6.2	BLIP-2	32
2.6.3	LLaVA	33
2.6.4	InstructBLIP	34
2.7	VideoQA	34
2.7.1	Question-based Taxonomy	35
2.7.2	Modality-based Taxonomy	35
2.7.3	Techniques and Algorithms	35
2.8	Image Sequence Generation	36
2.8.1	Image Synthesis	36
2.8.2	Sequence Generation	39
2.9	Summary	42
3	Zero-Shot Dialogue	
	Video Moment Retrieval	44
3.1	Computational Cost of Video Dialogue	44
3.2	Proposed Architecture	45
3.3	Video Frame Processing	46
3.3.1	Caption Embeddings	46
3.3.2	Visual Embeddings	47
3.4	Video Moment Indexing and Retrieval	47
3.5	Evaluation	47
3.5.1	Dataset	47
3.5.2	Results and Discussion	48
3.5.3	Conclusion	48
4	Task-Grounded	
	Image Sequence Synthesis	49
4.1	Challenges of Image Sequence Synthesis	50
4.2	Illustrating Real-World Manual Tasks	50
4.3	Proposed Model: Sequential Latent Diffusion Model	51
4.3.1	Sequence Context Decoder	52
4.3.2	Sequence Conditioned Reverse Diffusion	53
4.4	Experimental Methodology	55
4.4.1	Contextual Caption Generation	55
4.4.2	Dataset	55
4.4.3	Model Details	56
4.4.4	Negative Prompts	57

4.4.5	Human Annotations	57
4.5	Results and Discussion	57
4.5.1	Sequence Context Decoder	58
4.5.2	Fixed Latent Vector Iteration	59
4.5.3	Sequence Generation Results	59
4.5.4	Examples and Qualitative Analysis	61
4.6	Conclusions	64
5	Conclusions	65
5.1	Achievements and Limitations	65
5.2	Contributions	66
5.3	Critical Analysis and Future Work	66
Bibliography		68
Annexes		
.1	Generating Consistent Image Sequences for Real-World Manual Tasks	77
.2	Aligning Natural Prompts and Image Generation	99
.3	TWIZ 2022 - Report	117
.4	Human Annotations	140
.5	Additional Examples	141
.6	Datasets	141
.6.1	Dataset Crawling	142
.6.2	Data Augmentation	144

LIST OF FIGURES

1.1 Multimodal conversational assistants can guide users in executing complex manual tasks (picture taken from Amazon Alexa publicity).	2
1.2 Example of user-bot dialogue while executing a complex manual task.	5
1.3 Video Navigation example.	5
1.4 The properties of the elements in illustrations should remain coherent throughout the whole sequence. Beef that is cooked in step 1, should remain cooked in step 2.	6
2.1 Transformer model [84].	9
2.2 Visualizing positional embeddings.	10
2.3 (left) Scaled Dot-Product Attention. (right) Multi-Head Attention [84].	11
2.4 Encoder-Decoder structure (Figure from [91]).	13
2.5 BERT input embeddings.	16
2.6 BERT model [16].	16
2.7 GPT model, Figure from [93]).	17
2.8 ViT [19] model.	23
2.9 Four categories of V&L models, as proposed in [36]. “The height of each rectangle denotes its relative computational size. VE, TE, and MI are short for visual embedder, textual embedder, and modality interaction, respectively” [36].	26
2.10 ViLT model [36].	27
2.11 FLAVA model [71].	28
2.12 CLIP model [64].	30
2.13 BLIP model architecture. Image from [41].	31
2.14 BLIP-2’s framework. Image extracted from [40]	32
2.15 Latent Diffusion Model architecture. Image adapted from [69].	38
2.16 Story Visualization task. Image adapted from [43].	39
2.17 AR-LDM architecture. Image from [62].	40
2.18 The dependencies between frames are not all equal, and can be measured by the semantic relations between sentences.	42

3.1	Representation spaces for video and queries.	45
3.2	System architecture.	46
3.3	Video Navigation relevance results.	48
4.1	Dependencies between recipes steps. The same colour represents <i>sequentially dependent steps</i>	50
4.2	The proposed method uses the Sequence Context Decoder to maintain semantic coherence. The reverse diffusion process uses a conditioning seed z_T^i that is copied from a previous step and iteration z_k^j . See Equation 4.3.	51
4.3	Example captions generated by InstructBLIP. In the "No Context" example, the model only receives the image. In the "Current Step as Context" example, the model receives the image plus the "Original Step".	53
4.4	Maintaining visual coherence through the use of different memory latent vectors.	54
4.5	Examples of recipe illustrations with different methods for maintaining visual coherence. Each column illustrates steps 1 through 4.	62
4.6	Examples of DIY illustrations with different methods for maintaining visual coherence. Each column illustrates steps 1 through 4.	62
.1	Annotation of the Sequence Context Decoder.	140
.2	Annotation of the comparison between visual coherence methods.	141
.3	Annotation of the comparison between different heuristic thresholds.	142
.4	Annotation of the comparison between our method and the best visual coherence method.	146
.5	Annotation of the comparison between our method and the ground-truth images.	147
.6	Annotation guidelines for the comparison between our method and the ground-truth images.	148
.7	Examples of recipe illustrations with different methods for maintaining visual coherence.	148
.8	Examples of recipe illustrations with different methods for maintaining visual coherence.	149
.9	Examples of recipe illustrations with different methods for maintaining visual coherence.	149
.10	Examples of recipe illustrations with different methods for maintaining visual coherence.	150
.11	Examples of recipe illustrations with different methods for maintaining visual coherence.	150
.12	Examples of recipe illustrations with different methods for maintaining visual coherence.	151

.13 Examples of task illustrations with different methods for maintaining visual coherence.	151
.14 Examples of task illustrations with different methods for maintaining visual coherence.	152
.15 Examples of task illustrations with different methods for maintaining visual coherence.	152
.16 Example of a task that is very challenging to illustrate. We can see how the generated images still capture some of the more challenging elements of the steps, such as "Make a note of the longitude and latitude" in step 4, with the images showing a pen.	153

LIST OF TABLES

2.1	Characteristics of the most prevalent Transformer models. All sizes correspond to Base versions, when applicable.	14
4.1	Training parameters for the best model.	56
4.2	Sequence Context Decoder results for different context lengths, configurations and caption type.	58
4.3	The contribution of each error type to the overall performance.	58
4.4	Annotation results for the evaluation of the various methods of maintaining visual coherence. Annotators picked <i>No Good Sequence</i> in 18.99% of the sequences; we report the results for the remaining 81.01%.	59
4.5	Annotation results for the sequences generated with different threshold values, η . Annotators picked <i>No Good Sequence</i> in 20.34% of the generations; we show results for the remaining 79.66%.	60
4.6	Annotation results of the comparison between our proposed method and the winning method from Table 4.4 (Latent 1).	60
4.7	Human annotation results for the comparison of the proposed method with ground-truth images.	61
4.8	Qualitative analysis of the Sequence Context Decoder results.	63

LIST OF LISTINGS

INTRODUCTION

In this chapter, we present the motivation for this dissertation, as well as some needed context. We also present the main objectives of this work and how we plan to achieve them.

1.1 Scope and Motivation

Conversational assistants are computational systems capable of dialoguing and helping users accomplish different goals, see Figure 1.1. Most conversational systems process natural language using machine learning methods, rules, or knowledge based methods. The most successful of these systems use a combination of these methods [24, 45]. Recent groundbreaking advances in Large Language Models (LLMs) have pushed the frontier of conversational assistants into a more natural, knowledgeable and accurate level [61, 46, 60]. However, a significant limitation of these models is their inability to include visual information in the dialogue. **LLMs exceed at understanding and generating natural language, but they lack the capacity to do the same with visual information. The applications of conversational assistants, which can not only understand visual content but also generate it in the context of a dialogue, are numerous and of high-impact.**

Conversational assistants designed for social benefit have a high-impact in domains such as health monitoring and virtual companions [55]. The benefits of applying conversational assistants in these domains are apparent: the isolated population is growing, and such systems can provide entertainment and company, and establish healthier daily routines. One can imagine how much we could improve the quality of life of an elderly person by having them interact with an agent with these capabilities. We could think of an agent capable of dialoguing with these people and reminding them to take their medicine or perform important daily tasks, which they should not forget. With the integration of a visual component, the assistant could even make sure the medicine was correct, and the tasks were being done properly. Aside from helping with their safety, chatbots could hold daily conversation with these people, helping them feel accompanied and less lonely. We could also think of agents which could help people with disabilities, answering questions



Figure 1.1: Multimodal conversational assistants can guide users in executing complex manual tasks (picture taken from Amazon Alexa publicity).

about aspects of the surrounding environment, which they cannot perceive. Although agents like these exist, it is easy to see how they could benefit from visual content. This requires bringing together research from the fields of Language and Vision to advance algorithms that can robustly answer questions about images and videos, and also generate these visual contents.

Outside the domain of social-good, but also in a **wide impact domain**, we have task-bots that guide users through *complex manual tasks*, which might otherwise be too difficult to complete. By generating visual content, they could show the user how to perform tasks instead of just describing the action and, more interestingly, answer questions the users may have. We can think of a task assistant helping someone through an origami. The assistant, with the help of a camera and understanding of the visual world, could correct the person in real time and even show the correct way to perform some folds. Improving the overall experience in this way, these systems could encourage users to complete tasks, without giving up midway.

Hence, dialoguing about visual data and generating visual data are two particularly challenging tasks, requiring a high level of understanding of language and vision. The open nature of conversational agents further increases the complexity of this task, due to the unpredictability of user behaviour and generated visual content [15, 94, 98, 4, 20, 76, 22]. It requires an AI agent to hold dialogue with humans in natural, conversational language about visual content [15] and **reply with both text and images**. It is an important area of AI, bringing together Vision and Language (V&L). It is detached enough from specific downstream tasks, to serve as a “general test of machine intelligence” [15]. It is also an impactful field, which can help guide users and facilitate their experience when interacting with a system.

In this context, the long-term ambition of this thesis is to **provide LLM-based conversational assistants with the capacity to handle visual information in the same way they**

deal with natural language, i.e., they can understand and generate both visual content and natural language.

A by-product of this thesis is the contribution to the creation of a large-scale multimodal conversational agent, TWIZ [45], deployed in a real-world setting (AWS). TWIZ is capable of guiding real-world users through DIY and cooking tasks. This work can enrich users' experience by leveraging multimodal models to provide the conversational assistant with a visual dialogue dimension. Additionally, this dissertation will be developed in collaboration with a research team at Google. This team is focused on improving the factual consistency of language models.

1.2 Guiding Users Through Complex Manual Tasks

A complex manual task is any manual activity, such as cooking or DIY projects, which *involve some level of skill or dexterity, and follows a predefined sequence of manual steps*. Complex tasks are intricate, and sometimes difficult to complete, benefiting from the guidance of a conversational assistant. In this case, the assistant is an Amazon Alexa TaskBot created at Universidade Nova de Lisboa. Figure 1.2 shows an example interaction with a conversational assistant. As one can see, the interaction is rich and complex. The open nature of the conversation calls for different natural language and computer vision tasks to be applied in different contexts. The assistant must be knowledgeable about the task at hand, but also show real-world knowledge. We can also see how the user is steered by what the assistant says and shows, supporting the idea that we can control user engagement by showing them visual content. The user asks to rewatch a specific part of the video, and the assistant is capable of navigating through the video and showing it again. Towards the end of the interaction, the user asks a question which requires a visual aid. The assistant is capable of generating an image that clarifies the user's question, which would otherwise be difficult to answer.

TWIZ, the Wizard of Multimodal Conversational-Stimulus, is one of such multimodal conversational assistants, which is designed to guide users through complex manual tasks. The focus of TWIZ is on being as helpful and knowledgeable as possible, while maintaining a level of engagement that increases the chances of users completing the tasks. In this context, TWIZ TaskBot tackles three main research goals:

Humanly-Shaped Conversations Provide the users with fun and rewarding conversations, which are open in nature, but also knowledgeable and grounded on the task at hand.

Multimodal Conversational Stimulus Combine natural language with a strong visual component, to keep users engaged while performing a task. Visual content can better guide users through each task step, by letting them visualize the objectives. To this end, it is crucial to allow users to interact with the visual aspects shown on the screen and also generate any missing visual support.

Zero-shot Conversational Flows Respond robustly to user interactions which fall outside more predictable behaviour. To have a successful user-assistant interaction, it is important to support *conversation ramblings*, allowing the user to have a more open conversation with the assistant.

As a multimodal conversational assistant, TWIZ employs a variety of techniques from Natural Language Processing and Computer Vision, i.e., intent detection, question answering, information retrieval, image generation, video moment retrieval, *inter alia*. The context of this thesis is on supporting research topic two, Multimodal Conversation Stimulus, by devising algorithms that allow users to robustly interact with visual content, and also fill in any missing visual data.

1.3 The Visual Dimension of Conversational Task Assistants

To research and advance V&L methods that contribute towards the stated ambition, we propose to study and present a comprehensive analysis of the most promising multimodal Transformer and Diffusion models, with special emphasis on the tasks of video moment retrieval and image synthesis.

1.3.1 Dialogue Video Moment Retrieval

A task step's video provides a lot of information to the user, as they normally consist of people demonstrating how to perform the given step. In a fully multimodal assistant, it is natural for a user to try to dialogue about the video content. Video navigation can be troublesome, as the user probably does not know exactly where to jump to on the video timeline, despite knowing what they want to see again. To mitigate this problem, and give users a better experience, we propose to support navigational queries over the videos. We call this Dialogue Video Moment Retrieval, where the goal is to find a specific moment within the video, which answers the user request, see Figure 1.3.

Dialogue Video Moment Retrieval requires creating a useful representation of the video, so that we can find the most relevant moment according to what the user asks. Most Video Moment Retrieval approaches use strategies that have an associated high computational cost. In the context of this work, it is crucial to answer the user's query quickly, to keep the normal flow of the dialogue. To address this, we propose a query-time lightweight approach which shifts the computational cost to the video processing phase, by captioning the moments in the video. This leads to a low latency Video Dialogue system.

1.3.2 Task-Grounded Image Sequence Synthesis

As established in the beginning of the chapter, the visual contents of a complex manual task facilitate the users' experience and improve their engagement. The natural language

1.3. THE VISUAL DIMENSION OF CONVERSATIONAL TASK ASSISTANTS

BOT: "Excellent choice! Let's start Citrus bow-tie pasta recipe.",
 USER: "Will I be using premade pasta from the box or making it myself?",
 BOT: "You will be using premade pasta. The recipe calls for bowtie if available."
 USER: "Okay and what other ingredients do I need? [QUESTION]",
 BOT: "Here are all the ingredients you need besides the bowtie pasta: [RESOURCES]"
 USER: "Now that I have all of the things on your list, can you help me start cooking?",
 BOT: "Sure! Let's start by bringing a large pot of salted water to a boil."
 USER: "Gotcha! Anything I can do while I wait for it to boil? [PARALLEL ACTIONS]",
 BOT: "Yes. You can chop the tomatoes, cilantro and onions. You can peel and cube up the avocado. Watch the video carefully",
 USER: "Can you show me again how the chef chopped the onions? [VIDEO QUESTION]",
 BOT: "Sure, this is the moment in the video. [VIDEO_MOMENT_RETRIEVAL]",
 USER: "Okay, the veggies are ready and the water is boiling. What's up next?",
 BOT: "Go ahead and add the pasta to the boiling water, then cook the pasta according to the directions on the package.",
 USER: "I do know if the pasta is well cooked? [CLARIFYING QUESTION]",
 BOT: "You can cut a string of pasta and check if the inside looks like this [IMAGE_SYNTHESIS]"
 USER: "Okay, I got the pasta all cooked now. What should I be doing next?",

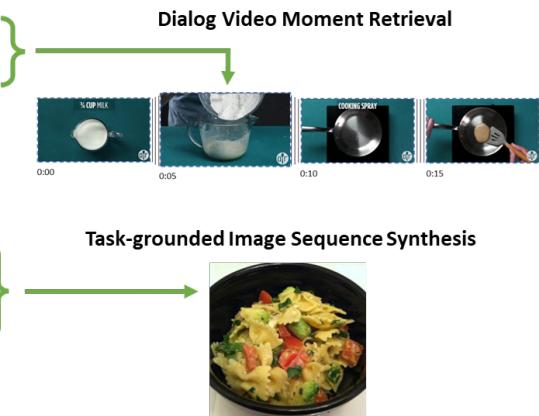


Figure 1.2: Example of user-bot dialogue while executing a complex manual task.



Figure 1.3: Video Navigation example.

descriptions of a task's steps are limited and can be complemented by illustrations, which limit what the user has to guess. To this end, we consider the problem of illustrating all the steps of a task.

Two main challenges arise when illustrating task steps. The first one is that we must ensure that the generated images correctly represent the steps, keeping as much of the information as possible, to lead the user in the right direction. This is challenging because step descriptions are not written with illustrations in mind. Instead, they rely upon a mental model where it is assumed that the user has memorized and executed

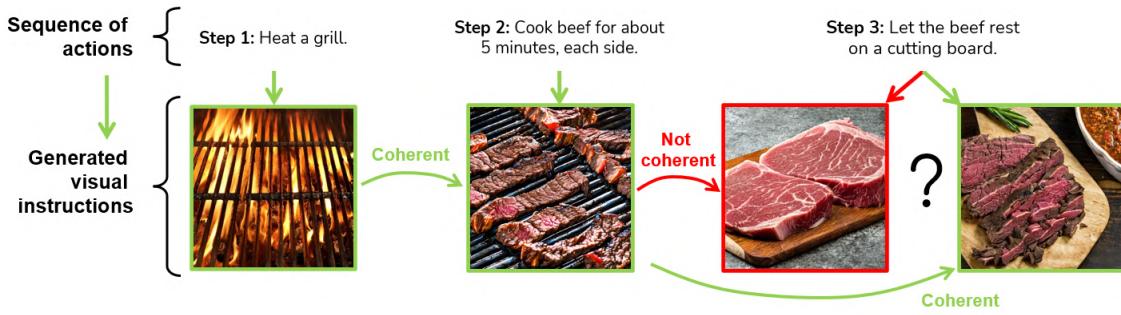


Figure 1.4: The properties of the elements in illustrations should remain coherent throughout the whole sequence. Beef that is cooked in step 1, should remain cooked in step 2.

the previous steps. Secondly, it is important that consecutive illustrations are coherent among themselves, see Figure 1.4. If this is not guaranteed, the user might be misled or end up confused. In sum, it is crucial that the illustrations are coherent with respect to the previous images and to the step descriptions. Only by ensuring that, can we safely improve the users' experience, when interacting with a task assistant.

1.4 Objectives

As mentioned in the beginning of this chapter, this work will be integrated in a conversational agent. The cooking domain will be the main focus of this work. The data we have available for this domain consists of recipes from which we are extracting the videos. In addition to the videos, we have recipe descriptions, instructions for each step, among other data, which we refer to in this work as *recipe knowledge*.

The objectives that we addressed in this thesis were:

Objective 1 - Dialogue Video Moment Retrieval: When the agent receives a user request with the intent of localizing a specific action or event in a video, the system should be able to identify the correct moment in the video and seek the video to the correct video timestamp:

$$f_{DVMR}(V, Q) \mapsto Ts \in V. \quad (1.1)$$

We proposed to leverage recent advances in LLMs and Vision Transformers to shift the video understanding computational cost to the indexing phase and, in this way, achieve a lightweight, yet, accurate approach at query-time.

Objective 2 - Task-Grounded Image Sequence Synthesis: When the dialogue is describing a sequence of complex actions, the assistant will be able to illustrate the action with AI generated images. Moreover, the system should guarantee the coherence of the sequence of generated images.

$$f_{TGISS}(\{I_1, \dots, I_{t-1}\}, A_t) \mapsto I_t. \quad (1.2)$$

We propose to leverage Stable Diffusion’s internal latent embeddings and LLMs’ capacity to rewrite prompts to generate an image sequence that is semantically and visually correct and coherent.

In both objectives, the input *query* is a natural language instruction and visual data provided by the user or by the assistant. The two core algorithms combine both the visual modality and natural language modality to compute the target answer: a video timestamp, in the first objective, and a synthesized image sequence, in the second.

1.5 Document Structure

Chapter 1: Presentation of the research problem and some background.

Chapter 2: Introduction to the Transformer [84] model, backbone of all the current state-of-the-art models. Followed by a study of unimodal models, arriving at multimodal models, and, finally, surveying the current state-of-the-art instruction tuned models, and Image Synthesis models.

Chapter 3: Detailed presentation of our framework for Dialogue Video Moment Retrieval, consisting of a description of the proposed video processing pipeline, its integration in a live system and the evaluation of its performance.

Chapter 4: Detailed presentation of our approach for Task-Grounded Image Sequence Synthesis. Review of common problems present in Image Synthesis, followed by a study of the task at hand, and a proposed model for generating image sequences. Finally, an evaluation of the proposed approach and a discussion about the obtained results.

Chapter 5: Final conclusions about the overall achievements and contributions of this work, with an analysis of its limitations and potential future directions.

LITERATURE REVIEW

In the beginning of the current chapter, we present an explanation of the Transformer [84] model, since it is the backbone of all current state-of-the-art models. This explanation is preceded by some needed background to fully grasp the workings of the model.

After this initial exposition, we begin studying Language models and Vision models separately, to give an overview of each unimodal field. After establishing the background, we present multimodal models, which aim to work effectively over both modalities. For each model, we talk about their architecture, training methods and data used to train. The chief goal is to show the key aspects which made each model stand out among the state-of-the-art models, at the time. Table 2.1 summarizes the characteristics of the currently most prevalent Transformer models.

2.1 Word Embeddings

In order to provide natural language input to models, we need a representation for words and sentences. The standard way to represent word meaning in NLP is through vector semantics, with the idea to “represent a word as a point in a multidimensional semantic space that is derived (...) from the distributions of word neighbors” [34]. Ideally, similar words will be close in the embedding space, while dissimilar words will be farther apart.

Initial approaches [17, 75, 74] relied on counting neighbours and using those values to represent the word vectors. Documents would be represented according to their word count, making it so that similar documents would have similar representation vectors. This approach results in very long sparse vectors, since most words never occur as neighbours of most other words.

Later approaches, such as the methods in *word2vec* [58], introduced short dense vectors or embeddings. These vectors work better in NLP tasks than sparse vectors as they require the classifiers to learn far fewer weights, since they are significantly smaller. The smaller parameter space may help with generalization and prevent overfitting [34]. Intuitively, instead of counting how often words occur near a target word, *word2vec* trains a classifier to predict if a word is likely to appear near the target word. The learned classifier weights

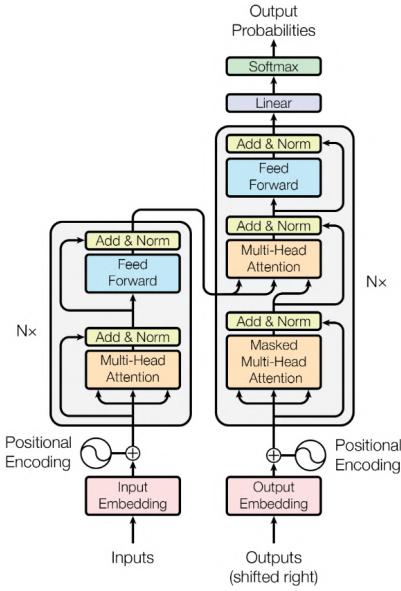


Figure 2.1: Transformer model [84].

are used as word embeddings.

These type of embeddings are called static embeddings, since the method learns a single fixed embedding for each word in the vocabulary. Ultimately, the classifier learns “embeddings that have a high dot product with embeddings of words that occur nearby and a low dot product with noise words” [34].

The main limitation of word embeddings is that they are static, and the same word may have different meanings depending on its context. Approaches based on the Transformer [84] model, introduced in Section 2.2, produce contextual embeddings, which are vectors representing the meaning of tokens in context.

2.2 Transformers

The Transformer [84] (see Figure 2.1) is a sequence transduction model architecture relying on attention mechanisms to draw global dependencies between input and output. The inputs to the encoder and decoder are learned embeddings, which convert input tokens and output tokens respectively to vectors of dimension d_{model} . In order for the model to have information about the order of the sequence, the model sums positional embeddings to the input embeddings (see Figure 2.5). This is to account for the fact that attention model dependencies without regard to their distance in the input and output sequences, as explained in Section 2.2.2.

The Transformer model relies solely on self-attention to compute representations of its input and output. The Transformer is composed of an encoder-decoder architecture. The Transformer’s encoder maps an input sequence of symbol representations to a sequence of continuous representations. Given this sequence, the decoder generates an output sequence of symbols, element by element. At each step, the model consumes the current

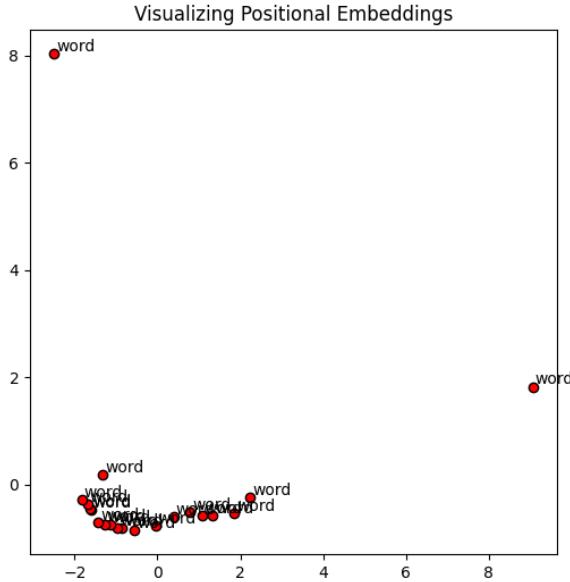


Figure 2.2: Visualizing positional embeddings.

and the previously generated symbols when generating the next. This architecture consists of stacked self-attention and point-wise, fully connected layers for both the encoder and the decoder. Transformers revolutionized the field of Natural Language Processing by achieving strong performances and being more computationally efficient than the previously used models, Recurrent Neural Networks, while also avoiding their main problem of not being able to capture long-term dependencies.

2.2.1 Positional Embeddings

Positional embeddings inject information about the “relative or absolute position of the tokens” [84] in a sequence. The Transformer [84] model uses fixed positional embeddings in the form of sine and cosine functions of different frequencies (see Equations 2.1 and 2.2). Some models take other approaches, instead of using fixed positional embeddings. For example, BERT [16] and GPT [63] use learnt positional embeddings. Positional embeddings do not apply solely to textual inputs. In fact, [19] makes use of learnable position embeddings to retain positional information in the vision domain.

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{model}}) \quad (2.1)$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{model}}) \quad (2.2)$$

To illustrate the effect of positional embeddings, we passed the sentence “*word word word*” through a BERT Transformer and plotted the results. Figure 2.5 shows an example

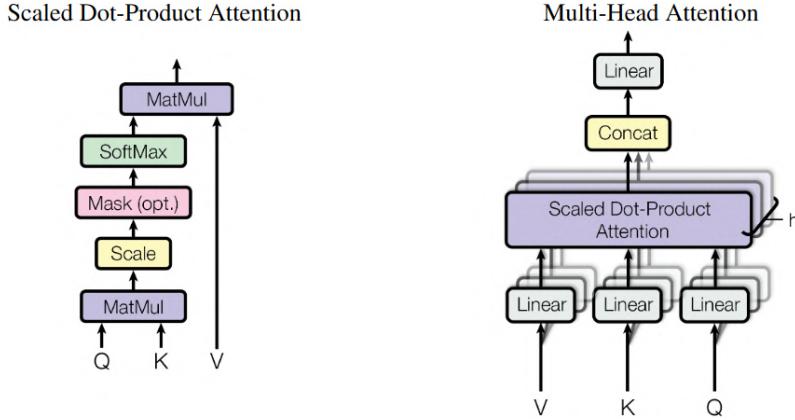


Figure 2.3: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention [84].

input to the BERT model. In our example, the words are all the same so the word embeddings (token embeddings, in Figure 2.5) are expected to be the same. The segment embeddings can be ignored in this context, as they will not change the input. As can be seen in the Figure 2.2, the embeddings are slightly offset. This is the result of positional embeddings, which retain the positional information of each input token. If not for positional embeddings, each word would have the same contextual embedding for the model, as it would lack positional information, further explained in Section 2.2.

2.2.2 Self-Attention

An attention function is a mapping from a query and a set of key-value pairs to an output. The output is a weighted sum of the values, where the weight corresponds to the value of a compatibility function between the query and the corresponding key. The authors of [84] used Scaled Dot-Product Attention (see Figure 2.3),

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (2.3)$$

where d_k is the dimension of the queries and keys. This attention is computed simultaneously on a set of queries, keys and values, packed together into the matrices Q, K and V, respectively.

In an initial step, we compute the similarity between the query and each key, the dot product in equation 2.3. This gives us attention weights, which we normalize. Following that, we apply a *softmax* function to the normalized attention weights and use them as weights for a weighted average of all the values, giving us the final output. The Transformer uses Multi-Head Attention (see Figure 2.3) to perform the attention function in parallel over different projections of queries, keys and values, allowing the model to attend to information from different representation subspaces at different positions. This is a more efficient approach for the computation.

Self-attention, or intra-attention, is an attention mechanism relating different positions of a single sequence to compute a representation for this sequence so, the keys, queries,

and values come from the same input. A self-attention layer, “connects all positions with a constant number of sequentially executed operations” [84], whereas previously used recurrent layers require $O(n)$ sequential operations. Self-attention layers are also faster than recurrent layers when the length of the sequence is smaller than the dimensionality of the latent space. With regard to convolutional layers, the complexity is comparable to the approach taken in this model, with a combination of a self-attention layer and a point-wise feed-forward layer.

Attention mechanisms allow modelling of dependencies without regard to their distance in the input and output sequences. The order of the attention inputs does not matter, making attention permutation invariant.

2.2.3 Encoder

The encoder (see Figure 2.1, left) is composed of a stack of identical layers, each of which consisting of two sub-layers. The first sub-layer is a multi-head self-attention mechanism and the second is a position-wise fully connected feed-forward network. The original model employs a residual connection around each of the two sub-layers, followed by layer normalization. The encoder maps the input into a continuous contextual vector representation of the inputs. Given an input sequence x_1, \dots, x_n , the encoder will produce a sequence y_1, \dots, y_n . We can interpret y_i as a representation of the meaning of the token x_i in the context of the sentence x_1, \dots, x_n . These embeddings can be used as “representations of word meanings in context for any task that might require a model of word meaning” [33]. In this way, it can be used to produce representations of sequences, as seen in approaches like BERT [16] (see Section 2.3.1.1).

2.2.4 Decoder

The decoder (see Figure 2.1, right) is also composed of a stack of identical layers. It also consists of two sub-layers similar to the encoder, but employs an additional one. This additional sub-layer performs multi-head attention over the output of the encoder stack and the output of the self-attention sub-layer. The input to the decoder consists of the outputs it generated so far and the outputs from the encoder. However, we only want the decoder to attend to the positions up to the current one, to have it predict the next position. To accomplish this, the self-attention sub-layer is modified, with masking. Masking consists of setting to $-\infty$ all values in the input of the *softmax* corresponding to illegal connections. This guarantees that the predictions for position i only depend on the previously known outputs, at positions less than i . We call this *masked self-attention*.

We want to model the probability distribution over all possible outputs given our input sequence, $p(z|x)$. This is so we can sample from that distribution to get the most likely output. Here z would be an output sequence, and we would have to model the distribution over all possible output sequences, which is unattainable. To overcome this, we can decompose z into tokens. We get $p(z|x) = p(z_1|x)p(z_2|z_1, x)p(z_3|z_2, z_1, x)\dots$, so the

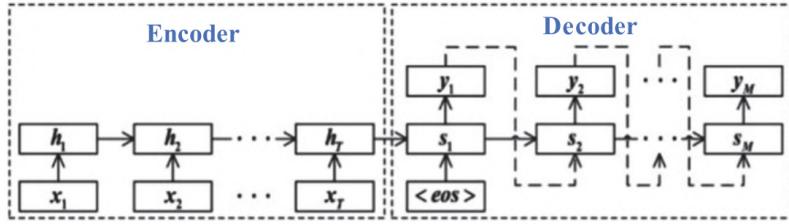


Figure 2.4: Encoder-Decoder structure (Figure from [91]).

model can generate one token at a time, looking at the previously generated tokens [6]¹. As seen in Figure 2.1, the output of the decoder goes through a linear layer which acts as a classifier, over the possible vocabulary. Finally, the model applies a *softmax* function over the output of the classifier, to get output probabilities, so the token with the highest probability can be picked.

Autoregressive or decoder-only models are able to generate sequences of text from the latent representations of input data or feature vectors. These models generate the next token in the output sentence, at each step, attending to the previously generated sequence, making them unidirectional. Decoder-only models, like GPT [63] [7] and PaLM [12], are trained to “autoregressively predict a text sequence” [87].

2.2.5 Encoder-Decoder

The Encoder-Decoder (see Figures 2.1 and 2.4) architecture fits well with the primary application showed in the original paper [84], machine translation. Models like BART [39] and T5 [65] leverage this type of architecture. It is a "more flexible 'text in, text out' model that learns to generate a sequence of tokens y_1, \dots, y_n given an input sequence x_1, \dots, x_m " [59]. The encoder processes its input to generate contextual representations and the decoder is able to leverage them to produce its output, autoregressively predicting the target sequence, one token at a time [92]. In encoder-decoder models, the decoder generally only looks at the "fully processed encoder input" [79]. This type of models is more "straightforward to fine-tune [...] to perform seq2seq tasks" [59], given its sequence-to-sequence nature.

2.3 Large Language Models

Language models assign probabilities to sequences of words [32]. These models take sequences of text as input and produce representations which can be used for downstream tasks. When trained on large corpora, these models end up capable of understanding natural language, due to the tasks they have to learn during training [63, 39]. After the proposal of BERT [16], “the paradigm of pre-training and fine-tuning” [21] became widely popular for Natural Language Processing models, with large Transformer models

¹Talk accessed at 30/01/2023, <https://www.youtube.com/watch?v=EixI6t5oif0&t=2322s&pp=ugMICgJlbhABGAE%3D>

CHAPTER 2. LITERATURE REVIEW

Model	Modal.	Architecture	Tokenizer	Pre. Tasks	Param.	Hidden Size	Heads	Layers	Year
BERT	Lang.	Enc	WordPiece	MLM, NSP	110M	768	12	12	2018
GPT	Lang.	Dec	BPE	CLM	117M	768	12	12	2018
GPT-2	Lang.	Dec	BPE	CLM	1.5B	1600	25	48	2019
GPT-3	Lang.	Dec	BPE	CLM	175B	12288	96	96	2020
BART	Lang.	Enc/Dec	BPE	DAE	140M	768	16	12	2019
T5	Lang.	Enc/Dec	SentencePiece	DAE	220M	768	12	12	2019
ViT	Vision	Enc	-	IC	86M	768	12	12	2020
ViLT	V&L	Enc	WordPiece	ITM, MLM	87.4M	768	12	12	2021
CLIP	V&L	Dual Enc	BPE	CL	63M	512	8	12 (V) 8 (L)	2021
FLAVA	V&L	Dual Enc + Cross Enc	WordPiece	GCL, ITM, MLM, MIM, MMM	350M	768	12 (V) 12 (L) 12 (V&L)	12 (V) 12 (L) 6 (V&L)	2021
OFA	V&L	Enc/Dec	BPE	VG, GC, ITM, IC, VQA, OD, MIM, MLM	180M	768	12	12	2022

Table 2.1: Characteristics of the most prevalent Transformer models. All sizes correspond to Base versions, when applicable.

being pre-trained on large corpora and further fine-tuned for a downstream task [19]. This pre-training has been shown to be effective for improving many NLP tasks such as natural language inference and paraphrasing, which are sentence-level tasks, as well as named entity recognition and question answering, which are token-level tasks [16]. The pre-training of these models differ. The first pre-trained Transformer, GPT [63], was trained as a causal language model while BERT [16] used a denoising objective. Two-stack encoder-decoder models, such as T5 [65] and BART [39], also gained popularity due to their increased performance on classification and sequence-to-sequence tasks [79]. These models showed limited performance on “open-text generation and prompt-based inference” [79], motivating the use of decoder-only models, like GPT-3 [7] and PaLM [12] (see Section 2.3.3.2).

The pre-trained language representations produced by Large Language Models can be applied to downstream tasks with feature-based or fine-tuning approaches. The feature-based approach uses task-specific architectures which include pretrained representations as additional features. The fine-tuning approach, on the other hand, introduces “minimal task-specific parameters” [16], being trained on the downstream task by fine-tuning all parameters.

Recent studies have shown that Large Language Models (LLMs) can learn from few examples without gradient updates, which is referred to as in-context learning, a capability which has been shown to improve with the size of the model and the pre-training data [72].

Typical downstream tasks for language models include:

Natural Language Inference (NLI): The model must determine whether a hypothesis is an entailment, a contradiction or neutral given a textual premise. [16, 48, 39, 10]

Question Answering: The model must output an answer to a given question. [16, 48, 65, 39, 10]

Sentiment Analysis: The model must output the sentiment present in a given input sequence. [16, 48]

Sequence Labelling: The model must assign a categorical value to each element of the input sequence, i.e. Part-of-Speech tagging. [16]

Article Classification: The model must classify a given article according to a set of classes. [16]

Machine Translation: The model must output a sentence in a target language starting from a sentence in a source language. [84, 39, 65]

Summarization: The model must output a summarized version of a document it receives as input. [39, 65]

These models are normally trained with objectives such as:

Masked Language Modelling (MLM)²: A denoising objective where a fraction of the input text tokens are masked and reconstructed from the other tokens. [16, 71, 39, 10, 36]

Next Sentence Prediction (NSP): The model learns to understand sentence relationship by predicting whether a sentence follows another sentence. [16]

Causal Language Modelling (CLM): The model is expected to continue the input. [63, 7, 12]

Most pre-training objectives rely on the huge amount of unsupervised data available, with different architectures leveraging different training objectives. Decoder-only models typically leverage causal language modelling to “mimic auto-regressive generation” [79]. Span corruption was found to be efficient for encoder-decoder architectures. The most successful self-supervised approaches have been variants of masked language models, denoising autoencoders trained to reconstruct text which has been partially masked out.

2.3.1 Encoder Models

2.3.1.1 BERT

BERT [16] is a language representation model “designed to pretrain deep bidirectional representations from unlabelled text by jointly conditioning on both left and right context in all layers” [16]. It can be fine-tuned with a single additional output layer to create SOTA models for a wide range of tasks. The premise of the paper is that previous techniques limit the power of the pre-trained representations. This is because standard language models are unidirectional, which is suboptimal for sentence-level tasks and could be negative

Input	[CLS]	my	dog	is	cute	[SEP]	he	likes	play	# #ing	[SEP]
Token Embeddings	$E_{[CLS]}$	E_{my}	E_{dog}	E_{is}	E_{cute}	$E_{[\text{SEP}]}$	E_{he}	E_{likes}	E_{play}	$E_{\#\text{ing}}$	$E_{[\text{SEP}]}$
Segment Embeddings	E_A	E_A	E_A	E_A	E_A	E_A	E_B	E_B	E_B	E_B	E_B
Position Embeddings	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}

Figure 2.5: BERT input embeddings.

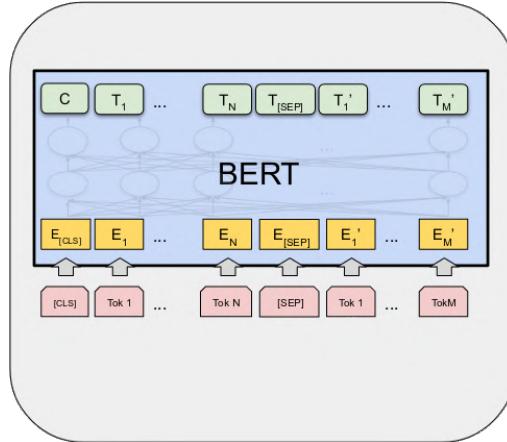


Figure 2.6: BERT model [16].

when fine-tuning for token-level tasks [16]. This model uses an MLM pre-training objective to get around this restriction. This objective enables the representation to “fuse the left and the right context” [16], allowing the pretraining of a deep bidirectional Transformer.

BERT is a multi-layer bidirectional Transformer encoder based on the original implementation ([84]). The authors use two training objectives, the aforementioned MLM objective and a Next Sentence Prediction objective (see Section 2.3).

In order for BERT to be able to handle many down-stream tasks, the input representation is able to represent both a single sentence and a pair of sentences. Receiving a pair of sentences is useful for tasks involving Question-Answer formats, for example. In the context of this paper, a “sentence” is an any span of contiguous text.

The input is the sum of token embeddings, segment embeddings and position embeddings. For the token embeddings, the authors use WordPiece [95] embeddings with a 30k token vocabulary. The input sequence starts with a [CLS] token, whose final hidden state is used as the “aggregate sequence representation for classification tasks” [16]. The two possible sentences which are part of the input sequence are separated with a [SEP] token. Furthermore, the authors add a learnt embedding indicating whether a token belongs to the first or the second sentence, which they call segment embeddings.

The framework proposed in the paper has two steps. In the initial step, the model is trained on unlabelled data with the two previously mentioned tasks, which are unsupervised. The training data for this step is obtained from the English Wikipedia and the BooksCorpus [99] dataset and consists of approximately, 3300 million words. The

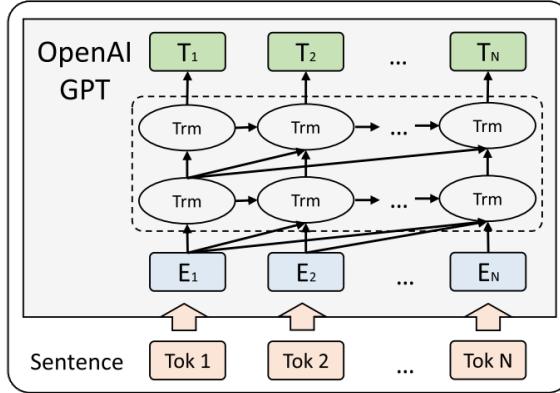


Figure 2.7: GPT model, Figure from [93]).

authors state that it is critical to use a document-level corpus instead of a shuffled sentence-level corpus to extract “long contiguous sequences” [16]. The second step is fine-tuning, whereby the authors pass task-specific inputs and outputs to the model and fine-tune all the parameters. This means that each downstream task has a separate fine-tuned model, initialized from the same pre-trained parameters.

The authors reported new SOTA results for 11 NLP tasks, showing that pre-trained representations reduce the need for task-specific architectures.

2.3.2 Decoder Models

2.3.2.1 Generative Pre-trained Transformer—GPT

The goal of GPT [63] is to combine unsupervised pretraining and supervised fine-tuning to create a model which learns a universal representation capable of transferring to many downstream tasks with minimal adaptation. The models from the GPT [63] family are based on a Transformer decoder (see Section 2.2.4). It receives text and position embeddings and applies multi-headed self-attention over the input context tokens. Following that, a position-wise feedforward layer produces an output distribution over the target tokens [63].

The training consists of two stages. In the first stage, the goal is to learn a high-capacity language model by training on a large corpus of text. Following that, the model is fine-tuned on a discriminative task with labelled data. The model is pretrained on the BooksCorpus [99] dataset, which contains “long stretches of contiguous text”, allowing for the generative model to learn to condition on long-range information using a “standard language modeling” [63] objective and maximizing the following likelihood,

$$L_1(U) = \sum_i \log P(u_i | u_{i-k}, \dots, u_{i-1}; \Theta), \quad (2.4)$$

given the unsupervised corpus of tokens $U = u_1, \dots, u_n$, with k being the size of the context window and Θ being the parameters of the neural network.

After being fine-tuned on a labelled dataset, the model can be evaluated on downstream tasks. The model can be directly fine-tuned for some tasks, like text classification, but

others, such as question answering or textual entailment, have structured inputs. Since the model is pretrained on contiguous sequences of text, these inputs require some modifications. To avoid significant task-specific customization, the inputs are converted to an ordered sequence the model can process. For example, in textual entailment, the premise and hypothesis' token sequences are concatenated with a special token. For question answering, the document and question are concatenated with each possible answer and each sequence is processed independently and “normalized via a softmax layer to produce an output distribution over possible answers” [63].

The authors found that the model outperformed the current SOTA models in a majority of NLI tasks, question answering and common-sense reasoning, semantic similarity and classification. It achieved new SOTA results in 9 out of 12 datasets it was evaluated on.

In sum, GPT [63] introduced a framework to achieve performant models capable of natural language understanding with a single “task-agnostic model through generative pre-training and discriminative fine-tuning” [63]. Pretraining on a large corpus allows the model to learn significant world knowledge [63] which can be transferred to downstream tasks with fine-tuning.

2.3.2.2 LLaMA

Touvron et al. [83] points out that recent work showed the best performances are not achieved by the largest models, but by smaller models trained on more data. This will also lead to faster inference times. The authors train a series of language models of different scales, from 7B to 65B parameters, on a larger number of tokens than usual. The architecture is based on the Transformer [84] model, with some optimizations like pre-normalization, a different activation function, and rotary embeddings, following the approaches of models such as PaLM [12] and GPT-3 [7]. The model was trained on a mixture of publicly available data, unlike others such as PaLM, or GPT-3, with the data covering a diverse set of domains. The entire dataset contains roughly 1.4T tokens. These models, despite being smaller, proved competitive with much larger SOTA foundation models. The authors report that LLaMA-7B outperforms GPT-3, despite being ten times smaller, and that LLaMA-65B is on par with much larger models, such as PaLM-540B. This work is specially important since it shows it is possible to achieve state-of-the-art performance by training only on publicly available data, and also with smaller scale models.

2.3.3 Encoder-Decoder Models

2.3.3.1 BART

BART [39] is a “denoising autoencoder for pretraining sequence-to-sequence models” [39], applicable to many downstream tasks. It combines a Bidirectional Transformer, like BERT [16] (see Section 2.3.1.1) and an Autoregressive Transformer, such as GPT [63] (see

Section 2.3.2.1). In BERT, missing tokens are predicted independently, so it can not be easily used for generation, unlike GPT where tokens are predicted autoregressively. However, since in GPT words can only condition on leftward context [39], the model can not learn bidirectional interactions. Due to BART’s architecture, the authors point out that it can be seen as generalizing BERT, GPT, and other pretraining schemes.

The input corrupted document is initially fed to BART’s bidirectional encoder, and then the likelihood of the original document is calculated with its autoregressive decoder. The model is thus trained by corrupting text with an “arbitrary noising function” [39] and by learning a model which reconstructs the original input text. A distinctive feature of BART is that it allows for any type of document corruption to be applied. In the extreme case where all the information from the source is corrupted, BART is “equivalent to a language model” [39].

BART achieved the “most consistently strong performance” [39] across the tasks the paper evaluated on. The model’s output is fluent and grammatically correct. The authors found that the output is, in general, factually accurate, integrating supporting evidence from across the input document with background knowledge. The strong performance across the evaluated tasks demonstrated that BART pretraining learnt a strong combination of NLU and generation [39].

2.3.3.2 PaLM

The idea behind PaLM [12] is to further understand the impact of scale in a few-shot learning setting, since LLMs have been shown to achieve strong performance across various NLP tasks using few-shot learning. With this objective, the authors trained a 540 billion parameter, densely activated Transformer language model. One key takeaway from this work is that there are still continuous improvements from scaling, showing that scaling improvements for large language-models have not plateaued nor saturated [12]. The authors also found what they refer to as *discontinuous improvements* where, for certain tasks, scaling the model from 62 billion parameters to 540 billion resulted in a drastic jump in accuracy, as compared to scaling the model from 8 billion parameters to 62 billion. This suggests that large language-models may only unlock certain new capabilities beyond a certain scale.

2.3.4 Instruction Tuned Models

An important goal of AI is for models to be able to generalize to unseen tasks. NLP models have made significant progress towards this goal, performing tasks given natural language descriptions [7]. Although these models have shown strong performance in a few-shot learning setting, they are less successful at zero-shot learning [90]. Fine-tuning models on collections of tasks phrased as instructions, has produced greater improvements, allowing for the models to respond better to human instructions and reducing the need for few-shot examples [90, 88, 14]. This is known as Instruction Fine-tuning and has

been shown to scale with the number of tasks and size of the model. A key concern is that LLMs are generally not aligned with humans, which can be improved by fine-tuning on human instructions [61]. One problem that arises with instruction tuning, is the fact that it depends heavily on human-written instructions, data which is costly to collect, and lacks diversity, as instructions tend to be popular NLP tasks [88]. This is mitigated by the framework suggested by Wang et al. [88], Self-Instruct, which is a way of instruct-tuning a language model with instruction examples generated by the model itself. This framework has an initial phase, where the LLM is prompted to generate instructions for tasks, leveraging a small pool of human-written examples, which helps create more broad-coverage instructions [88]. Low-quality or repeated instructions are filtered out to improve the newly generated data. The original model can then be fine-tuned on these data, allowing to better follow human instructions. This work showed that models trained with Self-Instruct are capable of competitive performance against state-of-the-art models. This work showed that it was viable to use LLMs to generate instruction datasets, making it cheaper to train instruction following models, such as Alpaca [78], see Section 2.3.4.3. The first instruction tuned models, which we present in the following sections, were language only, but there has also been recent development in vision-and-language instruction-tuning, which we will study in Sections 2.6.3 and 2.6.4.

2.3.4.1 InstructGPT

Scaling models does not, necessarily, make them better at following user instructions, as they can generate outputs which are not truthful, are toxic, or are simply not helpful to the user [61]. Ouyang et al. [61] proposes *aligning* these models with users, by fine-tuning them on human feedback. The proposed method has three steps. The initial step is to collect demonstration data, where labellers are asked to demonstrate the desired output behaviour, given a prompt sampled from a prompt dataset, spanning a wide range of tasks. These prompts consist of instructional data, where the labellers provide the instruction following answer. The collected data is then used to train a GPT-3 [7] model, in a supervised manner, resulting in an SFT model. The second step consists of using the resulting model from step one and sampling various outputs for a given prompt. Labellers are then asked to rank those outputs from best to worse, and that data is used to train a reward model, RM, which predicts the human-preferred output. The third, and final, step fine-tunes the SFT model with Reinforcement Learning, RL, with the output of the RM being used as the reward. The last two steps can be iterated continuously, which can improve the model’s performance, over time. The resulting models are referred to as InstructGPT. The evaluation is based on how aligned the models are with user intentions. A model is aligned with users if it is helpful, truthful, and harmless. The model was evaluated with human evaluation and on public NLP datasets. The authors report that human labellers significantly prefer InstructGPT over GPT-3, across all model sizes. Additionally, the models generalized to held-out labellers, that is, labellers that did

not produce any of the training data, which indicates that the models are not overfitting to the preferences of the training data labellers. It also showed generalization to unseen instructions. On public NLP datasets, the models performed better than GPT-3, on par with GPT-3 with well-chosen prompting, but worse than the SFT model, without the RL training. One important aspect is that InstructGPT models show improvements in truthfulness, over GPT-3. In sum, this work explores how one can align LMs to human intents, aiming to increase the positive impact of these models. The proposed technique seems to be promising for making LMs more helpful, truthful, and harmless, and for decreasing alignment failures.

2.3.4.2 FLAN

Wei et al. [90] proposes improving the zero-shot learning abilities of language models by fine-tuning them on instruction following data. The authors use a decoder-only 137B parameter LaMDA-PT model [81], pre-trained on 2.49 trillion tokens. The instruction tuning dataset is adapted from existing datasets, as creating one from scratch would be resource-intensive. The authors aggregate 62 publicly available text datasets and categorize them into 12 task clusters. For each dataset, they manually compose 10 templates, describing the task in natural language instructions. A key concern is how FLAN performs on unseen tasks, so the authors train various models, each with a held-out task cluster for evaluation. The model is then fine-tuned on the data, with the resulting model, FLAN, being the instruction-tuned version of LaMBDA-PT. The evaluation finds that instruction tuning is effective on tasks naturally verbalized as instructions, such as NLI, QA, or translation, but is less effective on tasks directly formulated as language modelling, where instructions are redundant [90]. FLAN outperforms zero-shot GPT-3 [7] on 20 of 25 datasets and surpasses GPT-3’s few-shot performance on 10 datasets. FLAN also outperforms baselines on tasks such as NLI, reading comprehension, and closed-book QA. To study the role of instructions, this work performs an ablation study, where the model is fine-tuned without instructions, either including only the input and output, or adding the dataset and task names to the input. The ablations perform significantly worse than FLAN, which demonstrates that training with instructions is "critical for zero-shot performance on unseen tasks" [90]. Chung et al. [13] further examines instruction tuning, by studying the impacts of scaling the number of tasks and the model sizes. The mixture of tasks is scaled to 1.8k instruction following tasks. Additionally, a chain-of-thought mixture is added to the training data. The fine-tuning procedure is based on FLAN and the resulting models are Flan-PaLM, Flan-U-PaLM and Flan-T5, based on the PaLM [12] and T5 [65] models, respectively. Increasing the number of tasks improved performance, but the authors found that the improvement occurred mainly when using up to 282 tasks. Some reasons for this could be the fact that the additional tasks are not very diverse, or that the gains come from the model learning to better express knowledge it acquired during pre-training, with more tasks not helping this further. The authors also found

that increasing the model size from 8B to 62B and 62B to 540B parameters, also improved performance substantially. The scaling curves suggest that the model size and number task could be further scaled. Chain-of-thought annotations further improved the models' reasoning capabilities, on all evaluations, and allow for better zero-shot performance. Wei et al. [90] and Chung et al. [13] together show a promising path for instruction tuning, by demonstrating its effectiveness and its ability to scale.

2.3.4.3 Alpaca

Alpaca [78] is an instruction tuned model based on the LLaMA [83] models, see Section 2.3.2.2, and trained with SelfInstruct [88]. The goal of the authors is to train an open-source instruction tuned model for research purposes that is on par with proprietary models, so the research community can study the behaviour of their behaviour. Alpaca is fine-tuned from a LLaMA model on 52k instruction-following demonstrations generated using text-davinci-003 [1]. Alpaca was evaluated by human evaluation by blind comparison with a state-of-the-model, text-davinci-003, and the evaluation set covered a diverse list of user-oriented instructions. The results showed that the performance of this model was comparable to text-davinci-003, despite the model size and amount of data. The authors did find limitations, such as more hallucinations than the baseline.

2.3.4.4 Vicuna

Vicuna [11] is a LLaMA-based [83] model fine-tuned on user-shared conversations with ChatGPT [2]. By fine-tuning on user-shared conversations, Vicuna is capable of competitive performance when compared to models such as Alpaca [78]. The fine-tuning dataset consists of approximately 70k conversations, with inappropriate or low-quality samples filtered out.

2.4 Vision Transformer Models

Vision Models are models which must produce valuable representations from visual inputs. Such representations are then applied to downstream tasks such as image classification, object detection and semantic segmentation, often after fine-tuning. "State-of-the-art computer vision systems are trained to predict a fixed set of predetermined object categories." [64] Some examples of these are the models pre-trained on the ImageNet dataset, an image database with a variety of images associated with a word classes.

Convolutional Neural Networks revolutionized the field of computer vision, and this field "shifted from engineering features to designing (ConvNet) architectures" [49]. This architecture's dominance is explained by its intrinsic inductive biases, such as translation equivariance, which make them well-suited to a wide variety of computer vision applications [49]. As the paradigm of NLP shifted from Recurrent Neural Networks to Transformers [84, 49], and these models proved very effective in this field, approaches

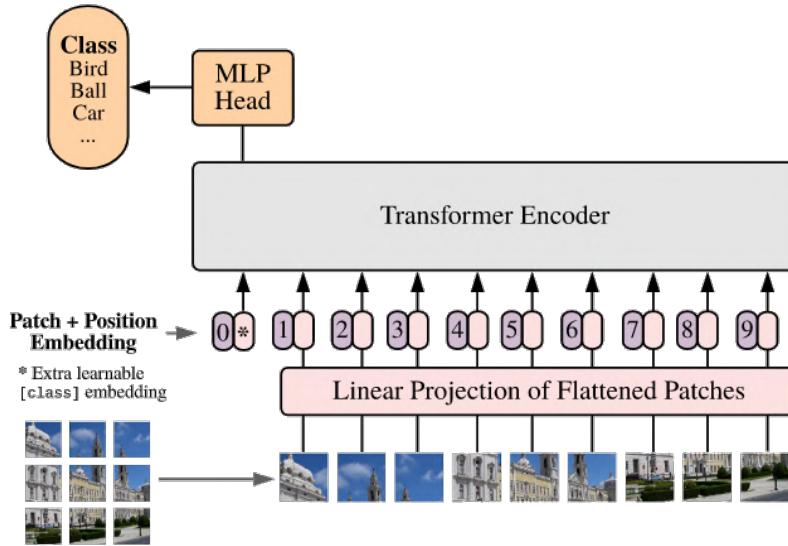


Figure 2.8: ViT [19] model.

like ViT (see Section 2.4) were suggested for applying these models to vision tasks. In this way, the paradigm of vision models also switched to Transformer-based models, instead of Convolutional Neural Networks.

Common pretraining objectives and downstream tasks for vision models include:

Classification: The model must predict the class of the object in the image. [10, 19]

Masked Image Modelling (MIM): A portion, patch, or region of the image is masked and reconstructed from other image portions. [71, 86]

Object Detection (OD): The model must learn to identify the objects which are present in the image, or detecting instances of objects of a certain class in the image. [86]

In the computer vision field, the dominant architectures were convolutional whereas, for natural processing, Transformers became the model of choice. Inspired by the success of Transformers in NLP, the authors of ViT [19] experimented with applying a standard Transformer directly to images, with the fewest possible modifications. Approaches which tried to combine CNN-like architectures with self-attention had been proposed, but not yet scaled effectively. Additionally, the authors do not introduce inductive biases into the architecture, unlike prior works. The authors found that, when trained on mid-sized datasets, these models performed below previously proposed convolutional models. This is hypothesized to be due to the fact that Transformers lack the inductive biases inherent to CNNs, such as translation equivariance and locality. When trained on larger datasets, the authors find that the results improve and the Vision Transformer attains “excellent results” [19] when pre-trained at sufficient scale, approaching or beating the current SOTA on multiple image recognition benchmarks.

The approach in this paper starts by splitting an image into fixed-size patches and linearly embedding each of them, referred to as *patch embeddings*, as seen in Figure 2.8.

This is done by flattening the patches and mapping them to D dimensions with a trainable linear projection, D being the latent vector size of the Transformer. After this, the model uses standard learnable 1D position embeddings, which are added to the previously obtained patch embeddings. These final embeddings are then fed to the Transformer encoder. In addition to the embeddings produced for each patch, the model prepends an extra learnable embedding to the sequence, whose state at the output of the encoder acts as a representation of the overall image. The output of the extra embedding is attached to a classification head implemented by an MLP with one hidden layer at pretraining and a single linear layer at fine-tuning time. The authors claim that the self-attention layers allow ViT to integrate information across the entire image, even in the lowest layers and the ability to integrate information globally is used by the model, seeing as some heads attend to most of the image in the lowest layers of the encoder. Other attention heads were found to have more localized attention, similar to early convolutional layers in CNNs. The model seemed to attend to image regions which were relevant for classification.

As a conclusion, this simple and scalable strategy achieved remarkable performance, matching or exceeding SOTA on many tasks, when coupled with pretraining on large datasets, and showing the viability of Transformers in the vision domain.

2.5 Vision and Language Encoder Models

Vision-and-Language understanding tasks, such as VQA, test a model’s ability to comprehensively understand the semantics of both the visual world and natural language [73]. Leveraging the success of pretraining in the fields of Natural Language Processing and Computer Vision, many works have applied the same idea to vision and language models. “Large-scale pre-training of vision and language Transformers has led to impressive performance gains in a wide variety of downstream tasks” [71]. These models are capable of learning universal cross-modal representations, essential for achieving strong performance in downstream tasks.

One of the biggest challenges in this field is that the models require extensive datasets. Early work relied on human-annotated training data, which is expensive and may require expert knowledge. This factor limited the scale of the datasets. With more recent models such as CLIP [64] and PaLI [10], the necessary huge datasets are crawled from the public web (see Section .6.1).

Some common downstream tasks for Language and Vision Models include:

Visual Question Answering: The model receives a question-image pair and provides an answer. [86, 71, 10]

Visual Entailment: The model receives an image and a hypothesis at inference time, and it must determine whether the hypothesis follows from the image, if it is neutral or if it is a contradiction. [86]

Natural Language for Visual Reasoning (NLVR): The model receives an image pair and a textual statement and determines whether the statement is true about the pair. [36]

Image-Text Matching: The model must predict whether a text and an image it receives were paired originally. [86, 36]

Image Captioning: The model must generate a description for a given image. [86, 10]

Grounded Captioning: The model must generate a description for a given region of an image. [86]

Referring Expression Comprehension: The model must localize a target region in an image, described by a referring expression. [86]

Visual Grounding: The model learns to generate location tokens specifying a region position, for example, locating an object referred by a query. [86]

Having these models learn generic multimodal representations from images paired with sentences is a “fundamental step towards a single interface for vision-and-language (V&L) tasks” [8]. To this end, many V&L models have been proposed, inspired by the success of large pretrained models in CV and NLP. The approach for pretraining these types of model consists in encoding image and text inputs into latent representations, designing an architecture capable of modelling the interaction between the two modalities and using effective pretraining tasks to train them. Pretraining objectives used for V&L models are mentioned in sections 2.5.1 and 2.5.2. After pretraining, the models are expected to have learnt universal vision and language features and can be further fine-tuned on various downstream tasks [21].

For these models to be effective in their tasks, there needs to be interaction between both modalities. The modality interaction schema can be classified into two categories, single-stream or cross-encoder and dual-stream or dual-encoder approaches. In single-stream approaches, the layers collectively operate on a concatenation of image and text inputs. For dual-stream approaches, the inputs are encoded separately, and the interactions are modelled at a later stage. These paradigms are further explained in Sections 2.5.1 and 2.5.2.

The authors of ViLT [36] propose a taxonomy for Vision-Language Models based on two points:

1. whether the two modalities have an even level of expressiveness in terms of dedicated parameters and computation;
2. whether the two modalities interact in a deep network.

These points lead to four models categories for vision-and-language models, as seen in Figure 2.9. There are models which use separate encoders to model textual and visual embeddings, with some of them modelling interactions between the two modalities with simple dot products and shallow attention layers, while others employ more computational

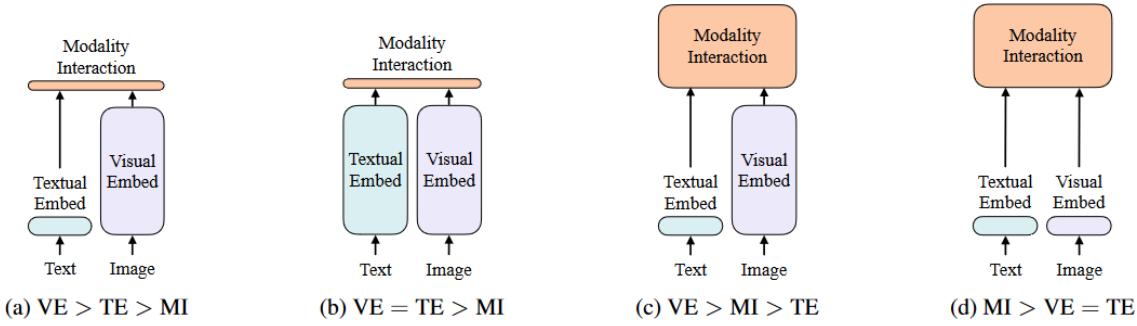


Figure 2.9: Four categories of V&L models, as proposed in [36]. “The height of each rectangle denotes its relative computational size. VE, TE, and MI are short for visual embedder, textual embedder, and modality interaction, respectively” [36].

power in this interaction. Other models like [64] give similar computational power to each Transformer embedder, but still use shallow interaction modelling. The final category of models focuses on strong and deep modelling of modality interaction while giving the same computational power to the Transformer embedders, which is the case of ViLT [36] (see Section 2.5.1.1).

2.5.1 Cross-Encoder—Early-Fusion

Early-fusion happens in the single-stream encoder or cross-encoder paradigm [36]. In this approach, the language and vision inputs are concatenated and fed to a single encoder, referred to as a fusion encoder, see Figure 2.10). This allows for an “early and unconstrained fusion of cross-modal information” [8]. We can further distinguish these models according to whether they use a single Transformer for “early and unconstrained fusion” [71] between both modalities, or if they only allow for cross-attention in specific co-attention layers.

For fusion encoders, various pretraining tasks based on unimodal pretraining have been explored:

Masked Language Modelling (MLM): The model must predict masked words from a caption with help of the paired image. [71]

PrefixLM: The model tries to complete a caption with the help of an image. [89]

Image-Text Matching: The model predicts whether an *(image, text)* pair match. [36, 86]

Masked Region Modelling: The model regresses onto the image features or predicts its object class.

In early-fusion Transformer models, which explicitly target the multimodal vision-and-language problems, the unimodal vision-only or language-only performances are often not a main concern, with some models losing performance on unimodal tasks [71]. The single-stream architecture performs self-attention directly on two modalities, possibly neglecting intra-modality interactions [21].

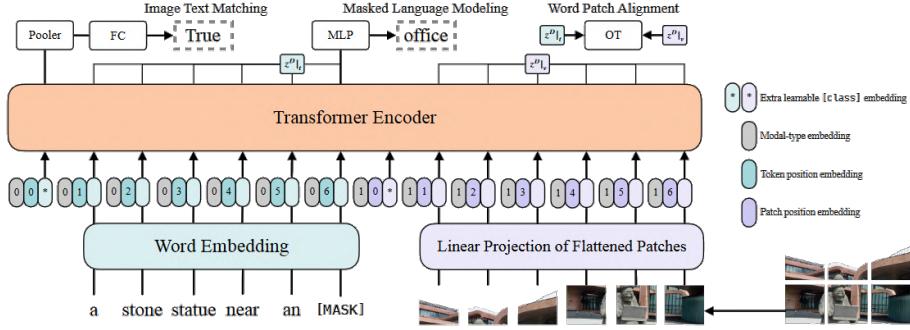


Figure 2.10: ViLT model [36].

2.5.1.1 ViLT

The authors of ViLT [36] point out that, although Vision-and-Language Pretraining (VLP) has improved performance on various downstream tasks, it relies heavily on image feature extraction processes, involving region supervision and the convolutional architectures. They also point out that this has limitations regarding efficiency and speed, since extracting input features requires more computation than the multimodal interaction steps. The expressive power of the models are also upper bounded by the expressive power of the visual embedder [36]. The Vision-and-Language Transformer (ViLT) is a minimal VLP model which processes visual inputs in the “same convolution-free manner” [36] as textual inputs. The removal of deep embedders makes this model shorter and faster to run, without degrading performance on downstream tasks. In terms of architecture, ViLT follows the single-stream approach (see Section 2.5.1). The interaction Transformer weights are initialized from a pretrained ViT (see Section 2.4). The text input is embedded with a word embedding matrix and a position embedding matrix. The input image is sliced into patches, flattened and linearly projected, with this representation being summed with positional embeddings. The text and image embeddings are summed with a corresponding modal-type embedding vector and are then concatenated. The resulting sequence is fed to the Transformer encoder, being iteratively updated until the final contextualized sequence. The model has an extra pooled representation of the whole multimodal input. ViLT uses an ITM training objective, by randomly replacing the aligned image with another and attaching an ITM head to the pooled representation of the whole multimodal input. It also uses Word Patch Alignment (WPA), which computes the alignment score between two subsets of the contextualized sequence, the textual subset and the visual subset. Furthermore, for the textual subset, the model uses MLM.

The authors experimented with ViLT on two widely used types of V&L tasks: classification, and retrieval. For classification tasks, the authors tested the model on VQAv2 [26] and found that it achieved poorer performance, when compared to VLP models with a heavy visual embedder. They suspect that the use of an object detector may facilitate the training of VQA, since questions typically ask about objects. The other classification task was Natural Language for Visual Reasoning (see Section 2.5). On this task, ViLT

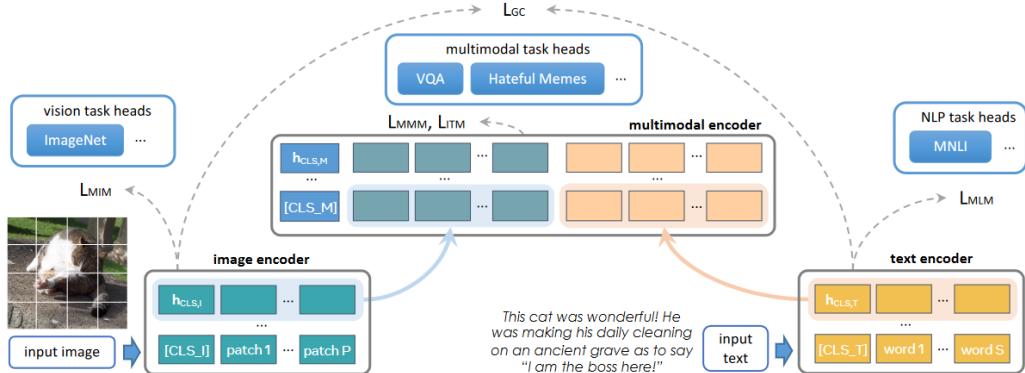


Figure 2.11: FLAVA model [71].

maintained competitive performance, “considering its remarkable inference speed” [36]. For retrieval tasks, the model had stronger performance than the baselines reported in the paper.

ViLT shows that VLP models can be competitive without convolutions or region supervision, and the authors consider it important for future work to focus on the modality interactions inside the Transformer module rather than scaling up the unimodal embedders.

2.5.1.2 FLAVA

The goal of FLAVA [71] is to learn a “foundational language and vision representation” [71] which enables unimodal vision and language understanding and multimodal reasoning within one model. This model learns strong representations through joint pretraining on unimodal and multimodal data. In addition, it encompasses cross-modal alignment objectives and multimodal fusion objectives.

The model has an image encoder which extracts unimodal image representations, a text encoder which obtains unimodal text representations and a multimodal encoder to fuse and align both representations for multimodal reasoning. All three encoders are based on the ViT architecture (see Section 2.4). The FLAVA model can be applied to both unimodal and multimodal downstream tasks in a simple way, as it contains specific classifier heads for vision, language, and multimodal tasks.

FLAVA employs various training objectives, both multimodal and unimodal. The multimodal objectives include:

- **Global Contrastive Loss** - An objective similar to the CLIP model (see Section 2.5.2.1).
- **Masked Multimodal Modelling (MMM)** - An objective which masks both image patches and text tokens and jointly works on both modalities. Previous V&L pretraining approaches focused solely on masked modelling of the text modality. The authors found that this pretraining objective lead to improvements over the contrastive loss pretraining, especially for multimodal downstream tasks.

- **Image-Text Matching** - See Section 2.5.1

For unimodal pretraining, FLAVA uses: Masked Image Modelling (see Section 2.4) and Masked Language Modelling (see Section 2.3). The pretraining has three sources of data: unimodal image data, unimodal text data and multimodal image-text paired data. Initially, the text encoder is pretrained with the MLM objective on the unimodal dataset and the image encoder is pretrained on unpaired images with MIM. After the unimodal pretraining, the whole model is trained jointly on the three types of datasets with round-robin sampling.

FLAVA was compared with several SOTA models on multimodal and unimodal tasks. It outperformed previous multimodal approaches pretrained on public data on language and multimodal tasks, coming close to BERT [16] on several GLUE [85] tasks. When compared to the CLIP model, FLAVA, despite being trained on approximately 6 times fewer data, shows better results on language and multimodal tasks falling slightly under CLIP for vision-only tasks. For a fairer comparison, the authors trained CLIP on the same dataset as FLAVA and found that the latter obtained stronger performance. This suggests that FLAVA could benefit from a larger pretraining dataset. Despite this performance improvement, it is important to point out that FLAVA employs an extra encoder, when compared to CLIP which one would expect to boost performance, so the comparison is not fully fair.

2.5.2 Dual-Encoder—Late-fusion

In the dual-encoder paradigm, the visual and linguistic features are processed by independent stacks of Transformer layers or encoders (see Figure 2.12). The output of these encoders is then fed into cross-modal Transformer layers where intra-modal, within the same modality, interactions are alternated with inter-modal, between the two different modalities, interactions. This late stage modelling of cross-modal interactions is referred to as Late-Fusion. This approach works well for unimodal and cross-modal retrieval tasks, but its lack of fusion usually causes the models to underperform on tasks involving visual reasoning and question-answering, which is where models based on fusion encoders shine.

Dual encoder models use contrastive pretraining. This task consists in giving the model a batch of N (*image, text*) pairs and have it predict which of the $N \times N$ pairings matched originally [71, 64].

2.5.2.1 CLIP

The authors of CLIP [64] point out that SOTA computer vision systems are trained to predict a fixed set of predetermined object categories, which is limiting in terms of generality and usability. To address this, this work proposes to learn directly from raw text about images, demonstrating that the pre-training task of predicting which caption matches which image

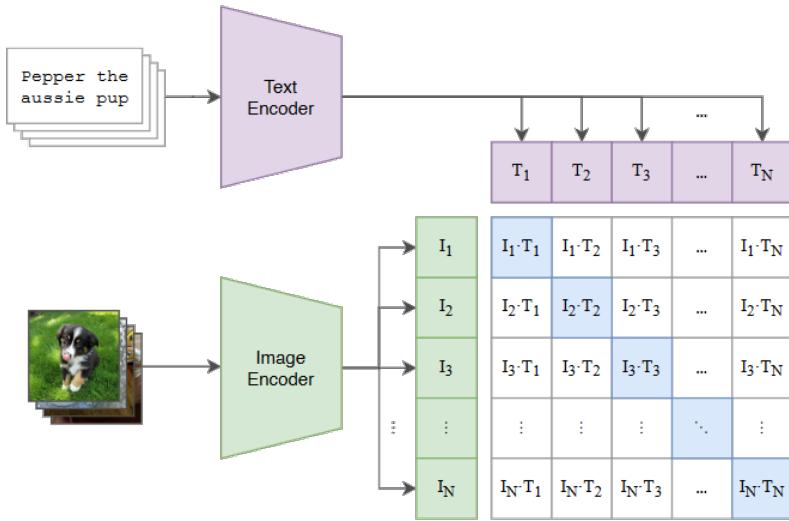


Figure 2.12: CLIP model [64].

is an efficient and scalable way to learn “SOTA image representations” [64]. The model is pretrained on a large dataset of *(image, text)* pairs, crawled from the internet, after which natural language can be used to reference learned visual concepts, or describe unseen ones, which enables zero-shot transfer to downstream tasks. The authors found that CLIP transfers non-trivially to most tasks, often being competitive with fully supervised baselines. This approach is based on natural language supervision—learning perception from supervision contained in natural language—which is easier to scale when compared to crow-sourced labelling for image classification. Natural Language Supervision does not require annotations to be in a “classic «machine learning compatible format»” [64]. Methods leveraging natural language supervision learn passively from the supervision contained in the huge amount of text online. Furthermore, learning from natural language does not only learn a representation but also connects it to language, enabling zero-shot transfer.

The authors point out that the efficiency of pretraining is key for scaling natural language supervision. They tried different approaches but, based on previous findings, suggesting that contrastive objectives can learn better representations than predictive objectives and that generative models require more compute than contrastive models with the same performance, the authors experimented with the “potentially easier proxy task” [64] of predicting only which text as a whole is paired with which image. To this end, CLIP learns a multimodal embedding space by training both an image encoder and a text encoder to maximize the cosine similarity of the image and text embeddings of the N real pairs in the batch, while minimizing the remaining pairs, known as Contrastive Pretraining (see Section 2.5.2). The model is trained from scratch, without initializing the encoders with pretrained weights. It uses a linear projection to map from each encoder’s representation to the multi-modal embedding space. The text encoder is a Transformer and, for the image encoder, the authors experimented with a ResNet50 and with ViT [19].

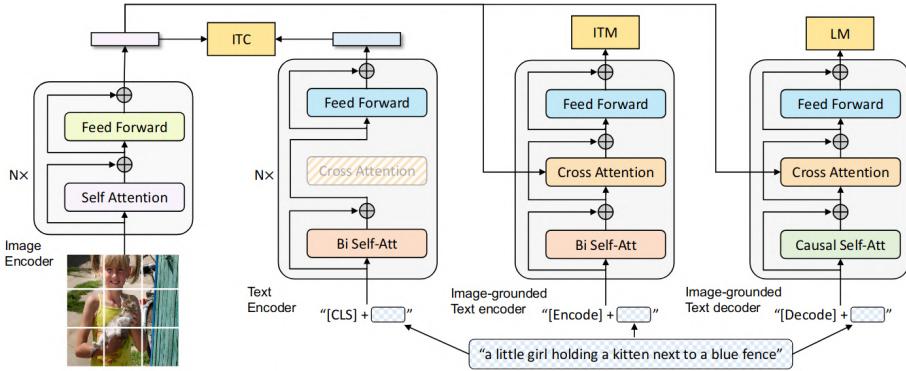


Figure 2.13: BLIP model architecture. Image from [41].

The ability of CLIP to predict if an image and text are paired together is used in zero-shot classification. For each dataset, the names of the classes are used as the set of potential text pairings and the most probable pair is predicted. This is done by precomputing the feature embeddings for both modalities and calculating the cosine similarity. CLIP matches the performance of fully supervised baselines in a zero-shot setting, suggesting that it is “a significant step towards flexible and practical zero-shot computer vision classifiers” [64]. Although the model’s zero-shot performance is strong on many tasks, it falls short on several types of fine-grained classification. It also has issues with more abstract and systematic tasks, such as counting the number of objects occurring in an image, and tasks which are unlikely to be included in the pretraining dataset. Essentially, although CLIP generalizes well to many natural image distributions, the authors observed that the model still generalizes poorly to data that is truly out-of-distribution, in a zero-shot setting.

In conclusion, a model like CLIP learns to perform various tasks during its “task-agnostic web-scale pre-training” [64], which can be leveraged via natural language prompts to enable zero-shot transfer to many existing datasets [64]. At sufficient scale, this approach is competitive with supervised models trained for specific tasks.

2.6 Vision and Language Decoder Models

2.6.1 BLIP

BLIP [41] is a Vision-and-Language pre-training framework, designed to learn from noisy image-text pairs, and which enables a wider range of downstream tasks for vision-language understanding and generation. The model architecture, as seen in Figure 2.13, is an encoder-decoder multimodal mixture, which the authors refer to as MED. It consists of a text encoder, an image encoder, an image-grounded text encoder, and an image-grounded text decoder. The text encoder is the same as BERT [16] and the image encoder is a visual transformer [19]. The image-grounded text encoder injects visual information into the text encoding and outputs a special token $[Encode]$ which serves as a multimodal representation of the image-text pair. The image-grounded text decoder is similar to the

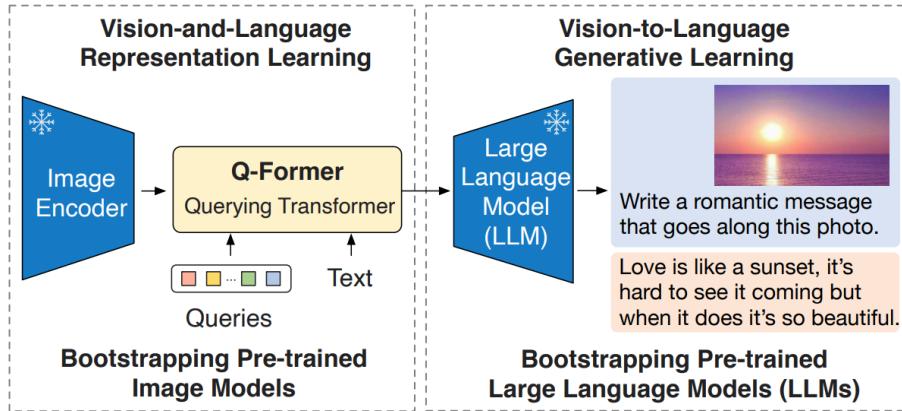


Figure 2.14: BLIP-2’s framework. Image extracted from [40]

image-grounded text encoder, but replaces the bidirectional self-attention layers by causal attention-layers, and the *[Encode]* token by a *[Decode]* token, which signals the beginning of a sequence. This model uses three pretraining objectives, Image-Text Contrastive Loss, see Section 2.5.2, Image-Text Matching Loss, and Language Modelling Loss, see Section 2.5.1. It is common to train VLP models on image-text pairs crawled from the web [64], but some texts often do not accurately describe the images. To tackle this issue, the authors propose *Captioning and Filtering*, *CapFilter*, a module consisting of a captioner, to generate captions given web images, and a filter, to remove noisy image-text pairs. The filtered image-text pairs are paired with human-annotated pairs to form a new dataset, used to pretrain the new model. The authors report improvements when using *CapFilter*, instead of training only on noisy pairs. This model showed substantial improvement over SOTA models in tasks such as Image-Text Retrieval, VQA, and Visual Dialogue. It also achieved competitive performance in Image Captioning, when compared to larger models.

2.6.2 BLIP-2

Li et al. [40] notes how the cost of pre-training vision-and-language models has become increasingly prohibitive. To tackle this emerging issue, the authors propose BLIP-2, a pre-training strategy that bridges together frozen pre-trained vision encoders and large language models. Vision models offer high-quality visual representations, while large language models are capable of strong language generation and zero-shot capabilities. To leverage the unimodal models for vision-and-language tasks, it is necessary to “facilitate cross-modal alignment” [40]. To bridge the modality gap, Li et al. [40] introduces the Querying Transformer, or Q-Former. The Q-Former is a lightweight Transformer, which uses a set of learnable query vectors to extract visual features from the image encoder, which are later fed to the LLM, see Figure 2.14. The Q-Former is trained in two stages. In the first stage, the *representation learning* stage, the model learns which visual representations are most relevant to the text. The Q-Former is connected to the frozen image encoder and is trained on image-text pairs, so that the query vectors can learn to extract visual information

relevant to the text. It uses three pre-training objectives, Image-Text Contrastive Loss, Image-Text Matching, and Image-grounded Text Generation. In the second pre-training stage, the *generative learning* stage, the Q-Former, still connected to the visual encoder, is connected to the LLM. The learned query embeddings are linearly projected to the same dimension as the text embeddings of the LLM by a fully-connected layer. This projection is prepended to the input text embeddings, providing the LLM with the most useful visual information. This trains the Q-Former to output visual representations that can be interpreted by the LLM. The Q-Former is pre-trained on 129M images paired with synthetic captions created with the *CapFilter* method, see Section 2.6.1. For the visual encoder, the authors use ViT, see Section 2.4. For the LLM, they experiment with FlanT5 [90], an encoder-decoder model, and a decoder-based OPT [97] model. With this approach, BLIP-2 achieves state-of-the-art performance on many VLP tasks, while reducing the amount of trainable parameters during pre-training, showing emerging capabilities in zero-shot instructed image-to-text generation. The authors consider this an important step towards a multimodal conversational AI agent.

2.6.3 LLaVA

LLaVA [46] is, according to the authors, the first attempt at multimodal instruction tuning, inspired by the success of this technique in large language models. The architecture of the model connects CLIP’s visual encoder [64, 19], and LLaMA’s language decoder [83]. The extracted image features are linearly projected onto the textual embeddings space, leading to a trainable projection matrix, which is a lightweight and cost-effective approach [46]. One of the key challenges is the lack of vision-and-language instruction-following data. To tackle this issue, and leveraging recent work demonstrating the good performance of GPT models in text annotation tasks [25], the authors use ChatGPT [2] and GPT-4 [60] to collect the needed data. Since these models are text-only, captions and bounding boxes are used to encode an image into its visual features. They generate three types of instruction-following data: *conversation* about the visual content, *detailed description* of the visual content, and *complex reasoning* over the visual content. The data consists of 158k unique instruction-following samples. The training consists of two stages. The first stage is pre-training for feature alignment, where both the visual encoder and the LLM are frozen, and only the projection matrix is trained. The second stage encompasses end-to-end fine-tuning, where the visual encoder remains frozen, but the LLM and the projection layer are trained. The model’s performance is evaluated in a multimodal chatbot setting, with GPT-4, motivated by Chiang et al. [11]. The evaluation found that fine-tuning on instruction data greatly improved the model’s ability to follow user instructions, as expected. Furthermore, adding a small amount of detailed description and complex reasoning questions also improved the model’s overall capabilities, including its conversational capabilities. Having the three types of data yields the best performance. The model is also evaluated on ScienceQA [51], a dataset containing 21k multimodal

multiple choice questions from diverse domains. It achieves results on par with the current SOTA. This work demonstrates the effectiveness of visual instruction tuning and presents an automatic pipeline for creating vision-and-language instruction-following data.

2.6.4 InstructBLIP

Dai et al. [14] points out that, although instruction-tuning has shown success in language models, it is still underexplored for general-purpose vision-and-language models. This work explores instruct-tuning in this setting, and proposes InstructBLIP, an instruction tuned model based on the BLIP-2 models. As mentioned in Section 2.6.2, BLIP-2 consists of an image encoder, an LLM, and a Query Transformer to connect them. InstructBLIP uses a ViT [19] visual encoder, and couples it with different LLMs: FlanT5-XL and FlanT5-XXL [13], and Vicuna-7B and Vicuna-13B [11]. During instruction-tuning, the visual encoder and LLM are kept frozen, and only the Q-Former is fine-tuned. In order to perform a “comprehensive and systematic” [14] study of vision-and-language instruction tuning, the authors transform 26 publicly available vision-and-language datasets into instruction tuning format, and group them into 11 task categories. Among the tasks are image captioning, visual reasoning, VQA, and knowledge-grounded VQA. Half of the datasets are used for instruction tuning, while the rest is held-out for zero-shot evaluation. Additionally, four of the task categories are held-out for zero-shot task evaluation, which is an important capability for generalizable instruction tuned models. For each task, the authors created 10 to 15 distinct instruction templates in natural language, which serve as a foundation for constructing instruction tuning data. The model is initially pre-trained similarly to BLIP-2, see Section 2.6.2 for details, and then fine-tuned on instruction data. The model is evaluated on the set of 13 held-out datasets and achieves new zero-shot state-of-the-art results on all datasets. It consistently outperforms its backbone, BLIP-2, which hints at the effectiveness of vision-and-language instruction tuning. The authors also report a boost in zero-shot generalization on unseen task categories. This work takes a solid step towards generalized vision-and-language models. It presents a study on vision-and-language instruction tuning and shows that the resulting model is able to generalize to unseen tasks with state-of-the-art performance.

2.7 VideoQA

VideoQA is a task which aims to predict the correct answer from a question-video pair. It remains as one of the most challenging V&L tasks as it requires the models to comprehensively understand the videos in different aspects and granularity, from fine-grained to coarse-grained in both temporal and spatial domains, to correctly answer questions [98]. The questions involve not only object, action, activity, or event recognition but also inference of their “semantic, spatial, temporal, and causal relationships” [98]. It poses an extra

challenge when compared to traditional Visual Question Answering (VQA) which is the spatio-temporal nature of videos. Furthermore, in order to learn a good representation, one needs large scale datasets. Due to the variability of video scenes and their textual descriptions, learning a generic embedding space may require a great amount of paired video clips and text captions [57]. Extending existing VQA techniques for VideoQA has proved to lead to suboptimal results, so there is a need for specific techniques that target video.

2.7.1 Question-based Taxonomy

The Question-based Taxonomy splits the task into Factoid VideoQA and Inference VideoQA based on the fundamental challenges present in the questions and answers. Factoid VideoQA directly asks about visual facts, such as location of objects and their attributes. It "invokes little relations to understand the questions and infer the correct answers" [98], emphasizing "the holistic understanding of the questions and the recognition of the visual elements" [98]. Inference VideoQA, on the other hand, explores the logic and knowledge reasoning ability of the system. To this end, it "features various relationships between the visual facts" [98], such as temporal and causal relationships.

2.7.2 Modality-based Taxonomy

The Modality-based Taxonomy divides the task according to the "data modality invoked in the questions and answers" [98]. Normal VideoQA solely invokes visual resources in order to understand the question and provide an answer, emphasizing visual understanding. Multimodal VideoQA involves other resources, such as subtitles or transcripts of the video, alongside the visual component. It mainly challenges multimodal information fusion and "long video story understanding" [98]. Knowledge-based VideoQA "demands external knowledge distillation from explicit knowledge bases or commonsense reasoning" [98].

2.7.3 Techniques and Algorithms

[98] proposed two taxonomies for VideoQA tasks, a Question-based Taxonomy and a Modality-based Taxonomy.

The task can be formulated as multi-choice QA where the model must pick the correct answer from a set of candidate answers or open-ended QA. In open-ended QA, the problem can be classification, word-by-word generation and regression, for counting problems.

A common framework for VideoQA consists of four parts: a video encoder, a question encoder, some type of cross-modal interaction and an answer decoder. The objective is to learn a "mapping of text and video into a shared embedding space, where related text fragments and video clips are close to each other" [57]. Some previous research has

focused on the cross-modality interaction, aiming to “gain understanding of videos under the guidance of questions” [98]. Some techniques presented in the paper follow.

The Attention mechanism (see Section 2.2.2) is used to attend to specific parts of the video in both spatial and temporal dimensions, namely temporal attention and spatial attention. While self-attention is able to model long-range dependencies and can be used in “intra-modal modeling, such as temporal information in the video and global dependencies of questions” [98], cross-modal attention can “attend to both relevant and critical multi-modal information” [98].

Cross-modal Pre-training and Fine-tuning has showed some promising results, with the approach being to transfer large pretrained models to this downstream task by fine-tuning them on smaller scale datasets with strong supervision. Cross-modal pre-trained Transformers achieved superior performance for Factoid VideoQA when compared to other approaches. In addition, this pre-training also “significantly improves the results on inference VideoQA” [98].

Another technique mentioned in the paper is Multi-Granularity Ensemble, which is essential to gain rich information to be able to answer “diverse and unconstrained” [98] questions which may demand video information of “different granularities” [98]. For text, “word-, phrase- and sentence-level feature representations are coordinated to achieve both fine- and coarse-grained information modeling” [98] whereas for vision, “domain, region-, trajectory-, frame- and clip-level feature representations can complement each other to achieve comprehensive video understanding” [98]. Hierarchical Learning aims to “organize multi-modal representation from low-level to high-level, and from local to global” [98], since video elements and their accompanying questions and answers are in different abstraction levels. It progressively processes the multi-granular information, gradually reasoning and aggregating lower level or local information into higher level or global video representation.

Some algorithms have been used for this task. The approach based on Transformer-style models has demonstrated promising results, with strong performance on popular Factoid VideoQA datasets. These models’ performances fall short in Inference VideoQA, however.

Video Moment Retrieval allows locating a specific moment within a video, given a user query. Some works try to score generated moment proposals, others predict the moment start-end indices, or regress moment coordinates [38].

2.8 Image Sequence Generation

2.8.1 Image Synthesis

Image Synthesis or Image Generation models aim to generate images from various input forms, such as natural text, or even other images. These models have applications in fields such as art generation, photo-editing, photo inpainting, and human-computer

interaction [5]. In the context of this work, we focus primarily on text-to-image models, which have shown “unprecedented text-to-image synthetic capacities” [62], and have attracted a lot of research interest [5, 67, 52, 69]. Text-to-Image synthesis aims to represent a natural text prompt visually, ensuring the input prompt is semantically consistent with the generated image. While synthesizing a realistic image on a specific dataset is a well understood problem, generating realistic scenes with many objects with complex relationships is a challenging task [5]. In the case of image-to-image synthesis [56], the goal is to transfer an initial image to a source image, by changing some visual properties, but generally preserving the content.

Modern machine learning approaches to Image Synthesis began with Variational Autoencoder Models [27, 53]. It was soon demonstrated that Generative Adversarial Networks (GANs) improved image fidelity [68, 67]. These networks could, not only generate images on the dataset they were trained on, but also generalize to held-out categories. Further advances, turned to generative models, with autoregressive transformers achieving good performance [67]. The paradigm changed from primarily using GAN-based models to using diffusion models, and these became the standard best performing models for Image Synthesis. These models do not suffer from problems such as mode-collapse and training instabilities, like GANs, are capable of modelling complex distributions of natural images more efficiently [69], and seem to have a good inductive bias for images [30].

2.8.1.1 DALL-E Models

DALL-E [67] is a transformer [84] model trained to autoregressively model text and image tokens as a single stream of data. The authors note that previous work used pixels directly as image tokens, which is inefficient. This is addressed by using a two-stage training procedure. In an initial stage, they train a discrete variational autoencoder, which compresses each RGB image into a lower dimensional grid of image tokens. This reduces the context size of the transformer by a considerable margin, without loss of visual quality. In the second stage, the encoded text tokens are concatenated with the image tokens, and that resulting vector is used to train an autoregressive transformer to model the joint distribution over these tokens. The model was initially trained on Conceptual Captions, an extension of MS-COCO [44]. As the authors scaled the model, they created a new dataset by collecting 250 million image-text pairs from the internet. This dataset includes Conceptual Captions and a filtered subset of YFCC100M [80]. After training, the model is evaluated in comparison to previous GAN-based approaches. It obtained FID scores³ within 2 points of the best prior approaches, when tested on MS-COCO [44], despite working in a zero-shot setting. The model is also evaluated by humans in comparison to DF-GAN [77], which had reported the best Inception Score⁴. DALL-E was chosen as the most realistic 90% of time, and was considered as better matching the input caption

³FID Score is a metric used to assess the quality of generated images.

⁴Inception Score is an algorithm used to measure the quality of generated images.

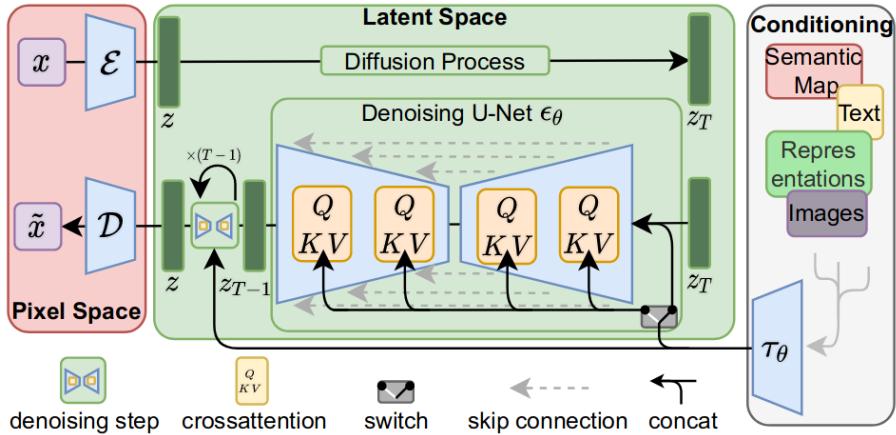


Figure 2.15: Latent Diffusion Model architecture. Image adapted from [69].

93% of the time. DALL-E was able to generalize to unseen data, showing the ability to “compose unusual concepts at high levels of abstraction” [67]. The authors found that the autoregressive approach produced good results and was able to generalize, specially at scale. Subsequent DALL-E models, employ diffusion models, which we will study in Section 2.8.1.2.

2.8.1.2 Diffusion Models

Diffusion models are latent variable models. These models progressively inject noise over the samples and then learn to reverse this process for sample generation [96]. A denoising diffusion probabilistic model, or DDPM [30], employs two Markov chains. One which performs a forward diffusion process, perturbing the data with noise, and another which applies a reverse diffusion process, learning to predict the noise that was added in and thus, generate a sample from noise. The forward process takes the sample from the data distribution to a prior distribution, which is usually random Gaussian noise. To generate a sample, DDPMs generate a noise vector from the prior distribution and then gradually remove noise. The denoising process uses a U-Net [70] backbone. With the DDPM approach, authors report an improvement in FID Scores, when compared to the SOTA models, presenting high quality images. These approaches trained the models in pixel space, which is inefficient. To address this issue, Rombach et al. [69] apply the diffusion model over a latent space, which reduces complexity, while preserving the models’ quality and flexibility. This model transforms the diffusion process into the low-dimensional latent space through an encoder $z = E(v)$ and recovers the real image with a decoder $v = D(z)$. This lower dimensional space is “perceptually equivalent” [69] to the data space. The complete model is also conditioned on an input y by augmenting the U-Net backbone with a cross-attention layer to support the encoded input $\tau_\theta(y)$. These models are known as Latent Diffusion Models, or LDMs. The conditional LDM is learned

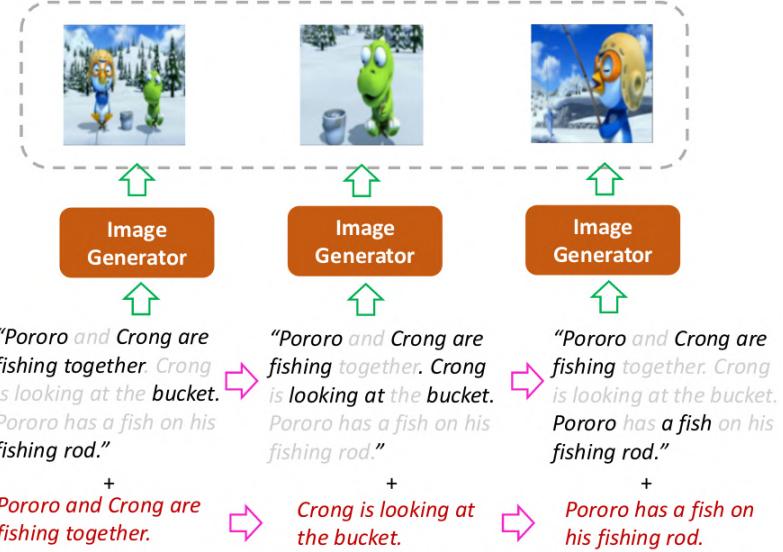


Figure 2.16: Story Visualization task. Image adapted from [43].

with the following loss,

$$\mathcal{L}_{LDM} = \mathbb{E}_{E(v), y, \epsilon, t} \left[\|\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))\|_2^2 \right], \quad (2.5)$$

where the conditioning encoder $\tau_\theta(y)$ in Eq. 2.5 uses the entire set of tokens in y to condition the U-Net denoising process in the LDM backbone. The full model architecture can be seen in Figure 2.15.

2.8.2 Sequence Generation

Most image synthesis work focuses on individual image generation, processing a single, short caption as input [52]. In real-world practical applications, it is often necessary to generate a sequence of coherent images, process long narratives, and generate more than one image to represent the input prompt [62, 52]. When attempting to represent a sequence of this nature, simply generating every image individually, with no regards for continuity, would lead to low coherence.

When dealing with Sequence Generation, we can consider the Story Visualization and Story Continuation tasks. Story Visualization consists of generating a coherent sequence of images, based on a multi-sentence paragraph or series of captions forming a narrative [43, 62, 52], see Figure 2.16. It is a challenging task, requiring understanding of both natural language and images [43, 52]. Story Continuation is a variation of Story Visualization, in which the generated sequence is initially conditioned on a source image.

2.8.2.1 StoryDALL-E

Maharana, Hannan, and Bansal [52] acknowledges the recent improvements in text-to-image models, but points out how these models are ill-suited for the Story Visualization task. The authors add that this task fails to generalize to unseen data, such as plots and

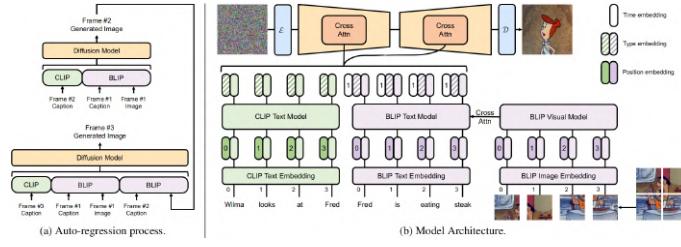


Figure 2.17: AR-LDM architecture. Image from [62].

characters, and propose the Story Continuation task. As it is difficult to collect a large corpus of data for these tasks, due to the need for continuity and an explicit narrative among the images, the authors propose to leverage pretrained text-to-image models and improve generation for story continuation. To do this, the authors fine-tune the pretrained model, DALL-E [67], on a sequential text-to-image generation task, also allowing the model to copy from previous input. They add additional layers to the model, to copy relevant output from the initial scene, and add a self-attention block for generating story embeddings, which provide global semantic context of the story by pooling information from all captions. These new modules are trained from scratch, during fine-tuning. The model is evaluated on the PororoSV [35, 43], FlintstonesSV [28], and DiDeMoSV, adapted from DiDeMo [29], datasets. These are story visualization datasets, with the first two containing animated images and the third one consisting of a video captioning dataset, adapted for the task. To adapt these datasets to story continuation, the first frame is used as the source image. For a fairer comparison of the new model, the authors developed a GAN-based model, StoryGANc, for the story continuation task, following the general framework of StoryGAN [43]. They found drastic improvements in FID Score over the GAN-based model, on the PororoSV and FlintstonesSV datasets, indicating superior visual quality of the generated stories. The character classification scores were lower with StoryDALL-E, however, indicating a shortcoming in the ability of the model to generate the distinct and fine details of characters. For the DiDeMoSV dataset, StoryDALL-E largely outperforms GAN models, in terms of FID Score. Finally, the authors conduct human evaluation, to capture the overall quality of the generated stories. Humans see a comparison between StoryDALL-E and StoryGANc and the ground truth caption. StoryDALL-E outperforms StoryGANc in terms of visual quality and relevance, with a higher win rate in each of the datasets. One factor to take into account is that StoryDALL-E produces higher quality images, specially by having access to large pretraining data and using a Variational Autoencoder designed for reconstructing higher resolution images.

2.8.2.2 AR-LDM

Pan et al. [62] proposes a new approach for tackling the story visualization and story continuation tasks, based on a history-aware autoregressive latent diffusion model, or AR-LDM. This model leverages the history captions and images for future frame generation.

The authors point out that previous approaches assume conditional independence between frames and generate according to the captions. AR-LDM abandons this assumption and additionally conditions the generation on history images. This model uses a generative network similar to Rombach et al. [69], which performs the diffusion processes over a latent space. The novelty is in the use of a history-aware conditioning network, used to encode the history caption-image pairs into a multimodal condition, which guides the denoising process. This conditioning network consists of a CLIP [64] text encoder, which encodes the current caption, and BLIP [41], which encodes the previous image-caption pair. BLIP uses cross-attention over both modalities, deeply integrating them, which the authors claim helps ground the entities generated in history frames. The model is trained on the PororoSV [35, 43], FlintstonesSV [28] and VIST [82] datasets. The first two datasets have animated images, so VIST is used to test the model in a setting closer to a real-world scenario. Two types of evaluation were conducted, quantitative evaluation, with an FID Score, and human evaluation. AR-LDM achieved SOTA FID Scores, outperforming previous methods by a large margin, both on story visualization and story continuation. FID is a measure of image quality, and it is also important to assess the relevance and consistency of the generated images. This was measured in comparison to StoryDALL-E, with AR-LDM outperforming it in terms of visual quality, relevance, and consistency, with the results expressed as a win rate, as chosen by humans. When compared to the ground-truth images, AR-LDM is still far behind in terms of consistency, despite being comparable in terms of visual quality and relevance. This work shows a promising approach to story visualization and continuation, by conditioning the generation on history captions and images, and achieves SOTA results on these tasks.

2.8.2.3 ACM-VSG

Feng et al. [23] notes that, despite its success, AR-LDM [62] conditions the current generation on all previous frames and captions equally, see Section 2.8.2.2 for details. This is a limitation, since not all frames are similarly related. This work is conducted under the assumption that the relationship between images can be measured by the semantic relations between sentences, see Figure 2.18 for an example. The authors propose ACM-VSG [23], an approach which selectively adopts historical text-image data for the generation of the new image. The model consists of an adaptive encoder, a conditional diffusion model, and an adaptive guidance. The adaptive encoder automatically finds the relevant historical text-image pairs and encodes them into *condition vectors*. It combines a CLIP [64] text encoder, which outputs the current text prompt vector, a BLIP [41] text-image encoder, which encodes the historical vectors, and a cross-attention module, which filters history information. The resulting vector is a concatenation between the encoding of the current text prompt and the filtered historical data. This vector is the input for the conditional diffusion model, which outputs an image. The adaptive guidance enforces the consistency between the newly generated images and the past images. It calculates the CLIP similarity

Id	Text	Image
1	Poby talks and gathers his hands. Poby, Loopy and Pororo are clapping their hands.	
2	There are eight glasses of different colors of fluids. Eddy is standing in front of the glasses.	
3	Eddy is holding two sticks and talking. Poby, Loopy, Pororo and Crong are sitting around the table.	
4	Poby, Loopy, Pororo and Crong are clapping.	
5	Eddy is standing in front of the eight glasses containing different colors of fluids.	

Figure 2.18: The dependencies between frames are not all equal, and can be measured by the semantic relations between sentences.

score between the current text and the historical texts and selects the text-image pair with the highest text similarity. When that similarity is above a threshold, the image is used to guide the sampling process of the diffusion model. The model is trained and tested on the PororoSV [35, 43] and FlintstonesSV [28] story visualization datasets. Only the diffusion model and cross-attention model parameters are trained, with the VAE, CLIP and BLIP models remaining frozen, to speed up training. Among others, the authors compared ACM-VSG to StoryGAN [43], StoryGANc [52], StoryDALL-E [52], and AR-LDM [62], and reported FID scores, Character F1 score (Char-F1), which calculates the proportion of characters present in the generated images matching the characters in the original input, and Frame Accuracy (F-Acc), which evaluates whether all characters from a story are correctly represented in the corresponding images. For story visualization, the model is evaluated on the PororoSV dataset, and it achieves a state-of-the-art FID score. The model also greatly outperforms prior work on the story continuation task, reporting state-of-the-art FID score, Char-F1, and F-Acc. This work shows that adaptive context modelling can help generate higher quality images and more consistent stories [23]. A limitation of the approach is that it is only tested on synthesized datasets of cartoons, and the results might differ when applied to real world images.

2.9 Summary

Throughout this chapter, we studied the most recent advancements in the fields of Natural Language Processing and Computer Vision. We started by looking at how we can represent language in a way these models can use effectively, followed by a presentation of the Transformer model, which is the backbone architecture of most state-of-the-art models.

We then explored this architecture's applications to Large Language Models, which focus on natural language tasks. We looked at Encoder Models, which produce strong natural language representations, and will be used in Chapter 3. We also looked at Decoder Models, which can generate language, and are crucial for the work presented in Chapter 4. Lastly, we looked at Encoder-Decoder models, which bring together both modalities. We then looked at a recent advancement in AI: instruction tuned models. These models can better follow human instructions, and are more aligned with their intents and expectations. Following that, we explored Vision Transformers, which model images and tackle Computer Vision tasks. These models will be used in Chapter 3 to create representations of images. Having studied the two types of unimodal models, we studied models that can interpret both vision and language together, Vision and Language Multimodal Models. We began by looking at Encoder models, which produce a joint multimodal representation that we can use for V&L tasks, and then moved on to Decoder models, which can generate natural language grounded, not only on natural language inputs, but also on visual content. These multimodal models are used in both Chapter 3 and Chapter 4. We then studied approaches to the problems we are tackling in this thesis, VideoQA and Image Sequence Generation. We summarized some common techniques and algorithms for VideoQA. We also studied Image Synthesis models, seeing how the paradigm went from Variational Autoencoders, to GAN-based models, to autoregressive models, and finally arrived at Diffusion Models. Finally, we looked at how these models can be adapted to generate sequences of images following coherent storylines, the main goal of Chapter 4.

In sum, we gathered a general understanding of the fields of natural language processing and computer vision and studied the tools that we will employ throughout this thesis.

ZERO-SHOT DIALOGUE VIDEO MOMENT RETRIEVAL

A video dialogue search system poses many research challenges that aim to answer user questions or requests. There are two main types of questions: those answered with a natural language answer and those answered with a specific video moment:

- Answering questions about the task video requires a video understanding method and a decoder to generate the answer.
- Searching for a specific action or event within a video requires a video understanding method and a matching algorithm to rank the different video moments, according to the user request.

From a research perspective, the first task leverages existing generative visual question answering models, which may be too expensive to run in a live dialogue system. The second task is an interesting research problem, leveraging video and language understanding over a timeline to find the correct moment within the video.

In the present chapter, we tackle the second task and present the implementation and evaluation of a video moment retrieval pipeline, targeting a dialogue setting. Firstly, we detail the implemented pipeline and data preparation and secondly, we present the evaluation of our framework.

3.1 Computational Cost of Video Dialogue

It is crucial for Dialogue Video Moment Retrieval to be fast at query-time, to maintain the flow of the dialogue. Common Video Moment Retrieval approaches use strategies that have an associated high computational cost, as they must encode the video at query-time. We propose to shift the computational cost to the video processing stage, offline. This requires the transformation of the video's representation space, see Figure 3.1.

In Figure 3.1, we depict how the problem exists in the visual domain and in the textual domain. Due to the heterogeneity of the original spaces, we studied three canonical

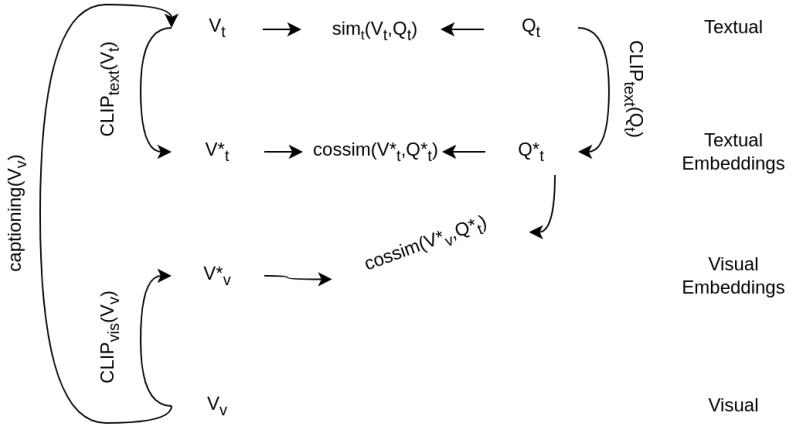


Figure 3.1: Representation spaces for video and queries.

solutions that solve the problem in different spaces. First, the problem can be solved in the textual domain, by applying a captioning method to the different video moments and computing the similarity between the user question and the video captions. Second, the problem can be solved in a text embedding space, with the video captions and user questions being transformed into an embedding space, where the similarity is computed. Third, the problem can be transformed into a cross-modal space, where both video moments and user requests are represented in a common representation space and a similarity function computes the similarity between the two representations.

In this work, we studied these three different approaches with different zero-shot vision-and-language models. The proposed approaches have a minimal computational cost at query-time, since we only need to transform the query into the desired representation space and then an efficient similarity function finds the optimal video moment. Next, we present the proposed architecture and the different methods.

3.2 Proposed Architecture

We implemented the abstraction discussed in the previous section as shown in Figure 3.2. At this stage, we can formalize the Video input in Equation 1.1 as being a set of its key frames and associated knowledge created for each frame, $Video = \{KF_1, KF_2, \dots, KF_n\}$, where $KF = \{Image, Caption\}$. In our initial solution, KF contains only the effective key frame image and the caption which describes it, but it can be extended according to the needs of the system. The recipe knowledge is any available data about the recipe the video belongs to, such as the recipe name or the ingredients. We represent the video in a textual space through image captions, using InstructBLIP [14]. The common cross-modal representation of both video and query, is achieved through CLIP [64] encodings of video frames, video frame captions, and query texts.

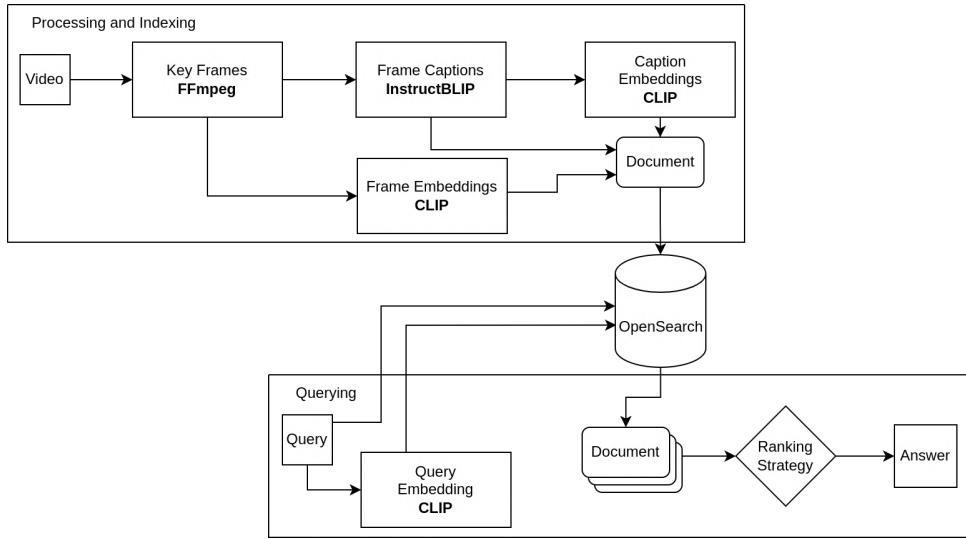


Figure 3.2: System architecture.

3.3 Video Frame Processing

A video consists of a time-ordered sequence of frames “stacked in the temporal dimension” [42]. Because the contents of consecutive frames are often similar, the video has temporal redundancy. This redundancy can be exploited so that some frames need to be “coded independently as a new image” [42]. To this end, the difference between the current or target frame and the other frames in the sequence is coded. These sequences of frames consist of the following frame types:

I-Frame: These frames are treated as independent images. They are also known as *key frames*.

P-Frame: These frames are not independent. The differences between the previous frames are coded by a predictive method.

B-Frame: These frames are also not independent. A B-Frame is predicted bidirectionally from both the previous and future frames.

Because I-Frames contain a full image, they will be extracted from the videos to be used in the pipeline.

3.3.1 Caption Embeddings

Since this solution is based on VQA through image captions, we compute image captions for each extracted key frame, $\text{captioning}(V_v) \rightarrow V_t$. We generate these captions with InstructBLIP [14], a model capable of producing detailed image captions. Additionally, to support embedding based searches, we use a CLIP [64] text encoder to generate caption embeddings, $\text{CLIP}_{\text{text}}(V_t) \rightarrow V_t^*$.

3.3.2 Visual Embeddings

We also want to enhance our search with visual embeddings. To achieve this, we input each key frame into a CLIP visual encoder, and extract the vision embeddings, $CLIP_{vis}(V_v) \rightarrow V_v^*$. This allows us to support image-text retrieval.

3.4 Video Moment Indexing and Retrieval

To support the different types of queries we index the videos with the extracted information, using OpenSearch [3], a “scalable, flexible, and extensible” [3] software suite for “search, analytics, and observability” [3]. It has support for full text queries, NLP and a “range of search features” [3], providing a flexible tool for search applications. It includes a data store and search engine, among other tools. In the context of the present work, it is used as a document search tool. Documents will be indexed according to determined rules, so they can later be retrieved.

Once we have all the data indexed, we can query it, according to our objectives. We support four ways of querying OpenSearch:

Text Query, TQ: Text only query, based on text similarity between the query and the captions, $sim_t(V_t, Q_t)$.

Text Embedding Query, TEQ: Embedding space search based on the similarity between the query and caption embeddings, $cossim(V_t^*, Q_t^*)$.

Frame Embedding Query, FEQ: Embedding space search based on the similarity between the query and frame embeddings, $cossim(V_v^*, Q_t^*)$. These embeddings are aligned, as explained in Section 2.5.2.1.

TQ + TEQ + FEQ: All of the above.

To choose the final result, we count the most frequently occurring frame. In case of a tie, we pick the one with highest OpenSearch score.

3.5 Evaluation

3.5.1 Dataset

The dataset consists of publicly available recipes, scraped from the web. Each recipe has a title, a description, a list of ingredients, and a sequence of step-by-step instructions, which may or may not be illustrated, and include a video. Since we want to be able to navigate in videos, we filter out recipes which do not contain a video. We end up with a set of recipes, where each recipe has a set of steps, consisting of a step instruction and a step instructional video, $R = \{(S_1, V_1), \dots, (S_n, V_n)\}$.

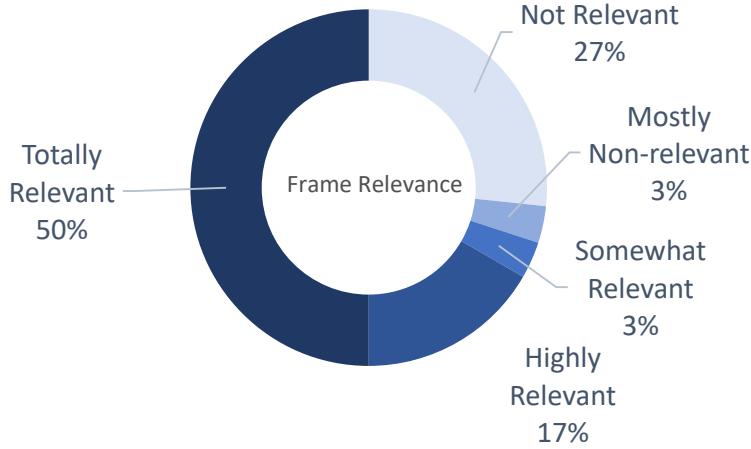


Figure 3.3: Video Navigation relevance results.

3.5.2 Results and Discussion

To evaluate the proposed method, we conducted a preliminary test on 30 user queries over a collection of 10 videos to assess the performance of our pipeline. For each query, we get a video frame, which we judge based on its relevance on a scale from 1 to 5, with 1 indicating a non-relevant frame and 5 indicating a totally relevant frame. Figure 3.3 shows the relevance results. We can see that about 30% of the results were not relevant to the user query. Nevertheless, about 67% of results were significantly relevant to the user query, confirming the strong performance of our solution.

3.5.3 Conclusion

In this chapter, we presented a framework for Dialogue Video Moment Retrieval with strict latency requirements. We developed a solution that performed the time-consuming video processing offline, allowing for fast inference times, when deployed in a live system. Furthermore, the results show that this solution has a strong performance, with the retrieved results being relevant to users' queries in over 70% of the cases.

TASK-GROUNDED IMAGE SEQUENCE SYNTHESIS

When performing a complex manual task, it is limiting to rely only upon textual information. By merely reading a step description, the user is left to imagine and guess some of the more intricate details. The user’s experience can be improved, and facilitated, by complementing the step descriptions with visual content. These make the task steps easier to follow, and keep the users more engaged, by better communicating and representing the text semantics and ideas. In this work, we focus on fully illustrating a recipe’s steps. Illustrating a recipe is challenging, as the sequence of illustrations must combine two keys aspects: (a) accurately portraying the actions outlined in the step descriptions, and (b) ensuring the coherence of all related step images. Training a sequential image synthesis model is costly; to avoid this computational cost, we propose to, instead, modify the image synthesis prompts, tackling the problem through natural language. This is more computationally efficient and robust. A task’s step descriptions describe what the user should do, not what should be represented in the accompanying images, making them inadequate as image generation prompts. To overcome this problem, we propose to train a language model to generate better prompts in the form of image captions. Additionally, to improve the coherence of the overall sequence of images, we propose to condition each image generation on the previously generated images.

In sum, we aim to tackle two main aspects of coherence:

Semantic Coherence The presence and persistence of objects in the images. We propose to tackle this aspect of coherence through the natural language prompts. We want to ensure the image generation prompts contain all the necessary objects that should appear in the final image. For this purpose, we propose a **Sequence Context Decoder**.

Visual Coherence The persistence of backgrounds and object properties, such as colour, texture, or materials, across the sequence. We propose to achieve this through a **Sequence Conditioned Reverse Diffusion** process.

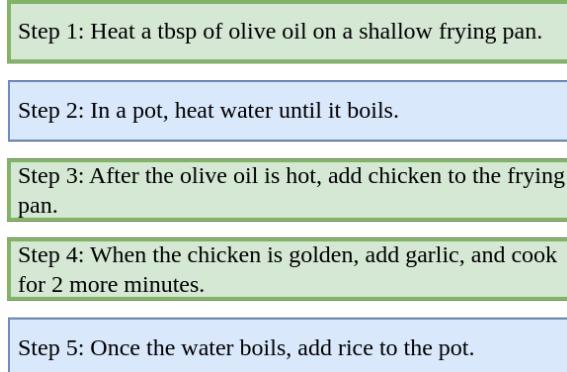


Figure 4.1: Dependencies between recipes steps. The same colour represents *sequentially dependent steps*.

4.1 Challenges of Image Sequence Synthesis

Real-World Sequences. Previous image sequence synthesis approaches [23, 62] work with synthetic cartoon datasets [35, 28], with limited characters and scenes. These datasets contain textual descriptions of the associated images, which are rarely available for real-world complex tasks. Moreover, these approaches show limited performance in real-world sequences [62] or fail to evaluate their solution in this setting altogether [23, 66]. Additionally, there is a lack of publicly available models trained for image sequence synthesis.

Problems with Image Synthesis. Through empirical analysis of generated images, we found image generation errors that were not a result of the conditioning input, such as the prompts or the concepts we were trying to illustrate, but general limitations of the image synthesis model. Among such examples were tiled images, and deformed objects, hands, and faces. We also found that long prompts were detrimental to the generations. Image synthesis models are not prepared to represent many actions or objects, with some context being lost in the generation. Furthermore, these models fail to illustrate distant relationships in the texts. From this analysis, we draw two main conclusions: 1) generated images often contain **artefacts** that we have no control over, and 2) it is imperative to choose the model’s input adequately, as **long noisy prompts lead to poor quality generations**.

4.2 Illustrating Real-World Manual Tasks

We consider a set of manual tasks \mathcal{D} , where each task $TS \in \mathcal{D}$ is composed of a sequence of n step-by-step instructions, $TS = \{(s_1, v_1), \dots, (s_n, v_n)\}$. A task step (s_i, v_i) , consists of a natural language instruction s_i , and its corresponding visual instruction v_i .

Given the sequence of steps $\{s_1, \dots, s_n\}$, our goal is to generate a sequence of images $\{v_1, \dots, v_n\}$, in which v_i visually represents step s_i . A step s_i may be dependent on any number of previous steps, in a non-linear sequential structure [18]. We define the

4.3. PROPOSED MODEL: SEQUENTIAL LATENT DIFFUSION MODEL

notion of *sequentially dependent steps*, a sequence of, not necessarily consecutive, steps which are dependent on each other for preserving all necessary knowledge that should be represented in the target illustration. Figure 4.1 shows an example of this relationship between steps. To generate each image accurately, the model needs to condition its output not only on s_i but also on previous steps $\{s_1, \dots, s_{i-1}\}$; this way, context is preserved even when steps are ambiguous or lack information, e.g. "Add two eggs and mix". In addition to previous step instructions, we also need to condition on previously generated images, $\{v_1, \dots, v_{i-1}\}$, to maintain the visual aspects that are only introduced in the images, such as object properties and background artefacts not mentioned in the text.

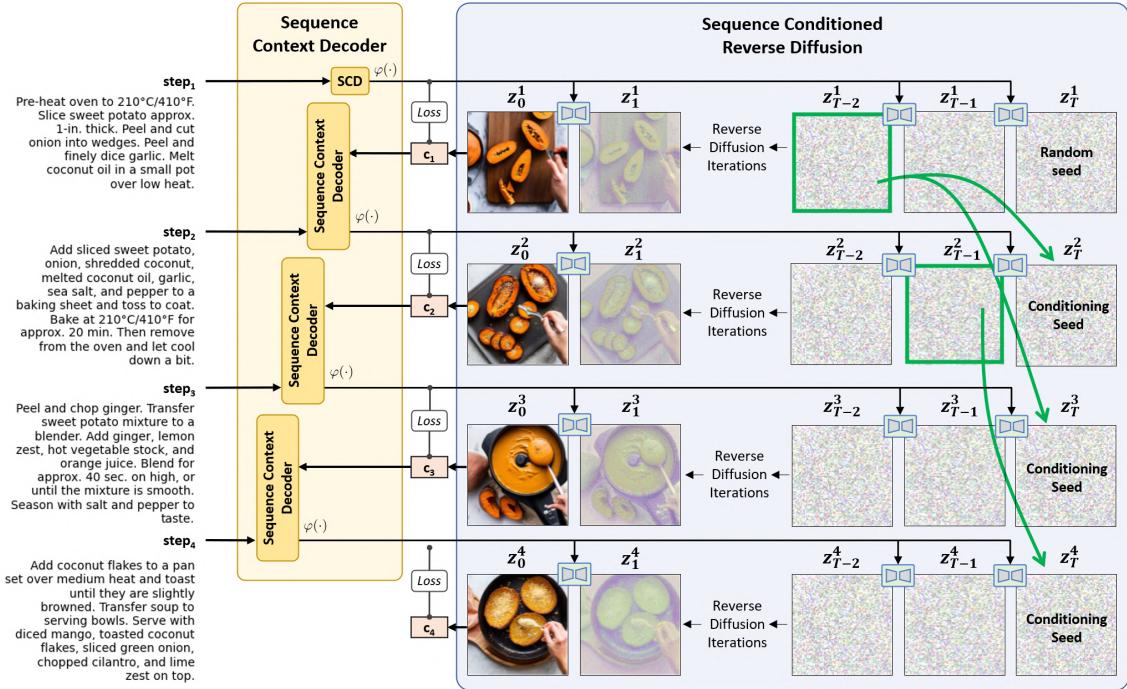


Figure 4.2: The proposed method uses the Sequence Context Decoder to maintain semantic coherence. The reverse diffusion process uses a conditioning seed z_T^i that is copied from a previous step and iteration z_k^j . See Equation 4.3.

4.3 Proposed Model: Sequential Latent Diffusion Model

As introduced in Section 2.8.1.2, the conditional LDM is learned with the following loss,

$$\mathcal{L}_{LDM} = \mathbb{E}_{E(v), y, \epsilon, t} \left[\|\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))\|_2^2 \right], \quad (4.1)$$

where the conditioning encoder $\tau_\theta(y)$ in Equation 4.1 uses the entire set of tokens in y to condition the U-Net denoising process in the LDM backbone. The above expression evidences how the conditional LDM is designed to generate one image v at a time. More relevant to our problem is the fact that the latent vectors z_t are independent across different image generations, since the reverse diffusion process iterates from T to 1 starting with a new random seed z_T for every new image generation.

4.3.1 Sequence Context Decoder

The initial approach for illustrating a step, is to leverage its description. Generally, however, textual step descriptions describe what the user should do at a specific step of their manual task, not the content of the accompanying step images. Additionally, they often contain information that is not visually representable, such as temporal information or multiple actions:

“Heat the Coconut Oil in a wide pan over a medium flame, then add the Onion, Garlic, Scallion, and Ground Black Pepper. Reduce the heat to low for about 3–4 minutes.”

It is also common for step descriptions to not be self-contained, as they depend on the previous step descriptions for context.

To overcome these limitations, for each step s_i , we use a decoder-only model, φ , which we call **Sequence Context Decoder**, to transform the step and its context into a visual caption c_i , which describes the contents of image v_i . We can provide the model with varying context lengths, but we found that longer contexts are detrimental to the generations. To keep the captions concise, but still contextually relevant, we consider a context window of w steps. Step s_{i-w} may not be the initial step in a sequence of *sequentially dependent steps* so, to avoid losing information, we swap step s_{i-w} , by the correspondent caption, which contains the information from all previous steps. Finally, the decoder receives the target step s_i and the context window.

Formally, we define the decoder

$$c_i = \varphi(s_i, \{s_{i-1}, \dots, s_{i-w}\}). \quad (4.2)$$

to generate a contextual caption c_i from its step description s_i and the associated context.

The decoder $\varphi(\cdot)$ is trained similarly to an image caption generator, but instead of receiving images as input, it receives the step description and its context. By training the model to output image captions for the original images that we are trying to replicate, the model learns to generate texts that are more appropriate as image generation prompts. The objective now is learning how to generate better image generation prompts, instead of training the image generation model.

4.3.1.1 Contextualized Step Captions

To train the decoder model $\varphi(\cdot)$, we generate contextual captions for each image in dataset \mathcal{D} using InstructBLIP [14]. To achieve richer and contextualized captions, we prompted InstructBLIP with additional context, conditioning the caption on the recipe steps, in addition to the image. Figure 4.3 shows example captions generated by the model, given a real data point in the dataset. While the original step in the example is long and has multiple actions, the generated captions are concise and better suited for image generation. We can also see how the textual context affects the generated caption’s details.

Original Step: "Slide Dutch baby onto serving platter or cutting board, or serve straight from pan, if desired. Top with fresh berries and sprinkle with lemon juice. Dust with confectioner's sugar and (...)"

No Context: "A person is sprinkling sugar on top of a pancake."

Current Step as Context: "A person is sprinkling confectioner's sugar on top of a Dutch baby pancake."



Figure 4.3: Example captions generated by InstructBLIP. In the "No Context" example, the model only receives the image. In the "Current Step as Context" example, the model receives the image plus the "Original Step".

4.3.2 Sequence Conditioned Reverse Diffusion

To maintain **visual coherence** among images in the sequence, we need to condition the current generation on the previous images. Image-to-image [56] generation follows this principle, but new images are too strongly influenced by previous ones and do not correctly integrate the new aspects present in the step descriptions.

Following the rationale of conditioning every reverse diffusion process on previous processes, we propose to leverage latent vector iterations from early reverse diffusion processes. This leads us to the final formulation of the proposed method,

$$\mathcal{L}_{SLDM}(s_i) = \mathbb{E}_{E(v_i), s_i, \epsilon, t} \left[\|\epsilon - \epsilon_\theta(z_t^i, t, \tau_\theta(c_i = \varphi(s_i, \{s_{i-1}, \dots, s_{i-w}\})))\|_2^2 \right] \quad (4.3)$$

where for each s_i , a new reverse diffusion process starts with a conditioning seed z_T^i copied from a previous step s_j with $j < i$ and a latent vector iteration k corresponding to the latent vector z_k^j , as illustrated in Figure 4.2. Next, we describe the details of how these two variables are determined.

4.3.2.1 Random and Fixed Seeds

To ground the generation, a straightforward method is to set a fixed initial seed z_T^i for every step i in the sequence. By fixing the initial seed, we aim to improve the coherence between generated images. The first two columns in Figure 4.5 demonstrate this approach. We observed a greater homogeneity between generated images when using a fixed seed. Hence, in this setting, all step illustrations share the same random seed, to achieve more coherent scenes and backgrounds.

4.3.2.2 Fixed Latent Vector Iteration

While using a fixed seed can improve the results, we argue that a better solution is achieved by using latent vectors from previous reverse diffusion processes. In particular, *latent vectors that have already been semantically conditioned on past steps*. Figure 4.2 shows how the latent vector representations z_t^i evolve with increasing iterations, until they arrive at the final image $v_i = D(z_0^i)$ for step s_i . These latent representations already contain meaningful

CHAPTER 4. TASK-GROUNDED IMAGE SEQUENCE SYNTHESIS

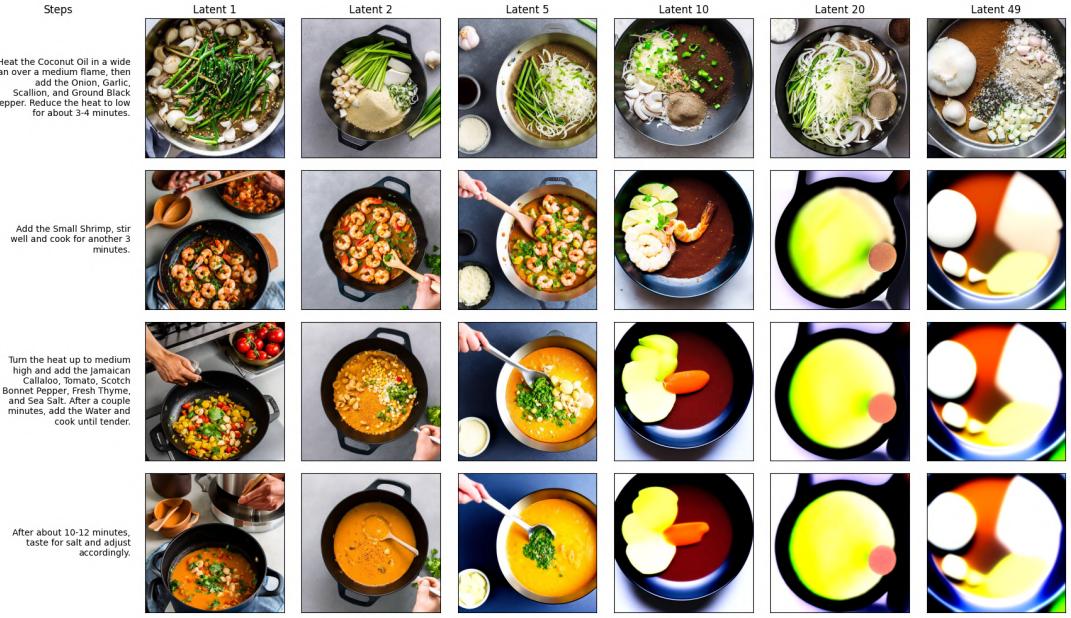


Figure 4.4: Maintaining visual coherence through the use of different memory latent vectors.

information about the image [54], which could be leveraged to improve the coherence of the following generations. Figure 4.4 shows example image sequences generated with fixed latent vector iterations. We used iterations k , with $k \in \{1, 2, 5, 10, 20, 49\}$, and conditioned image v_i , on image v_{i-1} .

4.3.2.3 Choosing the Latent Vector Iteration

Using a fixed latent vector iteration is limiting, as we can not choose how strongly we want to condition the new image on a previous generation. Additionally, in Section 4.3.2.2, we only consider conditioning on the previous step, which may not be the most relevant.

In this new setting, we first need to carefully select which step to choose, to use as input seed for the following step image generation.

A step s_i may be dependent on any previous step $j < i$, $\{s_{i-1}, \dots, s_1\}$. To select the optimal initialization of the reverse diffusion process for step s_i , we start by determining the most similar step s_j as the

$$\operatorname{argmax}_j sim(s_i, s_j \in \{s_{i-1}, \dots, s_1\}) \quad (4.4)$$

where $sim(\cdot)$ represents CLIP [64] text similarity. If this similarity score is above a predefined threshold η , we use s_j to extract the latent vector. If no step s_j has a similarity score above η , we generate image v_i with the shared random seed.

The reverse diffusion process progressively iterates over the latent vectors towards the final image. This means that conditioning the reverse diffusion process on latent vector iterations from a later iteration, i.e. a highly denoised latent vector, would force the

resulting image to be very close to the previous one (Figure 4.4). To decide how strongly we want to condition v_i on the step s_j , we select the k^{th} latent vector iteration as

$$k = \frac{n}{1.0 - \eta} \cdot (\text{sim}(s_i, s_j) - \eta) \quad (4.5)$$

where n is the maximum number of reverse diffusion iterations that we consider.

This brings us to the target reverse iteration vector z_k^j which will be used as a starting seed z_T^i in Equation 4.3, when calculating $\mathcal{L}_{SLDM}(s_i)$. Figure 4.2 illustrates the whole process: the proposed method captures the visual aspects that should be in the image, and the linked denoising latent vector provide the seed to generate a step image.

4.4 Experimental Methodology

In this section, we describe the details of our experiments, including the generation of contextual caption, the dataset and model details, the use of negative prompts, and how we conducted human annotations.

4.4.1 Contextual Caption Generation

As previously described, we provide InstructBLIP [14] with additional context to produce contextualized captions. We experimented with different context lengths, ranging from no context to the full context, i.e., all previous step instructions. Figure 4.3 illustrates how additional context helps generate more contextualized captions, so we ruled out the no context option. We also found that providing the model with the full context led to very long outputs that often repeated irrelevant information from the input context, instead of describing the image. We addressed the issue of long input prompts, by using a context window, as described in Section 4.3.1. This allows us to give the model additional context while mitigating possible errors in the generated training caption. We also experimented with different input prompts for InstructBLIP, which changed the quality of the outputs. The final prompt consists of the additional context window followed by *"Given the steps, give a short description of the image. Do NOT make assumptions, say only what you see in the image."* We generate two types of captions to train the Context Sequence Decoder: captions with a context window of 2 steps—short captions—and with a context window of 3 steps—long captions. Finally, we generate a long and a short caption for every image v_i in the dataset.

4.4.2 Dataset

We collected a dataset consisting of publicly available manual tasks in the recipes domain from [AllRecipes](#). We also considered DIY manual tasks from [WikiHow](#), in an out-of-domain evaluation. Each manual task has a title, a description, a list of ingredients/resources, and a sequence of step-by-step instructions, which may or may not be

Training Details	
Base Model	Alpaca-7B
Training Time	$\approx 10h$
Epochs	10
Loss Function	Cross-Entropy
Weight Decay	0.01
Model Max Length	400
Batch Size	2
Gradient Accumulation Steps	4
Effective Batch Size	8
Learning Rate	$1e^{-05}$
Learning Rate Scheduler	Cosine
Optimizer	AdamW
Adam β_1	0.900
Adam β_2	0.999
Adam ϵ	$1e^{-08}$
LoRA	
LoRA Rank	8
LoRA α	32
LoRA Dropout	0.1

Table 4.1: Training parameters for the best model.

illustrated. Since we want to illustrate the steps of a task, we focus on manual tasks which are fully illustrated, as we can use these images as ground-truth for training and evaluating our methods.

To focus on the task at hand, and reduce external problems, we refined the dataset. We filtered the recipes that had steps that were too long, and limited the number of steps in a recipe from 4 to 6 steps. We also removed any steps that did not contain actions which we could illustrate, such as steps merely saying “*Enjoy!*”. In total, we used 1100 recipes, with an average of 5.06 steps per recipe, resulting in 5562 individual steps.

4.4.3 Model Details

To train the sequence context decoder model, we chose to fine-tune an Alpaca-7B model, as it is an open-source instruction-tuned model, and there are implementations using LoRA [31], which reduces the computational cost during training. We experimented with different hyperparameters, namely different learning rates, learning rate schedulers, weight decay values, and AdamW optimizer hyperparameters, to find the most suitable ones for our problem. We point out that we used the loss of the models as the main criterion, as we do not have an automatic metric for evaluating model behaviour. The loss is not always indicative of the model’s performance in the task at hand, as we verified empirically. Finally, we fine-tuned Alpaca-7B for 10 epochs, on a single A100 40Gb GPU

with the final training details reported in Table 4.1. The dataset had a total of 5562 step-caption pairs, from which we used 80% for training and 180 examples for testing.

For generating all sequences of images, we used a frozen Stable Diffusion 2.1 [69] model.

4.4.4 Negative Prompts

As mentioned in Section 4.1, we found problems which were not related to the prompts or the concepts we were trying to illustrate, but general problems in image generation, i.e., tiled images, or deformed hands. In order to try to reduce some of these common problems, we used negative prompts. A negative prompt steers the generation away from the concepts present in it.

In the negative prompt, we included undesirable concepts such as *human* or *hands*, and also included some additional concepts, following common practices. Our final negative prompt was: "*hands, human, person, cropped, deformed, cut off, malformed, out of frame, split image, tiling, watermark, text.*"

4.4.5 Human Annotations

We evaluated our models through human annotation. For our comparisons, annotators inspected 30 sequences of images generated with different methods (withheld from annotators). For the comparison of different methods for maintaining visual coherence, and for tuning the threshold of our proposed method, annotators had to choose the 3 best sequences, out of a total of 5 sequences. Besides this annotation, we provided an additional *No good sequence* label, for when no sequence of images was of good quality. For finer-grain annotations, we conducted side-by-side comparisons to compare two sequence generation methods. In these cases, annotators were asked to choose the best method. In both cases, annotators could also provide feedback on the generated errors. To compare our method with ground-truth images, we asked annotators to rate each sequence on a 5-point Likert scale. For the evaluation of the Sequence Context Decoder, we showed example outputs of the model, next to the context, and asked the annotators to rate the output on a 5-point Likert scale. Section .4 in the Annex, shows examples of the annotation tasks and guidelines.

4.5 Results and Discussion

This section documents the results for the Sequence Context Decoder, followed by an analysis of our suggested visual conditioning method. Finally, we report the sequence illustration results obtained with the proposed framework composed of the Sequence Context Decoder and Sequence Conditioned Reverse Diffusion methods.

Sequence Context	Captions	Avg. Rating
$\{s_n\}$	<i>short</i>	3.00
$\{s_n\}$	<i>long</i>	3.23
$\{s_n, c_{n-1}\}$	<i>short</i>	3.68
$\{s_n, c_{n-1}\}$	<i>long</i>	<u>3.56</u>
$\{s_n, s_{n-1}, c_{n-2}\}$	<i>short</i>	3.41
$\{s_n, s_{n-1}, c_{n-2}\}$	<i>long</i>	3.35

Table 4.2: Sequence Context Decoder results for different context lengths, configurations and caption type.

Error type	%
Hallucinations	3.9%
Complex step with many actions	6.2%
Copied input	7.2%

Table 4.3: The contribution of each error type to the overall performance.

4.5.1 Sequence Context Decoder

We assessed the capacity of the Sequence Context Decoder of maintaining **semantic coherence** by manually annotating the decoder output for six settings with different context lengths and caption types, Table 4.2. We considered three context lengths: the shortest one uses only the current step, while the longest shows two steps and a caption. We used both *short* and *long* captions, as detailed in Section 4.4.1.

Table 4.2 shows the results for the different evaluation settings. These results indicate that the model attains the best semantic coherence with a context window of 2 and with short captions. We observed that captions generated with short contexts tend to lack some information, while captions generated with longer contexts introduced too much information, which was often noisy. This is aligned with a recent study [47] that highlights the fact that current large language models do not robustly make use of information in long input contexts. These results indicate that additional context needs to be carefully considered and curated, as the model is not able to filter out excess information. Another aspect to consider is the error introduced by annotators. When presented with longer contexts, annotators may expect that information to be present in the final output. This might lead to lower ratings when some information is missing, even if the output is correct.

We further analysed the errors of the best method and present the results in Table 4.3. Hallucinations occurred in 3.9% of the generations, and the LLM copied the input into the output in 7.2% of the cases. Finally, it is interesting to see that the input was too complex in 6.2% of the cases, i.e., describing more actions than what is possible to depict in the image.

Method	Best (%)	Second Best (%)	Third Best (%)
Random seed	17.70	41.20	13.00
Fixed seed	<u>29.40</u>	17.70	<u>33.30</u>
Latent 1	33.30	17.70	37.04
Latent 2	17.70	<u>23.50</u>	16.70
Img-to-Img	2.00	0.00	0.00

Table 4.4: Annotation results for the evaluation of the various methods of maintaining visual coherence. Annotators picked *No Good Sequence* in 18.99% of the sequences; we report the results for the remaining 81.01%.

4.5.2 Fixed Latent Vector Iteration

To better understand the strength of our **visual coherence** hypothesis, we conducted an experiment where all the images of the sequence are generated from the latent vectors of a fixed iteration from the previous task step, as described in Section 4.3.2.2.

Our empirical analysis of these generations, as seen in Figure 4.4, confirms our initial hypothesis, showing that using latent vector iterations from previous steps provides a good result in many cases. Additionally, this experiment evidences the strength of latent vector iterations as conditioning signals in a reverse diffusion process. This experiment shows that later iterations add a very strong bias to the generation, while earlier iterations, $k \in \{1, 2, 3\}$, still allow for the introduction of new aspects to the images. These results emphasise the importance of selecting the best conditioning seed from previous reverse diffusion iterations and processes.

4.5.3 Sequence Generation Results

We experimented with different methods for maintaining **visual coherence**. We used a **random seed** for all steps of the sequence, and a **fixed seed** for all steps of the sequence, as motivated by Section 4.3.2.1. We also used a **fixed latent vector iteration** from the previous step, represented by **Latent k** , where k is the fixed iteration, as described in Section 4.3.2.2. We chose $k \in \{1, 2\}$, according to the results discussed in Section 4.5.2. Finally, we also use our method, proposed in Section 4.3.2.3, that selects a latent vector iteration, z_i^j from the previous denoising steps as a starting seed.

4.5.3.1 Recipes Domain

To validate our initial hypothesis, we start by analysing its performance in the recipes domain. Our goal is to assess how conditioning the reverse diffusion process of each step affects the overall results. Table 4.4 provides the complete set of results across all competing methods. It is clear how using latent vector iterations and random seeds supports the intuition behind our method: a manual task is composed of continuous and independent actions, which should be conditioned by latent vector iterations and by

random seeds, respectively. Building on this observation, we calibrated the η parameter of the proposed method, using human evaluation. Table 4.5 shows the results for the calibration. We can see that there is a large margin in favour of $\eta = 0.50$; all following experiments use this value for the threshold.

η	Best (%)	Second Best (%)	Third Best (%)
0.70	19.20	12.80	14.90
0.65	12.80	14.90	25.50
0.60	19.20	23.40	21.30
0.55	14.90	23.40	21.30
0.50	34.04	25.53	17.02

Table 4.5: Annotation results for the sequences generated with different threshold values, η . Annotators picked *No Good Sequence* in 20.34% of the generations; we show results for the remaining 79.66%.

Method	Recipes (seen)	DIY (unseen)
Proposed method (wins)	46.67	30.00
Second best (wins)	26.67	23.33
Tie	10.00	16.67
No good sequence	16.67	30.00

Table 4.6: Annotation results of the comparison between our proposed method and the winning method from Table 4.4 (Latent 1).

To compare the proposed method to the best performing method in Table 4.4 (Latent 1), we asked annotators to select the best sequence out of two side-by-side sequences. Results reported in Table 4.6 show that in 46.7% of the cases, human annotators preferred our method over the alternative and in 10.0% of the cases the two methods were equally good. This confirms our hypothesis and supports the importance of selectively conditioning the denoising process on the previously generated steps of the sequence.

4.5.3.2 DIY Domain

Despite focusing on the recipes domain, we decided to assess the generalisation of the proposed method by evaluating its performance in an unseen domain: DIY tasks. With the results of our human annotation study, we observed that the transition from generating recipe images to DIY tasks has shown promising results. Results in Table 4.6 show that in 30.0% of the tasks, neither method produced satisfactory results. For the tasks that were correctly illustrated, we see that annotators preferred our method in 30.0% of the tasks, compared to 23.3% for the second-best approach. Additionally, 16.7% of comparisons resulted in a tie between the two methods. Although we see limitations in this domain, the

Method	Average Rating
Proposed method	2.93 ± 1.14
Ground-truth	4.58 ± 0.79

Table 4.7: Human annotation results for the comparison of the proposed method with ground-truth images.

results show that our approach is capable of generalizing to an unseen domain, producing satisfactory image sequences.

4.5.3.3 Generated vs Ground-Truth Sequences

We compared the quality of generated image sequences with the ground-truth sequences and asked human annotators to rate each sequence on a 5-point Likert scale. This is a particularly challenging setting because the real image sequences are photos taken by humans in a real-world setting, where sequence coherence is naturally captured. Table 4.7 shows that our method achieves over 60% of the ground-truth score, with the ground-truth sequences only 0.42 points below the maximum score.

4.5.4 Examples and Qualitative Analysis

In this section, we discuss some qualitative examples produced by the Sequence Conditioned Reverse Diffusion process and by the Sequence Context Decoder.

4.5.4.1 Sequence Conditioned Reverse Diffusion

To illustrate how different conditioning methods affect the quality of generated sequences, we present several examples in Figures 4.5 and 4.6, and in the Annex 5. In Figure 4.5, we can see that by using a fixed seed for all steps, we are able to preserve the background and add objects for each specific step. We can also see that the image-to-image method conditions the generations too strongly. Figure 4.5 also shows that our method is capable of preserving and recall key visual artefacts from several steps back in the sequence. We believe this is a distinctive and fundamental feature of the proposed method.

In the DIY out-of-domain experiment, Figure 4.6 shows a strong generalization to tasks involving simple object utilisation, such as using a broom for cleaning or a brush for painting. While fixed latent vector iteration methods show some memorization capability, the image-to-image generations are too biased on previous generations, and random seeds lead to very diverse generations. In this unseen domain, the proposed method encounters considerable challenges when tasked with more intricate activities, like performing a car’s oil change with its complex mechanical components, or engaging in tasks that involve philosophical or introspective elements. It is worth noting that these limitations are inherited from the core image generation method, which struggles to handle fine-details.

CHAPTER 4. TASK-GROUNDED IMAGE SEQUENCE SYNTHESIS



Figure 4.5: Examples of recipe illustrations with different methods for maintaining visual coherence. Each column illustrates steps 1 through 4.

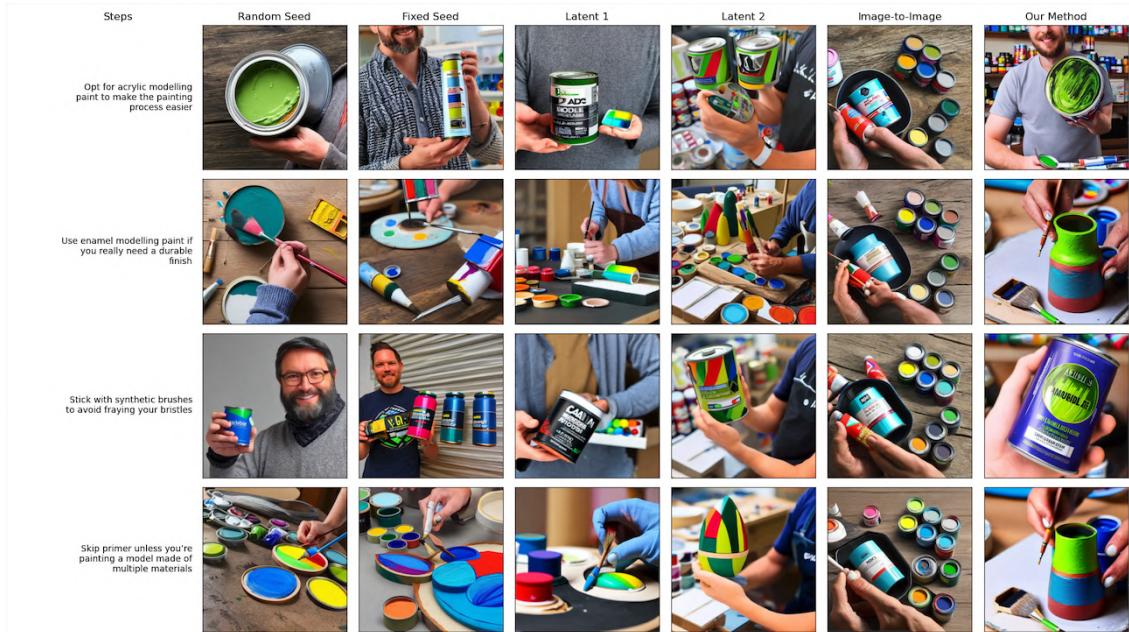


Figure 4.6: Examples of DIY illustrations with different methods for maintaining visual coherence. Each column illustrates steps 1 through 4.

4.5.4.2 Sequence Context Decoder

The examples in Table 4.8 highlight particular behaviours of the Sequence Context Decoder model. In Example 1, we can see the model correctly identifying the ingredients from the context, caption_{n-2} , going two steps back, and integrating them in the final output. It also recognizes the plate from step $_n$ as the object containing the ingredients. This shows the

4.5. RESULTS AND DISCUSSION

Example 1: The Sequence Context Decoder was able to dereference the objects from caption_{n-2} and create the correct caption for step_n.

Caption _{n-2}	A wooden cutting board with ingredients for a salad including avocado , crab meat , and grapes
Step _{n-1}	Dress all ingredients except the arugula with one fourth cup dressing, dress the arugula separately with two tablespoon dressing.
Step _n	Plate the arugula first. Combine the rest of the ingredients, and add on top.
Output	A plate of salad with avocado , crab meat , and grapes on top.

Example 2: The state of objects is not always preserved in the generated caption.

Caption _{n-1}	A purple plastic cup is sitting on a wooden table. Inside the cup, there is a mixture of black pepper, olive oil, brown sugar, curry powder, salt, and white vinegar. There are also some sultanas floating in the mixture. The cup is covered with plastic wrap.
Step _n	Get White Rice in a big bowl. Add in Celery and Scallion. In order to add a bit of color and some sweetness, chop up Red Apple and dust that up. Toss them together.
Output	A bowl of white rice is being prepared with celery, scallions, and red apples

Example 3: Some objects are correctly added to the generated caption.

Caption _{n-1}	In the image, there is a pot of soup on the stove with chicken and vegetables in it. There is also a person's hand reaching into the pot to add more ingredients to the soup.
Step _n	Serve with White Rice on the bottom and Lime Wedge on the side. Or with tortillas for dipping into the soup.
Output	In the image, there is a bowl of soup with a tortilla in it. The soup has chicken, vegetables, and lime juice in it.

Example 4: Sequence of actions was not fully captured by the model.

Step _{n-1}	Bring Water to boil in a medium saucepan.
Step _n	Once water has boiled, remove saucepan from heat and add the Black Tea. Steep for 15 minutes.
Output	a saucepan with black tea in it on a stove top

Example 5: When steps provide plenty of context, using a single step is sufficient.

Step _n	Halve fennel, remove stalk, and cut into fine strips. Finely dice onion. Peel garlic. Clean zucchini and bell pepper and cut into bite-sized pieces. Combine vegetables with rosemary, thyme, and some of the olive oil in a plastic bag. Close, shake, and marinate for approx. 20 min.
Output	A woman is cutting up vegetables on a cutting board with a knife and a peeler.

Table 4.8: Qualitative analysis of the Sequence Context Decoder results.

potential in giving the model additional context to generate richer prompts. In Example 2, we can see that, despite being able to maintain the *red apples*, the model makes no explicit reference to their state: *chopped up*. This is still a limitation, which may lead to a wrongful representation of intact apples. In Example 3, we want to highlight two main aspects of the generation: we can see the model adding the *bowl of soup* from the context to the prompt, maintaining semantic coherence. We can also see that the model kept *lime juice*. This is correct, from the point of view of the task at hand, but shows the lack of understanding of what can be perceived in an image. We reason that this knowledge should come from the pretraining of the model, and not from our fine-tuning to this task. Example 4 shows an example of a depiction that is mostly correct, but misses a step of the sequence. The representation of the saucepan with black tea in it is plausible, but step_n indicates the saucepan should be removed from the heat. Finally, in Example 5, we see a very long step, with various actions. In this case, we consider it plausible for the model to pick one of these actions. This is a better result than attempting to represent them all, which would lead to an inadequate prompt. Despite this, this specific generation lacks some context, as the word *vegetables* is generic; it is important for the generated prompts to be specific, containing the ingredients mentioned in the context.

4.6 Conclusions

Generating visual instructions to illustrate complex manual tasks is a challenging problem, requiring a method that understands the sequence of actions and can generate a sequence of images with visual and semantic coherence. In this chapter, we addressed the problem of illustrating complex manual tasks and proposed a framework for generating a sequence of images that illustrates the manual task. The framework is composed of a novel **Sequence Context Decoder** that preserves the **semantic coherence** of a sequence of actions by transforming it into a visual caption. The full sequence illustration framework is completed by a **Sequence Conditioned Reverse Diffusion** process that uses a latent vector iteration from a past image generation process to maintain **visual coherence**. Experiments in the target domain demonstrated the strong performance of the framework in generating coherent sequences of visual instructions of manual tasks. The **generalization to unseen domains** was successfully validated in the DIY domain with positive results.

CONCLUSIONS

In the present chapter, we present the main achievements of this dissertation and discuss existing limitations. We then present our scientific contributions and, finally, analyse the next steps to take for improving the proposed solutions.

5.1 Achievements and Limitations

During this thesis, I was able to work with some of the most recently released Language and Vision-and-Language models, that define the current state-of-the-art in AI. I applied them to a novel problem as I studied it and its difficulties. Throughout this process, I understood the limitations and strengths of Image Synthesis methods, Language Models, and my proposed solutions. I identified common problems with Image Synthesis and proposed methods of mitigating these problems. I investigated approaches to improve the generation of image sequences without the computational cost of training the diffusion models, relying on Natural Language Models. This meant understanding how well language models can reason about visual aspects.

The approach is still limited by practical considerations, such as very long tasks, or determining dependencies between task steps, but it is a step towards a lightweight solution to the problem. Another limitation is the lack of an automatic method for evaluation. Most of the evaluation required human annotations, which can be costly and time-consuming. The main difficulty lies in the fact that the task is hard to define, with some aspects being subjective.

I also explored lightweight solutions for Video Moment Retrieval, which meant understanding how video data can be processed, what information we can create about this data, and how we can integrate it in a retrieval system. I also had to consider the implications of having a solution that must work in a real-world live system with latency requirements. It would be worth studying how the proposed solution could be improved with new ways to preprocess, represent, and retrieve the data.

5.2 Contributions

The Video Moment Retrieval framework implemented provides a lightweight approach to the problem, with the computational cost being moved to the processing stage, offline. This resulted in a practical solution that was successfully integrated into a live system, with acceptable latency, and strong performance. The navigation through captions approach proved to be robust and work well with real users.

As I studied Image Synthesis models and their applications to real-world manuals tasks, I contributed to a scientific paper, currently submitted to a conference and awaiting review, where we study how to capture the visual elements of a naturally occurring text to create better image generation prompts. This paper can be found in Annex [2](#).

Previous sequence generation methods focused chiefly on two synthetic cartoon datasets. I applied my method to a real-world setting, illustrating real recipes and tasks. This required working with real-world textual data, which was not prepared for image generation, introducing additional challenges. The results showed that it is possible to address the problem through natural language, and some modifications in the image generation algorithm, without training it. My proposed method proved effective in illustrating real-world manuals tasks, and resulted in a scientific paper, currently submitted to a conference and awaiting review, see Annex [1](#).

These methods were applied to a real-world live system, TWIZ [45], enhancing it with rich visual content. This assistant placed first in the Alexa Prize TaskBot Challenge 2, showing consistently strong performance with real users. The technical report for the assistant can be found in Annex [3](#).

5.3 Critical Analysis and Future Work

Regarding Image Sequence Synthesis, the main problem still present is preserving the appropriate context over consecutive generations and forgetting the unimportant aspects. In the future, we suggest that approaches explore fine-tuning the image synthesis model, instead of attacking the problem solely through prompting and using the latent variables in a zero-shot setting. We still think that it is important to manipulate the prompts, as this seems to be a promising direction and can mitigate problems present in image synthesis algorithms. Most natural descriptions will contain aspects which are not representable in an image, manipulating the prompts can help reduce these and produce more visually rich natural language. It would be interesting to see how scaling the language model can help with this task. We believe that a combination between prompt rewriting and approaches like those presented in Section [2.8](#) can push the boundaries of what is possible in coherent image sequence synthesis. Another aspect to explore is connecting the LLM directly to the image generation model and training them together, feeding the LLM embeddings as directly as possible to the generation models, instead of the inference results in natural language, which may lose some information. A key concern that we think is worth

--- 5.3. CRITICAL ANALYSIS AND FUTURE WORK

investigating is how to evaluate the generated image sequences. An automatic metric would be extremely helpful during the development process, and would serve as a means for future researchers to evaluate their approaches. The task of generating coherent image sequences is one showing promising results and various research directions. Despite its application to real-world problems, it is still underexplored, and we urge other researchers to study it deeply and propose innovative solutions.

BIBLIOGRAPHY

- [1] URL: <https://openai.com/blog/openai-api> (cit. on p. 22).
- [2] URL: <https://openai.com/chatgpt> (cit. on pp. 22, 33).
- [3] URL: <https://opensearch.org/> (cit. on p. 47).
- [4] A. Agrawal et al. *VQA: Visual Question Answering*. 2015. doi: [10.48550/ARXIV.1505.00468](https://doi.org/10.48550/ARXIV.1505.00468). URL: <https://arxiv.org/abs/1505.00468> (cit. on p. 2).
- [5] S. S. Baraheem, T.-N. Le, and T. V. Nguyen. "Image synthesis: a review of methods, datasets, evaluation metrics, and future outlook". In: *Artificial Intelligence Review* 56.10 (2023-02), pp. 10813–10865. doi: [10.1007/s10462-023-10434-2](https://doi.org/10.1007/s10462-023-10434-2). URL: <https://doi.org/10.1007/s10462-023-10434-2> (cit. on p. 37).
- [6] L. Beyer. *[Public, Approved] Intro to Transformers*. URL: https://docs.google.com/presentation/d/1ZXF1hYczos679r70Yu8vV9u06B1J0ztzeDxrnBxD1S0/edit#slide=id.g13dd67c5ab8_0_3666 (cit. on p. 13).
- [7] T. B. Brown et al. *Language Models are Few-Shot Learners*. 2020. doi: [10.48550/ARXIV.2005.14165](https://doi.org/10.48550/ARXIV.2005.14165). URL: <https://arxiv.org/abs/2005.14165> (cit. on pp. 13–15, 18–21).
- [8] E. Bugliarello et al. *Multimodal Pretraining Unmasked: A Meta-Analysis and a Unified Framework of Vision-and-Language BERTs*. 2020. doi: [10.48550/ARXIV.2011.15124](https://doi.org/10.48550/ARXIV.2011.15124). URL: <https://arxiv.org/abs/2011.15124> (cit. on pp. 25, 26).
- [9] S. Changpinyo et al. *All You May Need for VQA are Image Captions*. 2022. doi: [10.48550/ARXIV.2205.01883](https://doi.org/10.48550/ARXIV.2205.01883). URL: <https://arxiv.org/abs/2205.01883> (cit. on pp. 144, 145).
- [10] X. Chen et al. *PaLI: A Jointly-Scaled Multilingual Language-Image Model*. 2022. doi: [10.48550/ARXIV.2209.06794](https://doi.org/10.48550/ARXIV.2209.06794). URL: <https://arxiv.org/abs/2209.06794> (cit. on pp. 14, 15, 23–25).
- [11] W.-L. Chiang et al. *Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality*. 2023-03. URL: <https://lmsys.org/blog/2023-03-30-vicuna/> (cit. on pp. 22, 33, 34).

- [12] A. Chowdhery et al. *PaLM: Scaling Language Modeling with Pathways*. 2022. arXiv: [2204.02311 \[cs.CL\]](https://arxiv.org/abs/2204.02311) (cit. on pp. 13–15, 18, 19, 21).
- [13] H. W. Chung et al. “Scaling Instruction-Finetuned Language Models”. In: *CoRR* abs/2210.11416 (2022). doi: [10.48550/arXiv.2210.11416](https://doi.org/10.48550/arXiv.2210.11416). arXiv: [2210.11416](https://arxiv.org/abs/2210.11416). URL: <https://doi.org/10.48550/arXiv.2210.11416> (cit. on pp. 21, 22, 34).
- [14] W. Dai et al. “InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning”. In: *CoRR* abs/2305.06500 (2023). doi: [10.48550/arXiv.2305.06500](https://doi.org/10.48550/arXiv.2305.06500). arXiv: [2305.06500](https://arxiv.org/abs/2305.06500). URL: <https://doi.org/10.48550/arXiv.2305.06500> (cit. on pp. 19, 34, 45, 46, 52, 55).
- [15] A. Das et al. *Visual Dialog*. 2016. doi: [10.48550/ARXIV.1611.08669](https://doi.org/10.48550/ARXIV.1611.08669). URL: <https://arxiv.org/abs/1611.08669> (cit. on p. 2).
- [16] J. Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2018. doi: [10.48550/ARXIV.1810.04805](https://doi.org/10.48550/ARXIV.1810.04805). URL: <https://arxiv.org/abs/1810.04805> (cit. on pp. 10, 12–18, 29, 31).
- [17] S. F. Dierk. “The SMART retrieval system: Experiments in automatic document processing — Gerard Salton, Ed. (Englewood Cliffs, N.J.: Prentice-Hall, 1971, 556 pp., \$15.00)”. In: *IEEE Transactions on Professional Communication* PC-15.1 (1972), pp. 17–17. doi: [10.1109/TPC.1972.6591971](https://doi.org/10.1109/TPC.1972.6591971) (cit. on p. 8).
- [18] L. Donatelli et al. “Aligning Actions Across Recipe Graphs”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Ed. by M.-F. Moens et al. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021-11, pp. 6930–6942. doi: [10.18653/v1/2021.emnlp-main.554](https://doi.org/10.18653/v1/2021.emnlp-main.554). URL: <https://aclanthology.org/2021.emnlp-main.554> (cit. on p. 50).
- [19] A. Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2020. doi: [10.48550/ARXIV.2010.11929](https://doi.org/10.48550/ARXIV.2010.11929). URL: <https://arxiv.org/abs/2010.11929> (cit. on pp. 10, 14, 23, 30, 31, 33, 34).
- [20] C. Du et al. “Stable Diffusion is Unstable”. In: *CoRR* abs/2306.02583 (2023). doi: [10.48550/arXiv.2306.02583](https://doi.org/10.48550/arXiv.2306.02583). arXiv: [2306.02583](https://arxiv.org/abs/2306.02583). URL: <https://doi.org/10.48550/arXiv.2306.02583> (cit. on p. 2).
- [21] Y. Du et al. *A Survey of Vision-Language Pre-Trained Models*. 2022. doi: [10.48550/ARXIV.2202.10936](https://doi.org/10.48550/ARXIV.2202.10936). URL: <https://arxiv.org/abs/2202.10936> (cit. on pp. 13, 25, 26).
- [22] W. Feng et al. “Training-Free Structured Diffusion Guidance for Compositional Text-to-Image Synthesis”. In: *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL: <https://openreview.net/pdf?id=PUIqjT4rzq7> (cit. on p. 2).

BIBLIOGRAPHY

- [23] Z. Feng et al. "Improved Visual Story Generation with Adaptive Context Modeling". In: *Findings of the Association for Computational Linguistics: ACL 2023*. Toronto, Canada: Association for Computational Linguistics, 2023-07, pp. 4939–4955. doi: [10.18653/v1/2023.findings-acl.305](https://doi.org/10.18653/v1/2023.findings-acl.305). URL: <https://aclanthology.org/2023.findings-acl.305> (cit. on pp. 41, 42, 50).
- [24] R. Ferreira et al. "TWIZ: The Multimodal Conversational Task Wizard". In: *MM '22: The 30th ACM International Conference on Multimedia, Lisboa, Portugal, October 10 - 14, 2022*. Ed. by J. Magalhães et al. ACM, 2022, pp. 6997–6999. doi: [10.1145/3503161.3547741](https://doi.org/10.1145/3503161.3547741). URL: <https://doi.org/10.1145/3503161.3547741> (cit. on p. 1).
- [25] F. Gilardi, M. Alizadeh, and M. Kubli. "ChatGPT outperforms crowd workers for text-annotation tasks". In: *Proceedings of the National Academy of Sciences* 120.30 (2023-07). doi: [10.1073/pnas.2305016120](https://doi.org/10.1073/pnas.2305016120). URL: <https://doi.org/10.1073/pnas.2305016120> (cit. on p. 33).
- [26] Y. Goyal et al. *Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering*. 2016. doi: [10.48550/ARXIV.1612.00837](https://arxiv.org/abs/1612.00837). URL: <https://arxiv.org/abs/1612.00837> (cit. on p. 27).
- [27] K. Gregor et al. *DRAW: A Recurrent Neural Network For Image Generation*. 2015. arXiv: [1502.04623 \[cs.CV\]](https://arxiv.org/abs/1502.04623) (cit. on p. 37).
- [28] T. Gupta et al. *Imagine This! Scripts to Compositions to Videos*. 2018. arXiv: [1804.03608 \[cs.CV\]](https://arxiv.org/abs/1804.03608) (cit. on pp. 40–42, 50).
- [29] L. A. Hendricks et al. *Localizing Moments in Video with Natural Language*. 2017. arXiv: [1708.01641 \[cs.CV\]](https://arxiv.org/abs/1708.01641) (cit. on p. 40).
- [30] J. Ho, A. Jain, and P. Abbeel. *Denoising Diffusion Probabilistic Models*. 2020. arXiv: [2006.11239 \[cs.LG\]](https://arxiv.org/abs/2006.11239) (cit. on pp. 37, 38).
- [31] E. J. Hu et al. "LoRA: Low-Rank Adaptation of Large Language Models". In: *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL: <https://openreview.net/forum?id=nZeVKeeFYf9> (cit. on p. 56).
- [32] D. Jurafsky and J. H. Martin. "N-GRAM LANGUAGE MODELS". In: *Speech and language processing*. Pearson, 2014. URL: <https://web.stanford.edu/~jurafsky/slp3/3.pdf> (cit. on p. 13).
- [33] D. Jurafsky and J. H. Martin. "Transfer Learning with Pretrained Language Models and Contextual Embeddings". In: *Speech and language processing*. Pearson, 2014. URL: <https://web.stanford.edu/~jurafsky/slp3/11.pdf> (cit. on p. 12).
- [34] D. Jurafsky and J. H. Martin. "Vector Semantics and Embeddings". In: *Speech and language processing*. Pearson, 2014. URL: <https://web.stanford.edu/~jurafsky/slp3/6.pdf> (cit. on pp. 8, 9).

- [35] K.-M. Kim et al. *DeepStory: Video Story QA by Deep Embedded Memory Networks*. 2017. arXiv: [1707.00836 \[cs.CV\]](https://arxiv.org/abs/1707.00836) (cit. on pp. 40–42, 50).
- [36] W. Kim, B. Son, and I. Kim. *ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision*. 2021. doi: [10.48550/ARXIV.2102.03334](https://doi.org/10.48550/ARXIV.2102.03334). URL: <https://arxiv.org/abs/2102.03334> (cit. on pp. 15, 25–28).
- [37] R. Krishna et al. *Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations*. 2016. doi: [10.48550/ARXIV.1602.07332](https://doi.org/10.48550/ARXIV.1602.07332). URL: <https://arxiv.org/abs/1602.07332> (cit. on p. 142).
- [38] J. Lei, T. L. Berg, and M. Bansal. “QVHighlights: Detecting Moments and Highlights in Videos via Natural Language Queries”. In: *CoRR* abs/2107.09609 (2021). arXiv: [2107.09609](https://arxiv.org/abs/2107.09609). URL: <https://arxiv.org/abs/2107.09609> (cit. on p. 36).
- [39] M. Lewis et al. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*. 2019. doi: [10.48550/ARXIV.1910.13461](https://doi.org/10.48550/ARXIV.1910.13461). URL: <https://arxiv.org/abs/1910.13461> (cit. on pp. 13–15, 18, 19).
- [40] J. Li et al. “BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models”. In: *International Conference on Machine Learning, ICML 2023, 23–29 July 2023, Honolulu, Hawaii, USA*. Ed. by A. Krause et al. Vol. 202. Proceedings of Machine Learning Research. PMLR, 2023, pp. 19730–19742. URL: <https://proceedings.mlr.press/v202/li23q.html> (cit. on p. 32).
- [41] J. Li et al. “BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation”. In: *International Conference on Machine Learning, ICML 2022, 17–23 July 2022, Baltimore, Maryland, USA*. Ed. by K. Chaudhuri et al. Vol. 162. Proceedings of Machine Learning Research. PMLR, 2022, pp. 12888–12900. URL: <https://proceedings.mlr.press/v162/li22n.html> (cit. on pp. 31, 41).
- [42] Z.-N. Li, M. S. Drew, and J. Liu. *Fundamentals of Multimedia*. 2nd. Springer Publishing Company, Incorporated, 2014. ISBN: 3319052896 (cit. on p. 46).
- [43] Y. Li et al. *StoryGAN: A Sequential Conditional GAN for Story Visualization*. 2019. arXiv: [1812.02784 \[cs.CV\]](https://arxiv.org/abs/1812.02784) (cit. on pp. 39–42).
- [44] T.-Y. Lin et al. *Microsoft COCO: Common Objects in Context*. 2014. doi: [10.48550/ARXIV.1405.0312](https://doi.org/10.48550/ARXIV.1405.0312). URL: <https://arxiv.org/abs/1405.0312> (cit. on pp. 37, 142).
- [45] N. U. Lisbon. “TWIZ: A conversational Task Wizard with multimodal curiosity-exploration”. In: *Alexa Prize TaskBot Challenge 1 Proceedings*. 2022. URL: <https://www.amazon.science/alexaprize/proceedings/twiz-a-conversational-task-wizard-with-multimodal-curiosity-exploration> (cit. on pp. 1, 3, 66).
- [46] H. Liu et al. “Visual Instruction Tuning”. In: *CoRR* abs/2304.08485 (2023). doi: [10.48550/arXiv.2304.08485](https://doi.org/10.48550/arXiv.2304.08485). arXiv: [2304.08485](https://arxiv.org/abs/2304.08485). URL: <https://doi.org/10.48550/arXiv.2304.08485> (cit. on pp. 1, 33).

BIBLIOGRAPHY

- [47] N. F. Liu et al. *Lost in the Middle: How Language Models Use Long Contexts*. 2023. arXiv: [2307.03172 \[cs.CL\]](https://arxiv.org/abs/2307.03172) (cit. on p. 58).
- [48] Y. Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. doi: [10.48550/ARXIV.1907.11692](https://doi.org/10.48550/ARXIV.1907.11692). URL: <https://arxiv.org/abs/1907.11692> (cit. on pp. 14, 15).
- [49] Z. Liu et al. *A ConvNet for the 2020s*. 2022. doi: [10.48550/ARXIV.2201.03545](https://doi.org/10.48550/ARXIV.2201.03545). URL: <https://arxiv.org/abs/2201.03545> (cit. on p. 22).
- [50] J. M. Lourenço. *The NOVAthesis L^AT_EX Template User's Manual*. NOVA University Lisbon. 2021. URL: <https://github.com/joaomlourenco/novathesis/raw/master/template.pdf> (cit. on p. ii).
- [51] P. Lu et al. "Learn to Explain: Multimodal Reasoning via Thought Chains for Science Question Answering". In: *The 36th Conference on Neural Information Processing Systems (NeurIPS)*. 2022 (cit. on p. 33).
- [52] A. Maharana, D. Hannan, and M. Bansal. *StoryDALL-E: Adapting Pretrained Text-to-Image Transformers for Story Continuation*. 2022. arXiv: [2209.06192 \[cs.CV\]](https://arxiv.org/abs/2209.06192) (cit. on pp. 37, 39, 42).
- [53] E. Mansimov et al. *Generating Images from Captions with Attention*. 2016. arXiv: [1511.02793 \[cs.LG\]](https://arxiv.org/abs/1511.02793) (cit. on p. 37).
- [54] J. Mao, X. Wang, and K. Aizawa. "Guided Image Synthesis via Initial Image Editing in Diffusion Model". In: *Proceedings of the 31st ACM International Conference on Multimedia, MM 2023, Ottawa, ON, Canada, 29 October 2023- 3 November 2023*. Ed. by A. El-Saddik et al. ACM, 2023, pp. 5321–5329. doi: [10.1145/3581783.3612191](https://doi.org/10.1145/3581783.3612191). URL: <https://doi.org/10.1145/3581783.3612191> (cit. on p. 54).
- [55] J. A. Mencia et al. "ADELA: a conversational virtual assistant to prevent delirium in hospitalized older persons". In: *J. Supercomput.* 79.15 (2023), pp. 17670–17690. doi: [10.1007/s11227-023-05352-7](https://doi.org/10.1007/s11227-023-05352-7). URL: <https://doi.org/10.1007/s11227-023-05352-7> (cit. on p. 1).
- [56] C. Meng et al. "SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations". In: *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL: https://openreview.net/forum?id=aBsCjcPu%5C_tE (cit. on pp. 37, 53).
- [57] A. Miech et al. "HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips". In: *ICCV*. 2019 (cit. on pp. 35, 143, 144).
- [58] T. Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. 2013. doi: [10.48550/ARXIV.1301.3781](https://doi.org/10.48550/ARXIV.1301.3781). URL: <https://arxiv.org/abs/1301.3781> (cit. on p. 8).

- [59] B. Min et al. *Recent Advances in Natural Language Processing via Large Pre-Trained Language Models: A Survey*. 2021. doi: [10.48550/ARXIV.2111.01243](https://doi.org/10.48550/ARXIV.2111.01243). URL: <https://arxiv.org/abs/2111.01243> (cit. on p. 13).
- [60] OpenAI. “GPT-4 Technical Report”. In: *CoRR* abs/2303.08774 (2023). doi: [10.48550/arXiv.2303.08774](https://doi.org/10.48550/arXiv.2303.08774). arXiv: [2303.08774](https://arxiv.org/abs/2303.08774). URL: <https://doi.org/10.48550/arXiv.2303.08774> (cit. on pp. 1, 33).
- [61] L. Ouyang et al. “Training language models to follow instructions with human feedback”. In: *NeurIPS*. 2022. URL: http://papers.nips.cc/paper%5C_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html (cit. on pp. 1, 20).
- [62] X. Pan et al. *Synthesizing Coherent Story with Auto-Regressive Latent Diffusion Models*. 2022. arXiv: [2211.10950 \[cs.CV\]](https://arxiv.org/abs/2211.10950) (cit. on pp. 37, 39–42, 50).
- [63] A. Radford et al. “Improving language understanding by generative pre-training”. In: (2018) (cit. on pp. 10, 13–15, 17, 18).
- [64] A. Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. doi: [10.48550/ARXIV.2103.00020](https://doi.org/10.48550/ARXIV.2103.00020). URL: <https://arxiv.org/abs/2103.00020> (cit. on pp. 22, 24, 26, 29–33, 41, 45, 46, 54, 142, 143).
- [65] C. Raffel et al. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. 2019. doi: [10.48550/ARXIV.1910.10683](https://doi.org/10.48550/ARXIV.1910.10683). URL: <https://arxiv.org/abs/1910.10683> (cit. on pp. 13–15, 21).
- [66] T. Rahman et al. “Make-A-Story: Visual Memory Conditioned Consistent Story Generation”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*. IEEE, 2023, pp. 2493–2502. doi: [10.1109/CVPR52729.2023.00246](https://doi.org/10.1109/CVPR52729.2023.00246). URL: <https://doi.org/10.1109/CVPR52729.2023.00246> (cit. on p. 50).
- [67] A. Ramesh et al. *Zero-Shot Text-to-Image Generation*. 2021. arXiv: [2102.12092 \[cs.CV\]](https://arxiv.org/abs/2102.12092) (cit. on pp. 37, 38, 40).
- [68] S. Reed et al. *Generative Adversarial Text to Image Synthesis*. 2016. arXiv: [1605.05396 \[cs.NE\]](https://arxiv.org/abs/1605.05396) (cit. on p. 37).
- [69] R. Rombach et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2022. arXiv: [2112.10752 \[cs.CV\]](https://arxiv.org/abs/2112.10752) (cit. on pp. 37, 38, 41, 57).
- [70] O. Ronneberger, P. Fischer, and T. Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*. Ed. by N. Navab et al. Vol. 9351. Lecture Notes in Computer Science. Springer, 2015, pp. 234–241. doi: [10.1007/978-3-319-24574-4_28](https://doi.org/10.1007/978-3-319-24574-4_28). URL: https://doi.org/10.1007/978-3-319-24574-4%5C_28 (cit. on p. 38).

BIBLIOGRAPHY

- [71] A. Singh et al. *FLAVA: A Foundational Language And Vision Alignment Model*. 2021. doi: [10.48550/ARXIV.2112.04482](https://doi.org/10.48550/ARXIV.2112.04482). URL: <https://arxiv.org/abs/2112.04482> (cit. on pp. 15, 23, 24, 26, 28, 29, 143).
- [72] S. Soltan et al. *AlexaTM 20B: Few-Shot Learning Using a Large-Scale Multilingual Seq2Seq Model*. 2022. doi: [10.48550/ARXIV.2208.01448](https://doi.org/10.48550/ARXIV.2208.01448). URL: <https://arxiv.org/abs/2208.01448> (cit. on p. 14).
- [73] H. Song et al. *CLIP Models are Few-shot Learners: Empirical Studies on VQA and Visual Entailment*. 2022. doi: [10.48550/ARXIV.2203.07190](https://doi.org/10.48550/ARXIV.2203.07190). URL: <https://arxiv.org/abs/2203.07190> (cit. on p. 24).
- [74] K. SPARCK JONES. *A STATISTICAL INTERPRETATION OF TERM SPECIFICITY AND ITS APPLICATION IN RETRIEVAL*. en. 1972-01. doi: [10.1108/eb026526](https://doi.org/10.1108/eb026526). URL: <http://dx.doi.org/10.1108/eb026526> (cit. on p. 8).
- [75] K. Sparck Jones. *Synonymy and Semantic Classification*. GBR: Edinburgh University Press, 1986. ISBN: 0852245173 (cit. on p. 8).
- [76] R. Tang et al. “What the DAAM: Interpreting Stable Diffusion Using Cross Attention”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2023, Toronto, Canada, July 9-14, 2023. Ed. by A. Rogers, J. L. Boyd-Graber, and N. Okazaki. Association for Computational Linguistics, 2023, pp. 5644–5659. doi: [10.18653/v1/2023.acl-long.310](https://doi.org/10.18653/v1/2023.acl-long.310). URL: <https://doi.org/10.18653/v1/2023.acl-long.310> (cit. on p. 2).
- [77] M. Tao et al. *DF-GAN: A Simple and Effective Baseline for Text-to-Image Synthesis*. 2022. arXiv: [2008.05865 \[cs.CV\]](https://arxiv.org/abs/2008.05865) (cit. on p. 37).
- [78] R. Taori et al. *Stanford Alpaca: An Instruction-following LLaMA model*. https://github.com/tatsu-lab/stanford_alpaca. 2023 (cit. on pp. 20, 22).
- [79] Y. Tay et al. *UL2: Unifying Language Learning Paradigms*. 2022. doi: [10.48550/ARXIV.2205.05131](https://doi.org/10.48550/ARXIV.2205.05131). URL: <https://arxiv.org/abs/2205.05131> (cit. on pp. 13–15).
- [80] B. Thomee et al. “YFCC100M”. In: *Communications of the ACM* 59.2 (2016-01), pp. 64–73. doi: [10.1145/2812802](https://doi.org/10.1145/2812802). URL: <https://doi.org/10.1145%2F2812802> (cit. on pp. 37, 142).
- [81] R. Thoppilan et al. “LaMDA: Language Models for Dialog Applications”. In: *CoRR* abs/2201.08239 (2022). arXiv: [2201.08239](https://arxiv.org/abs/2201.08239). URL: <https://arxiv.org/abs/2201.08239> (cit. on p. 21).
- [82] Ting-Hao et al. *Visual Storytelling*. 2016. arXiv: [1604.03968 \[cs.CL\]](https://arxiv.org/abs/1604.03968) (cit. on p. 41).
- [83] H. Touvron et al. “LLaMA: Open and Efficient Foundation Language Models”. In: *CoRR* abs/2302.13971 (2023). doi: [10.48550/arXiv.2302.13971](https://doi.org/10.48550/arXiv.2302.13971). arXiv: [2302.13971](https://arxiv.org/abs/2302.13971). URL: <https://doi.org/10.48550/arXiv.2302.13971> (cit. on pp. 18, 22, 33).

- [84] A. Vaswani et al. *Attention Is All You Need*. 2017. doi: [10.48550/ARXIV.1706.03762](https://doi.org/10.48550/ARXIV.1706.03762). URL: <https://arxiv.org/abs/1706.03762> (cit. on pp. 7–13, 15, 16, 18, 22, 37).
- [85] A. Wang et al. “GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding”. In: In the Proceedings of ICLR. 2019 (cit. on p. 29).
- [86] P. Wang et al. “OFA: Unifying Architectures, Tasks, and Modalities Through a Simple Sequence-to-Sequence Learning Framework”. In: *CoRR* abs/2202.03052 (2022) (cit. on pp. 23–26).
- [87] T. Wang et al. *What Language Model Architecture and Pretraining Objective Work Best for Zero-Shot Generalization?* 2022. doi: [10.48550/ARXIV.2204.05832](https://doi.org/10.48550/ARXIV.2204.05832). URL: <https://arxiv.org/abs/2204.05832> (cit. on p. 13).
- [88] Y. Wang et al. “Self-Instruct: Aligning Language Models with Self-Generated Instructions”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2023, Toronto, Canada, July 9–14, 2023. Ed. by A. Rogers, J. L. Boyd-Graber, and N. Okazaki. Association for Computational Linguistics, 2023, pp. 13484–13508. doi: [10.18653/v1/2023.acl-long.754](https://doi.org/10.18653/v1/2023.acl-long.754). URL: <https://doi.org/10.18653/v1/2023.acl-long.754> (cit. on pp. 19, 20, 22).
- [89] Z. Wang et al. *SimVLM: Simple Visual Language Model Pretraining with Weak Supervision*. 2021. doi: [10.48550/ARXIV.2108.10904](https://doi.org/10.48550/ARXIV.2108.10904). URL: <https://arxiv.org/abs/2108.10904> (cit. on p. 26).
- [90] J. Wei et al. “Finetuned Language Models are Zero-Shot Learners”. In: *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25–29, 2022*. OpenReview.net, 2022. URL: <https://openreview.net/forum?id=gEZrGCozdqR> (cit. on pp. 19, 21, 22, 33).
- [91] Q. Wu et al. “Classical Visual Question Answering”. In: *Visual Question Answering: From Theory to Application*. Singapore: Springer Nature Singapore, 2022, pp. 35–72. ISBN: 978-981-19-0964-1. doi: [10.1007/978-981-19-0964-1_4](https://doi.org/10.1007/978-981-19-0964-1_4). URL: https://doi.org/10.1007/978-981-19-0964-1_4 (cit. on p. 13).
- [92] Q. Wu et al. “Deep Learning Basics”. In: *Visual Question Answering: From Theory to Application*. Singapore: Springer Nature Singapore, 2022, pp. 15–26. ISBN: 978-981-19-0964-1. doi: [10.1007/978-981-19-0964-1_2](https://doi.org/10.1007/978-981-19-0964-1_2). URL: https://doi.org/10.1007/978-981-19-0964-1_2 (cit. on p. 13).
- [93] Q. Wu et al. “Vision-and-Language Pretraining for VQA”. In: *Visual Question Answering: From Theory to Application*. Singapore: Springer Nature Singapore, 2022, pp. 91–107. ISBN: 978-981-19-0964-1. doi: [10.1007/978-981-19-0964-1_6](https://doi.org/10.1007/978-981-19-0964-1_6). URL: https://doi.org/10.1007/978-981-19-0964-1_6 (cit. on p. 17).

BIBLIOGRAPHY

- [94] Q. Wu et al. “Visual Dialogue”. In: *Visual Question Answering: From Theory to Application*. Singapore: Springer Nature Singapore, 2022, pp. 199–218. ISBN: 978-981-19-0964-1. DOI: [10.1007/978-981-19-0964-1_14](https://doi.org/10.1007/978-981-19-0964-1_14). URL: https://doi.org/10.1007/978-981-19-0964-1_14 (cit. on p. 2).
- [95] Y. Wu et al. *Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. 2016. DOI: [10.48550/ARXIV.1609.08144](https://arxiv.org/abs/1609.08144). URL: <https://arxiv.org/abs/1609.08144> (cit. on p. 16).
- [96] L. Yang et al. *Diffusion Models: A Comprehensive Survey of Methods and Applications*. 2023. arXiv: [2209.00796 \[cs.LG\]](https://arxiv.org/abs/2209.00796) (cit. on p. 38).
- [97] S. Zhang et al. “OPT: Open Pre-trained Transformer Language Models”. In: *CoRR* abs/2205.01068 (2022). DOI: [10.48550/arXiv.2205.01068](https://doi.org/10.48550/arXiv.2205.01068). arXiv: [2205.01068](https://arxiv.org/abs/2205.01068). URL: <https://doi.org/10.48550/arXiv.2205.01068> (cit. on p. 33).
- [98] Y. Zhong et al. *Video Question Answering: Datasets, Algorithms and Challenges*. 2022. DOI: [10.48550/ARXIV.2203.01225](https://arxiv.org/abs/2203.01225). URL: <https://arxiv.org/abs/2203.01225> (cit. on pp. 2, 34–36).
- [99] Y. Zhu et al. *Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books*. 2015. DOI: [10.48550/ARXIV.1506.06724](https://arxiv.org/abs/1506.06724). URL: <https://arxiv.org/abs/1506.06724> (cit. on pp. 16, 17).

.1 Generating Consistent Image Sequences for Real-World Manual Tasks

Generating Coherent Sequences of Visual Illustrations for Real-World Manual Tasks

Anonymous ACL submission

Abstract

Multistep instructions, such as recipes and how-to guides, greatly benefit from visual aids, such as a series of images that accompany the instruction steps. While Large Language Models (LLMs) have become adept at generating coherent textual steps, Large Vision/Language Models (LVLMs) are less capable of generating accompanying image sequences. The most challenging aspect is that each generated image needs to adhere to the relevant textual step instruction, as well as be visually consistent with earlier images in the sequence. To address this problem, we propose an approach for generating consistent image sequences, which integrates a Latent Diffusion Model (LDM) with an LLM to transform the sequence into a caption to maintain the semantic coherence of the sequence. In addition, to maintain the visual coherence of the image sequence, we introduce a copy mechanism to initialise reverse diffusion processes with a latent vector iteration from a previously generated image from a relevant step. Both strategies will condition the reverse diffusion process on the sequence of instruction steps and tie the contents of the current image to previous instruction steps and corresponding images. Experiments demonstrate that the proposed approach successfully maintains semantic coherence and visual consistency across steps in both domains.

1 Introduction

When humans undertake a task with numerous intricate steps, merely reading a step description is limiting, leaving the user to imagine and infer some of the more nuanced details (Choi et al., 2022). Complementing the textual step instructions with images enhances the user experience by better communicating and representing the text semantics and ideas (Serafini, 2014).

Although prompt-based image generation has advanced significantly (Betker et al., 2023; Rombach et al., 2022; Saharia et al., 2022), state-of-the-art

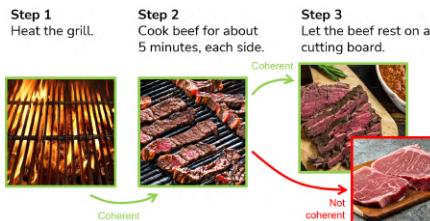


Figure 1: The properties of the elements in illustrations should remain coherent throughout the whole sequence.

(SOTA) models such as Latent Diffusion Models (LDMs) (Rombach et al., 2022) still struggle when generating image sequences to accompany textual instruction steps (Lu et al., 2023). The challenge lies in effectively combining two key aspects: (a) accurately portraying the actions outlined in the step instructions, and (b) ensuring coherence between successive images to avoid confusing the user. Existing storytelling approaches (Feng et al., 2023; Pan et al., 2022; Rahman et al., 2023) operate mostly on linear storytelling and use synthetic cartoon datasets with explicit sequence information, i.e., the textual prompts describe the images appropriately and have no implicit co-references. These aspects limit the applicability of existing methods to real-world scenarios (Figure 1), where there is a lack of informative prompts accompanying images, and dependencies between prompts are not necessarily linear.

In this paper, we explore the generation of image sequences within two domains: recipe instructions, and Do It Yourself (DIY) guides, both showing increasing online consumption (Bausch et al., 2021; Brimble, 2020; Sarpong et al., 2020; Quader, 2022). In these domains, accuracy and coherence are of utmost importance to ensure that the result of all manual actions is correct, and that the user is correctly guided to the target output, Figure 1. These domains contain (i) complex sequential manual

.1. GENERATING CONSISTENT IMAGE SEQUENCES FOR REAL-WORLD MANUAL TASKS

072	tasks of detailed actions, (ii) coherence requirements for the images accompanying the sequence step descriptions, and (iii) a non-linear sequential structure, where steps may be related to earlier steps—not necessarily the previous step.	122
073		123
074		124
075		125
076		126
077	To tackle these challenges, we propose to extend Latent Diffusion Models (Rombach et al., 2022), with an LLM decoder to semantically condition the reverse diffusion process in the sequence of steps and a copy mechanism to select the best LDM initialisation. The image generation process is conditioned on the current step and the previous steps, to increase semantic coherence. In addition, our method initializes the reverse diffusion process with a latent vector iteration copied from a previous generation process to ensure the visual coherence of the generated image. Through this dual attendance to past textual and visual items in the sequence, we aim to achieve <i>semantic coherence</i> , which pertains to the presence and persistence of objects in consecutive images, and <i>visual coherence</i> , which aims to ensure the consistency of backgrounds and visual object properties across successive images.	127
078		128
079		129
080		130
081		131
082		132
083		133
084		134
085		135
086		136
087		137
088		138
089		139
090		140
091		141
092		142
093		143
094		144
095		145
096	An extensive human evaluation confirmed that our model outperforms strong baselines in terms of the overall quality of the generated sequence of illustrations in the cooking and DIY domains.	146
097		147
098		148
099		149
100	2 Related Work	150
101	Methods to generate sequences of images, conditioned on textual input, have been explored in the story visualization and story continuation tasks.	151
102	Story Visualization aims at generating a coherent sequence of images, based on a multi-sentence paragraph or a series of captions forming a narrative (Li et al., 2019; Pan et al., 2022; Maharana et al., 2022).	152
103	Story Continuation is a variant of Story Visualization, in which the generated sequence is initiated by a source image. In both tasks, generating every image independently of other images in the sequence leads to low visual coherence.	153
104		154
105		155
106		156
107	Several works addressed these tasks. Pan et al. (2022) proposed AR-LDM, a method to tackle Story Visualization and Continuation using a history-aware autoregressive latent diffusion model (Rombach et al., 2022), which encodes the history of caption-image pairs into a multimodal representation that guides the LDM denoising process. Despite the intuitive idea, the computational complexity of the conditioning network makes this approach too costly. Additionally, AR-LDM still shows room for improvement in terms of coherence. Feng et al. (2023) noted that AR-LDM conditions the current generation on all historic frames and captions equally, despite not all frames being similarly related. To tackle this limitation, they proposed ACM-VSG, a method that selectively adopts historical text-image data for the generation of the new image. The adaptive encoder automatically finds the relevant historical text-image pairs via CLIP similarity. A key difference between AR-LDM and ACM-VSG is the computational cost: while AR-LDM fine-tunes CLIP, BLIP, and the LDM, ACM-VSG trains only the cross-attention module, at a much lower cost.	157
108		158
109		159
110		160
111		161
112		162
113		163
114		164
115		165
116		166
117		167
118		168
119		169
120		170
121		171

BIBLIOGRAPHY

172 $\{s_1, \dots, s_{i-1}\}$; this way, context is preserved even
 173 when steps are ambiguous or lack information, e.g.
 174 “Add two eggs and mix”. In addition to previous
 175 step instructions, we also need to condition on pre-
 176 viously generated images, $\{v_1, \dots, v_{i-1}\}$, to main-
 177 tain the visual aspects that are only introduced in
 178 the images, such as object properties and back-
 179 ground artefacts not mentioned in the text.

4 Sequential Latent Diffusion Model

The latent diffusion model proposed by Rombach et al. (2022) transforms the diffusion process into a low-dimensional latent space through an encoder $z = E(v)$ and recovers the real image with a decoder $v = D(z)$. The complete model is also conditioned on an input y by augmenting the U-Net backbone with a cross-attention layer to support the encoded input $\tau_\theta(y)$. The conditional LDM is learned with the following loss,

$$\mathcal{L}_{LDM} = \mathbb{E}_{E(v), y, \epsilon, t} \left[\|\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))\|_2^2 \right], \quad (1)$$

where the conditioning encoder $\tau_\theta(y)$ in Eq. 1 uses the entire set of tokens in y to condition the U-Net denoising process in the LDM backbone. The above expression evidences how the conditional LDM is designed to generate one image v at a time. More relevant to our problem is the fact that the latent vectors z_t are independent across different image generations, since the reverse diffusion process iterates from T to 1 starting with a new random seed z_T for every new image generation.

4.1 Sequence Context Decoder

Generally, textual step descriptions describe what the user should do at a specific step of their manual task. These descriptions do not make accurate captions of the accompanying step images as they often contain information that is not visually representable, such as temporal information, “Cook for 10 minutes”, or multiple actions, “Chop the rosemary, dice the carrots, and peel the cucumber.” Additionally, it is also common for steps to not be self-contained, as they depend on the previous step descriptions for context.

To overcome this, for each step s_i , we use a decoder-only model, φ , which we call Sequence Context Decoder, to transform the step and its context into a visual caption c_i which describes the contents of the image v_i . To ensure the generated captions are contextually relevant, we adopt a middle-ground approach and consider the target

Original Step “Slide Dutch baby onto serving platter or cutting board, or serve straight from pan, if desired. Top with fresh berries and sprinkle with lemon juice. Dust with confectioner’s sugar and (...).”



No Context “A person is sprinkling sugar on top of a pancake.”

Current Step as Context “A person is sprinkling confectioner’s sugar on top of a Dutch baby pancake.”

Figure 2: Example captions generated by InstructBLIP. In the “No Context” example, the model only receives the image. In the “Current Step as Context” example, the model receives the image plus the “Original Step”.

step s_i and a context window of w steps. Formally, we define the decoder

$$c_i = \varphi(s_i, \{s_{i-1}, \dots, s_{i-w}\}). \quad (2)$$

to generate a contextual caption c_i from its step description s_i and context.

The decoder $\varphi(\cdot)$ is trained similarly to an image caption generator, but instead of receiving images as input, it receives the step and its context. By training the model to output image captions for the original images that we are trying to replicate, the model learns to generate texts that are more appropriate as image generation prompts. The objective now is learning how to generate better image generation prompts, instead of training the image generation module.

To train the decoder model $\varphi(\cdot)$, we generated contextual captions for each image in dataset \mathcal{D} using InstructBLIP (Li et al., 2022). To achieve richer and contextualized captions, we prompted InstructBLIP with additional context, conditioning the caption on the recipe steps, in addition to the image. Figure 2 shows example captions generated by the model, given a real data point in the dataset.

4.2 Sequence Conditioned Reverse Diffusion

To maintain visual coherence among images in the sequence, we need to condition the current generation on the previous images. Image-to-image (Meng et al., 2022) generation follows this principle, but new images are too strongly influenced by previous ones and do not correctly integrate the new aspects present in the step description.

Following the rationale of conditioning every reverse diffusion process on previous processes, we propose to leverage latent vector iterations from early reverse diffusion processes. This leads us to

.1. GENERATING CONSISTENT IMAGE SEQUENCES FOR REAL-WORLD MANUAL TASKS

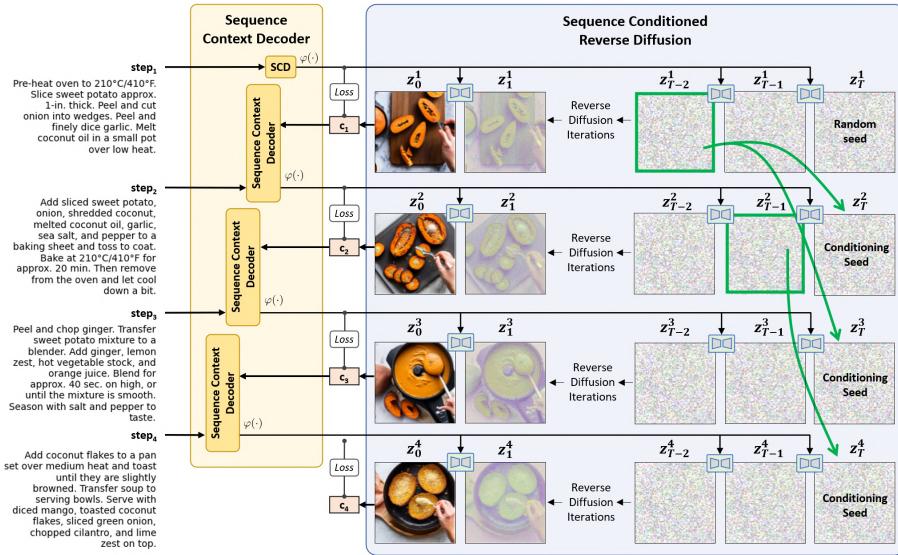


Figure 3: The proposed method uses the sequence context decoder to maintain semantic coherence. The reverse diffusion process uses a conditioning seed z_T^j that is copied from a previous step and iteration z_k^j . See Equation 3.

256 the final formulation of the proposed method,

257

$$\mathcal{L}_{SLDM}(s_i) = \mathbb{E}_{E(v_i), s_i, \epsilon, t} \left[\|\epsilon - \epsilon_\theta(z_t^i, t, \right. \\ \left. \tau_\theta(c_i = \varphi(s_i, \{s_{i-1}, \dots, s_{i-w}\})))\|_2^2 \right] \quad (3)$$

258

259

260 where for each s_i a new reverse diffusion process
261 starts with a conditioning seed z_T^i copied from a
262 previous step s_j with $j < i$ and a latent vector
263 iteration k corresponding to the latent vector z_k^j ,
264 as illustrated in Figure 3. Next, we describe the
265 details of how these two variables are determined.

266

4.2.1 Random and Fixed Seeds

To ground the generation, a straightforward method is to set a fixed seed for every step i in the sequence, $z_T^i = c^{te}$. By fixing the initial seed, we aim to improve the coherence between generated images. The first two columns in Figure 4 demonstrate this approach. We observed a greater homogeneity between generated images when using a fixed seed. Hence, all step illustrations share the same random seed, to achieve more coherent scenes and backgrounds but without the capacity to select the optimal starting seed.

4.2.2 Conditioned Initialisation

While using a fixed seed can improve the results, we argue that a better solution is achieved by using latent vectors from previous reverse diffusion processes. In particular, *latent vectors that have already been semantically conditioned on past steps*. Figure 3 shows how the latent vector representations z_t^i evolve with increasing iterations, until they arrive at the final image $v_i = D(z_0^i)$ for step s_i . These latent representations already contain meaningful information about the image (Mao et al., 2023), which could be leveraged to improve the coherence of the following generations. To achieve this, we need to carefully select which step to choose, to use as input seed for the next step image generation.

A step s_i may be dependent on any previous step $j < i, \{s_{i-1}, \dots, s_1\}$. To select the optimal initialization of the reverse diffusion process for step s_i , we start by determining the most similar step s_j as the

$$\arg \max_j sim(s_i, s_j \in \{s_{i-1}, \dots, s_1\}) \quad (4)$$

where $sim(\cdot)$ represents CLIP text similarity. If this similarity score is above a predefined threshold η , we use s_j to extract the latent vector. If no step

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303	<i>s_j</i> has a similarity score above η , we generated	351
304	image v_i with the shared random seed.	352
305	The reverse diffusion process progressively iterates over the latent vectors towards the final image.	353
306	This means that conditioning the reverse diffusion	354
307	process on latent vector iterations from a later iteration, i.e. a highly denoised latent vector,	355
308	would force the resulting image to be very close to the	356
309	previous one. To decide how strongly we want to	357
310	condition v_i on the step s_j , we select the k^{th} latent	358
311	vector iteration as	359
312		360
313		361
314	$k = n \cdot (\text{sim}(s_i, s_j) - \eta) / (1.0 - \eta) \quad (5)$	362
315	where n is the maximum number of reverse diffusion	363
316	iterations that we consider.	
317	This brings us to the target reverse iteration vector z_k^j which will be used as a starting seed z_T^i in	364
318	Eq 3 when calculating $\mathcal{L}_{SLDM}(s_i)$. Figure 3 illustrates the whole process: the proposed method	365
319	captures the visual aspects that should be in the image, and the linked denoising latent vector provide	366
320	the seed to generate a step image.	367
321		368
322		369
323		370
324	5 Experimental Methodology	371
325	5.1 Dataset	372
326	We collected a dataset consisting of publicly available manual tasks in the recipes domain from All-Recipes .	373
327	We also considered DIY manual tasks from WikiHow , in an out-of-domain evaluation.	374
328	Each manual task has a title, a description, a list	
329	of ingredients/resources, and a sequence of step-by-step instructions, which may or may not be illustrated.	
330	Since we want to illustrate the steps of a task, we focus on manual tasks which are fully illustrated, as we can use these images as ground-truth for training and evaluating our methods.	
331	In total, we used 1100 recipes, with an average of 5.06	
332	steps per recipe, resulting in 5562 individual steps.	
333		
334		
335		
336		
337		
338		
339	5.2 Contextual Caption Generation	375
340	As previously described, we provide InstructBLIP (Li et al., 2022) with additional context to	376
341	produce contextualized captions. We experimented	377
342	with different context lengths and decided to rule	378
343	out experiments that gave InstructBLIP the full context, i.e., all previous step instructions, as this led	379
344	to very long outputs that often repeated irrelevant	380
345	information from the input context, instead of de-	381
346	scribing the image. We addressed the issue of long	382
347	input prompts, by using a context window, as de-	383
348	scribed in Section 4.1. This allows us to give the	384
349		385
350		386
351	model additional context while mitigating possi-	387
352	ble errors in the generated training caption. When	388
353	generating the image captions used to train the Se-	389
354	quence Context Decoder, we produced two sets of	390
355	captions: long and short. The prompt to Instruct-	391
356	BLIP consists of the additional context window	392
357	followed by <i>"Given the steps, give a short descrip-</i>	393
358	<i>tion of the image. Do NOT make assumptions, say</i>	394
359	<i>only what you see in the image."</i>	395
360	We generate captions with a window of 2 steps—short captions—and	396
361	with a window of 3 steps—long captions. Finally,	397
362	we generate a long and a short caption for every	398
363	image, v_i in the dataset.	
364	5.3 Model Details	399
365	To train the sequence context decoder model, we	400
366	fine-tuned an Alpaca-7B model for 10 epochs on a	401
367	single A100 40Gb GPU. We used a cross-entropy	402
368	loss and a cosine learning rate scheduler, starting at	403
369	$1e^{-5}$. The batch size was set to 2, with a gradient	404
370	accumulation step of 4. The dataset had a total of	405
371	5562 step-caption pairs, from which we used	406
372	80% for training and 180 examples for testing. We	407
373	used a frozen Stable Diffusion 2.1 (Rombach et al., 2022) for image generation.	408
374		
375	5.4 Human Annotations	409
376	For evaluating our models, annotators inspected	410
377	30 sequences of images generated with different	411
378	methods (withheld from annotators) and selected	412
379	the 3 best sequences, out of a total of 5 sequences.	413
380	Besides this annotation, we provided an additional	414
381	<i>No good sequence</i> label, for when no sequence of	415
382	images was of good quality. For finer-grain annotations,	416
383	we conducted a side-by-side comparison to	417
384	compare two sequence generation methods. In both	418
385	cases, annotators could also provide feedback on	419
386	the generated errors. See Appendix E for details.	420
387	6 Results and Discussion	421
388	In this section, we report the sequence illustration	422
389	results obtained with the proposed framework	423
390	composed of the sequence context decoder and	424
391	sequence conditioned reverse diffusion methods.	425
392	As baselines, we experimented with using (1) a	426
393	random seed for all steps of the sequence, (2) a	427
394	fixed seed for all steps of the sequence, (3) a fixed	428
395	latent vector iteration from the previous step, repre-	429
396	sented by Latent k , where k is the fixed iteration,	430
397	and (4) the proposed method that selects a latent	431
398	vector iteration, z_i^j from the previous denoising	432

.1. GENERATING CONSISTENT IMAGE SEQUENCES FOR REAL-WORLD MANUAL TASKS

Method	Best (%)	Second Best (%)	Third Best (%)
Random seed	17.70	41.20	13.00
Fixed seed	<u>29.40</u>	17.70	<u>33.30</u>
Latent 1	33.30	17.70	37.04
Latent 2	17.70	<u>23.50</u>	16.70
Img-to-Img	2.00	0.00	0.00

Table 1: Annotation results for the evaluation of the various methods of maintaining visual coherence. Annotators picked *No Good Sequence* in 18.99% of the sequences; we report the results for the remaining 81.01%.

399 steps as a starting seed.

400 6.1 Sequence Generation Results

401 **Recipes Domain.** To validate our initial hypothesis,
402 we start by analysing its performance in the recipes
403 domain. Our goal is to assess how conditioning the
404 reverse diffusion process of each step affects the
405 overall results. Table 1 provides the complete set
406 of results across all competing methods. It is clear
407 how using latent vector iterations and random seeds
408 supports the intuition behind our method: a manual
409 task is composed of continuous and independent
410 actions, which should be conditioned by latent vector
411 iterations and by random seeds, respectively.
412 Building on this observation, we calibrated the η
413 parameter of the proposed method, using human
414 evaluation, as reported in annex in Table 7.

415 To compare the proposed method to the best per-
416 forming method (Latent 1), we asked annotators
417 to select the best sequence out of two side-by-side
418 sequences (see Appendix E for details about the an-
419 notation instructions). Results reported in Table 1
420 show that in 46.7% of the cases, human annotators
421 preferred our method over the alternative and in
422 10.0% of the cases the two methods were equally
423 good. This confirms our hypothesis and supports
424 the importance of selectively conditioning the de-
425 noising process on the previously generated steps
426 of the sequence.

427 **DIY Domain.** To assess the generalisation of the
428 proposed method, we evaluated its performance
429 in an unseen domain: DIY tasks. With the re-
430 sults of our human annotation study, we observed
431 that the transition from generating recipe images
432 to DIY tasks has shown promising results. Results
433 in Table 1 show that in 30.0% of the tasks, neither
434 method produced satisfactory results. For the tasks

Method	Recipes (seen)	DIY (unseen)
Proposed method (wins)	46.67	30.00
Second best (wins)	26.67	23.33
Tie	10.00	16.67
No good sequence	16.67	30.00

Table 2: Annotation results of the comparison between our proposed method and the winning method from Table 1 (Latent 1).

Method	Average Rating
Proposed method	2.93 ± 1.14
Ground-truth	4.58 ± 0.79

Table 3: Human annotation results for the comparison of the proposed method with ground-truth images.

435 that were correctly illustrated, we see that annota-
436 tors preferred our method in 30.0% of the tasks,
437 compared to 23.3% for the second-best approach.
438 Additionally, 16.7% of comparisons resulted in a
439 tie between the two methods. Although we see
440 limitations in this domain, the results show that our
441 approach is capable of generalizing to an unseen
442 domain, producing satisfactory image sequences.

443 **Generated vs Ground-Truth Sequences.** We
444 compared the quality of generated image sequences
445 against the ground-truth sequences and asked hu-
446 man annotators to rate each sequence in a 5 point
447 Likert scale. This is a particularly challenging set-
448 ting because the real image sequences are photos
449 taken by humans in a real-world setting, where
450 sequence coherence is naturally captured. Ta-
451 ble 3 shows that our method achieves over 60%
452 of the ground-truth score, with the ground-truth
453 sequences only 0.42 points below the maximum
454 score.

455 **Qualitative Analysis.** To illustrate how different
456 conditioning methods affect the quality of gener-
457 ated sequences, we present several examples in
458 Figures 4 and 5, and in Appendix F. In Figure 4,
459 we can see that by using a fixed seed for all steps,
460 we are able to preserve the background and add ob-
461 jects for each specific step. We can also see that the
462 image-to-image method conditions the genera-
463 tions too strongly. Figure 4 also shows that our method
464 is capable of preserving and recall key visual arte-
465 facts from several steps back in the sequence. We

BIBLIOGRAPHY



Figure 4: Examples of recipe illustrations with different methods for maintaining visual coherence. Each column illustrates steps 1 through 4.

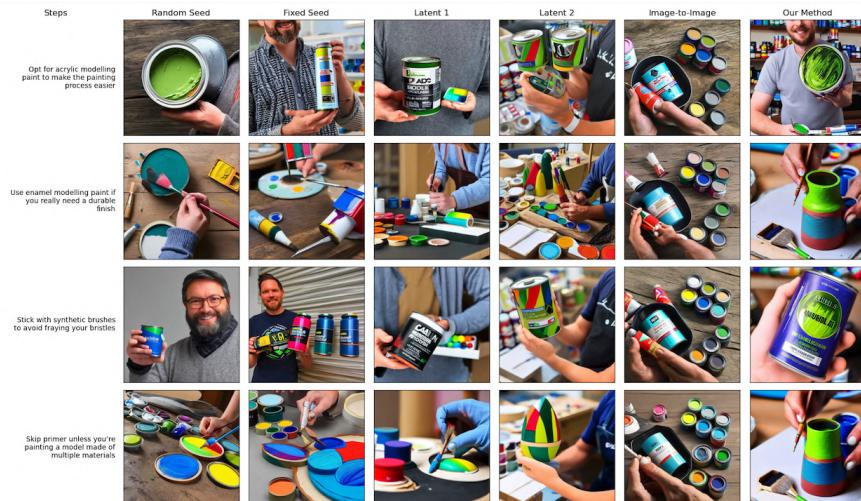


Figure 5: Examples of DIY illustrations with different methods for maintaining visual coherence. Each column illustrates steps 1 through 4.

believe this is a distinctive and fundamental feature of the proposed method.

In the DIY out-of-domain experiment, Figure 5 shows a strong generalization to tasks involving simple object utilisation, such as using a broom for cleaning or a brush for painting. While fixed latent vector iteration methods show some memorization

capability, the image-to-image generations are too biased on previous generations, and random seeds lead to very diverse generations. In this unseen domain, the proposed method encounters considerable challenges when tasked with more intricate activities, like performing a car's oil change with its complex mechanical components, or engaging

466	believe this is a distinctive and fundamental feature of the proposed method.	473
467		474
468	In the DIY out-of-domain experiment, Figure 5 shows a strong generalization to tasks involving simple object utilisation, such as using a broom for cleaning or a brush for painting. While fixed latent	475
469	vector iteration methods show some memorization	476
470		477
471		478
472		479

.1. GENERATING CONSISTENT IMAGE SEQUENCES FOR REAL-WORLD MANUAL TASKS

480 in tasks that involve philosophical or introspective
481 elements, see Appendix F for visual examples. It
482 is worth noting that these limitations are inherited
483 from the core image generation method, which
484 struggles to handle fine-details.

485 6.2 Sequence Conditioned Reverse Diffusion

486 To better understand the strength of our **visual co-**
487 **herence** hypothesis, we conducted an experiment
488 where all the images of the sequence are generated
489 from the latent vectors of a fixed iteration from the
490 previous task step. Specifically, we use the latent
491 vector iteration k , with $k \in \{1, 2, 5, 10, 20, 49\}$, of
492 the previous task step as the starting point of the
493 current reverse diffusion process.

494 Our empirical analysis of these generations,
495 from which some examples can be seen in annex
496 in Figure 6, confirms our initial hypothesis showing
497 that using latent vector iterations from previous
498 steps provides a good result in many cases. Ad-
499 ditionally, this experiment evidences the strength
500 of latent vector iterations as conditioning signals
501 in a reverse diffusion process. This experiment
502 shows that later iterations add a very strong bias
503 to the generation emphasising the importance of
504 selecting the best conditioning seed from previous
505 reverse diffusion iterations and processes. An ex-
506 treme case of this phenomenon is observed in the
507 image-to-image method in Figure 4.

508 6.3 Sequence Context Decoder

509 We assessed the capacity of the sequence context
510 decoder of maintaining the **semantic coherence** by
511 manually annotating the decoder output for six set-
512 tings with different context lengths and caption
513 lengths, Table 4. We considered three context
514 lengths: the shortest one considers the current step,
515 while the longest, shows two steps and a caption.
516 For the captions, we used both *short* and *long* cap-
517 tions, as detailed in Section 5.2.

518 Table 4 shows the results for the different evalua-
519 tion settings. These results indicate that the model
520 attains the best semantic coherence with a context
521 window of 2 and with short captions. We observed
522 that captions generated with short contexts tend to
523 lack some information, while captions generated
524 with longer contexts introduced too much infor-
525 mation, which was often noisy. This is aligned
526 with a recent study (Liu et al., 2023) that highlights
527 the fact that current large language models do not
528 robustly make use of information in long input con-
529 texts. These results indicate that additional context

Sequence Context	Captions	Avg. Rating
{ s_n }	<i>short</i>	3.00
{ s_n }	<i>long</i>	3.23
{ s_n, c_{n-1} }	<i>short</i>	3.68
{ s_n, c_{n-1} }	<i>long</i>	3.56
{ s_n, s_{n-1}, c_{n-2} }	<i>short</i>	3.41
{ s_n, s_{n-1}, c_{n-2} }	<i>long</i>	3.35

Table 4: Sequence Context Decoder results for different context lengths, configurations and caption type.

Error type	%
Hallucinations	3.9%
Complex step with many actions	6.2%
Copied input	7.2%

Table 5: The contribution of each error type to the overall performance.

needs to be carefully considered and curated, as the
model is not able to filter out excess information.

We further analysed the errors of the best method
and present the results in Table 5. Hallucinations
occurred in 3.9% of the generations, and the LLM
copied the input into the output in 7.2% of the cases.
Finally, it is interesting to see that the input was too
complex in 6.2% of the cases, i.e., describing more
actions than what is possible to depict in the image.

7 Conclusions

In this paper, we addressed the problem of illustrating
complex manual tasks and proposed a framework
for generating a sequence of images that illustrate
the manual task. The framework is composed of a novel **sequence context decoder** that preserves the **semantic coherence** of a sequence of actions by transforming it into a visual caption. The full sequence illustration framework is completed by a **sequence conditioned reverse diffusion** process that uses a latent vector iteration from a past image generation process to maintain **visual coherence**. Experiments in the target domain demonstrated the strong performance of the framework in generating coherent sequences of visual instructions of manual tasks. The **generalization to unseen domains** was successfully validated in the DIY domain with positive results.

BIBLIOGRAPHY

<p>558 8 Risks and Limitations</p> <p>559 While we conducted a thorough set of experiments 560 and validations, we acknowledge that more experiments 561 could shed more light in some aspects. First, 562 we did not experiment with larger LLMs to dis- 563 cover the impact of scaling. Second, we also did 564 not consider that steps may dependent on multiple 565 previous steps. Third, the proposed solution only 566 considers a limited window of steps as context, 567 which may lead to some information being lost.</p> <p>568 Finally, in terms of risks, we acknowledge that 569 our work could potentially be used to generate false 570 information.</p> <p>571 References</p> <p>572 Cleber Lemes Bausch, Gabriel Sperandio Milan, 573 Ana Paula Graciola, Luciene Eberle, and Suélen Bebb- 574 er. 2021. The covid-19 pandemic and the changes 575 in consumer habits and behavior. <i>Revista Gestão e Desenvolvimento</i>, 18(3):3–25.</p> <p>576 James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jian- 577 feng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, 578 Joyce Lee, Yufei Guo, et al. 2023. Improving image 579 generation with better captions. <i>Computer Science</i>. 580 https://cdn.openai.com/papers/dall-e-3.pdf.</p> <p>581 Lucy Brimble. 2020. More than 70% of adults use 582 social media for recipes instead of cookbooks, survey 583 finds. <i>Independent UK</i>.</p> <p>584 Jason Ingyu Choi, Saar Kuzi, Nikhita Vedula, Jie Zhao, 585 Giuseppe Castellucci, Marcus Collins, Shervin Mal- 586 masi, Oleg Rokhlenko, and Eugene Agichtein. 2022. 587 Wizard of tasks: A novel conversational dataset for 588 solving real-world tasks in conversational settings. 589 In <i>Proceedings of the 29th International Conference on 590 Computational Linguistics, COLING 2022, Gyeongju, 591 Republic of Korea, October 12–17, 2022</i>, pages 592 3514–3529. International Committee on Com- 593 putational Linguistics.</p> <p>594 Lucia Donatelli, Theresa Schmidt, Debanjali Biswas, 595 Arne Köhn, Fangzhou Zhai, and Alexander Koller. 596 2021. Aligning actions across recipe graphs. In <i>Pro- 597 ceedings of the 2021 Conference on Empirical Meth- 598 ods in Natural Language Processing</i>, pages 6930– 599 6942, Online and Punta Cana, Dominican Republic. 600 Association for Computational Linguistics.</p> <p>601 Zhangyin Feng, Yuchen Ren, Xinmiao Yu, Xiaocheng 602 Feng, Duyu Tang, Shuming Shi, and Bing Qin. 2023. 603 Improved visual story generation with adaptive con- 604 text modeling. In <i>Findings of the Association for 605 Computational Linguistics: ACL 2023</i>, pages 4939– 606 4955, Toronto, Canada. Association for Computa- 607 tional Linguistics.</p> <p>608 Tanmay Gupta, Dustin Schwenk, Ali Farhadi, Derek 609 Hoiem, and Aniruddha Kembhavi. 2018. Imagine 610 this! scripts to compositions to videos. In <i>European Conference on Computer Vision</i>.</p>	<p>611</p> <p>612</p> <p>613</p> <p>614</p> <p>615</p> <p>616</p> <p>617</p> <p>618</p> <p>619</p> <p>620</p> <p>621</p> <p>622</p> <p>623</p> <p>624</p> <p>625</p> <p>626</p> <p>627</p> <p>628</p> <p>629</p> <p>630</p> <p>631</p> <p>632</p> <p>633</p> <p>634</p> <p>635</p> <p>636</p> <p>637</p> <p>638</p> <p>639</p> <p>640</p> <p>641</p> <p>642</p> <p>643</p> <p>644</p> <p>645</p> <p>646</p> <p>647</p> <p>648</p> <p>649</p> <p>650</p> <p>651</p> <p>652</p> <p>653</p> <p>654</p> <p>655</p> <p>656</p> <p>657</p> <p>658</p> <p>659</p> <p>660</p> <p>661</p> <p>662</p> <p>663</p> <p>664</p> <p>665</p>
---	--

.1. GENERATING CONSISTENT IMAGE SEQUENCES FOR REAL-WORLD MANUAL TASKS

- 666 Xichen Pan, Pengda Qin, Yuhong Li, Hui Xue, and
667 Wenhui Chen. 2022. *Synthesizing coherent story with*
668 *auto-regressive latent diffusion models*.
- 669 Shams Bin Quader. 2022. How the central sydney inde-
670 pendent musicians use pre-established ‘online diy’ to
671 sustain their networking during the covid-19 pan-
672 demic. *The Journal of International Communication*,
673 28(1):90–109.
- 674 Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya
675 Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sas-
676 try, Amanda Askell, Pamela Mishkin, Jack Clark,
677 Gretchen Krueger, and Ilya Sutskever. 2021. *Learn-
678 ing transferable visual models from natural language
679 supervision*.
- 680 Tanzila Rahman, Hsin-Ying Lee, Jian Ren, Sergey
681 Tulyakov, Shweta Mahajan, and Leonid Sigal. 2022.
682 *Make-a-story: Visual memory conditioned consistent
683 story generation*.
- 684 Tanzila Rahman, Hsin-Ying Lee, Jian Ren, Sergey
685 Tulyakov, Shweta Mahajan, and Leonid Sigal. 2023.
686 *Make-a-story: Visual memory conditioned consistent
687 story generation*. In *Proceedings of the IEEE/CVF
688 Conference on Computer Vision and Pattern Recog-
689 nition (CVPR)*, pages 2493–2502.
- 690 Robin Rombach, Andreas Blattmann, Dominik Lorenz,
691 Patrick Esser, and Björn Ommer. 2022. High-
692 resolution image synthesis with latent diffusion mod-
693 els. In *Proceedings of the IEEE/CVF Conference on
694 Computer Vision and Pattern Recognition (CVPR)*,
695 pages 10684–10695.
- 696 Chitwan Saharia, William Chan, Saurabh Saxena, Lala
697 Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed
698 Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi,
699 Rapha Gontijo Lopes, et al. 2022. Photorealistic
700 text-to-image diffusion models with deep language
701 understanding. *arXiv preprint arXiv:2205.11487*.
- 702 David Sarpong, George Ofosu, David Botchie, and Fin-
703 tan Clear. 2020. Do-it-yourself (diy) science: The
704 proliferation, relevance and concerns. *Technological
705 Forecasting and Social Change*, 158:120127.
- 706 Frank Serafini. 2014. *Reading the visual: An intro-
707 duction to teaching multimodal literacy*. Teachers
708 College Press.

709

A Data Preparation

710
711
712
713
714
715

We found that recipes with long steps descriptions or many steps, were difficult to tackle and produced worse results. This pointed towards a refinement of the dataset, so that our approach could better focus on the issue of coherence, instead of tackling other problems.

716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731

When a step description is too long, it contains too much information, often with multiple actions, which is hard to represent in a single image. Adding to this issue, the CLIP (Radford et al., 2021) text encoder, used in the Stable Diffusion (Rombach et al., 2022) model, truncates the input text at 77 tokens. We filtered the recipes that had steps that were too long. A second problem arises when a recipe has too many steps, as it is difficult to produce coherent illustrations over such a long sequence of steps. To mitigate this concern, we limited the number of steps in a recipe from 4 to 6 steps. In a final stage of refining the dataset, we removed any steps that did not contain actions which we could illustrate, such as steps merely saying “Enjoy!”.

732
733

To the best of our knowledge, we do not use any personal identifiable information.

734
735
736

We plan to release the data under an Apache-2.0 licence. We intend to allow future work to further research the problem, using the provided data.

737

B Model training

738
739
740
741
742
743
744
745
746
747
748
749
750
751
752

We chose Alpaca-7B as our base model, as it is an open-source instruction-tuned model, and there are implementations using LoRA (Hu et al., 2022), which reduces the computational cost during training. We encourage future work to study the impacts of scaling the LLM. We experimented with different hyperparameters, namely different learning rates, and learning rate schedulers, to find the most suitable ones for our problem. We point out that we are using the loss of the models as the main criterion for future experiments, as we do not have an automatic metric for evaluating model behaviour. The loss is not always indicative of the model’s performance in the task at hand, as we verified empirically.

753
754
755
756
757
758

Since the cosine scheduler has greater variability between runs, due to different number of epochs leading to different loss curves, we decided to run further experiments with the next best-performing model. We experimented with varying the weight decay parameter but found that there were no sig-

Training Details	
Base Model	Alpaca-7B
Training Time	$\approx 10h$
Epochs	10
Loss Function	Cross-Entropy
Weight Decay	0.01
Model Max Length	400
Batch Size	2
Gradient Accumulation Steps	4
Effective Batch Size	8
Learning Rate	$1e^{-5}$
Learning Rate Scheduler	Cosine
Optimizer	AdamW
Adam β_1	0.900
Adam β_2	0.999
Adam ϵ	$1e^{-8}$
LoRA	
LoRA Rank	8
LoRA α	32
LoRA Dropout	0.1

Table 6: Training parameters for the best model.

nificant differences in the loss curves for the three weight decay values.

For the aforementioned runs, we used $\beta_1 = 0.900$ and $\beta_2 = 0.999$ as the AdamW optimizer’s hyperparameters. As a final test, we changed these to the values proposed by Ouyang et al. (2022), $\beta_1 = 0.900$ and $\beta_2 = 0.950$. We did not see any improvement in the loss curve.

Based on these results, we fine-tuned our Alpaca-7B models for 10 epochs on a single A100 40Gb GPU. We used a cross-entropy loss, a cosine learning rate scheduler, starting at $1e^{-5}$. Our batch size was 2, with a gradient accumulation step of 4, leading to an effective batch size of 8. The dataset had a total of 5562 examples; we used 80% for training and the remaining for evaluation. Figure 6 summarizes the training information for our best-performing model.

C Fixed Latent Iteration Generation

As mentioned in Section 6.2, we conducted an empirical analysis of generations using a fixed latent vector iteration from the previous step. This analysis helped us understand how different latent vector iterations impact the generation of the following

759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782

.1. GENERATING CONSISTENT IMAGE SEQUENCES FOR REAL-WORLD MANUAL TASKS

η	Best (%)	Second Best (%)	Third Best (%)
0.70	19.20	12.80	14.90
0.65	12.80	14.90	25.50
0.60	19.20	23.40	21.30
0.55	14.90	23.40	21.30
0.50	34.04	25.53	17.02

Table 7: Annotation results for the sequences generated with different threshold values, η . Annotators picked *No Good Sequence* in 20.34% of the generations; we show results for the remaining 79.66%.

783 image. An example from this analysis is shown in
784 Figure 6.

D Negative Prompts

786 We found problems which were not related to the
787 prompts or the concepts we were trying to generate,
788 but general problems in image generation, i.e., tiled
789 images, or deformed hands. In order to reduce
790 some of these common problems, present in Stable
791 Diffusion generations, we used negative prompts.
792 A negative prompt steers the generation away from
793 the concepts present in it. This string of text is
794 added to the end of the original prompt.

795 In the negative prompt, we included undesirable
796 concepts such as *human* or *hands*, and also
797 included some additional concepts, following common
798 practices. Our final negative prompt follows:
799 *negative_prompts = ["hands", "human", "person",*
800 *"cropped", "deformed", "cut off", "malformed",*
801 *"out of frame", "split image", "tiling", "watermark",*
802 *"text"]*.

E Human Annotations

803 In this Section, we present examples of the annotation
804 tasks.

805 The human annotation pool consisted of 3 PhD
806 students and 5 MSc students. 25% of the annotators
807 were women and 85% were men.

808 Figure 7 shows the annotation task to choose the
809 best visual coherence maintaining method. The
810 annotators saw 5 sequences: *Random Seed*, *Fixed*
811 *Seed*, *Latent 1*, *Latent 2*, and *Image-to-Image*. They
812 were then asked to pick the best, second best, and
813 third best sequences. They could also indicate that
814 there were no good sequences, by checking the
815 *No good sequence* checkbox. Additionally, they
816 could leave an observation, in the appropriate text

area. Figure 8 shows the annotation task to tune
818 the threshold of our method. The annotators had
819 to pick between 5 sequences, generated with dif-
820 ferent values of threshold: 0.50, 0.55, 0.60, 0.65,
821 0.70. They were asked to pick the best, second
822 best, and third best sequences. They could also
823 indicate that there were no good sequences, by
824 checking the *No good sequence* checkbox. They
825 could also leave an observation, in the appropriate
826 text area. Figure 9 shows the annotation task to
827 choose between our method and the winning visual
828 coherence maintaining method. The annotators
829 saw 2 sequences, one generated with *Latent 1* and
830 another with our method. They had to choose the
831 win sequence, or deem them equivalent. If there
832 was no good sequence, they could check the *No
833 good sequence* checkbox. They could also leave
834 an observation. Figure 10 shows the annotation
835 task to rate sequences generated with our method
836 and the ground-truth images, from a scale of 1 to
837 5. Additionally, the annotators could select that
838 there was no good sequence, or leave an observa-
839 tion. Figure 11 shows the annotation guidelines
840 for the task to rate sequences generated with our
841 method and the ground-truth images.

F Examples and Qualitative Analysis

842 Table 8 shows some example generations from the
843 Sequence Context Decoder, each highlighting a
844 particular behaviour of the model. In Example 1,
845 we can see the model correctly identifying the
846 ingredients from the context, *caption_{n-2}*, going two
847 steps back, and integrating them in the final out-
848 put. It also recognizes the plate from *step_n* as the
849 object containing the ingredients. This shows the
850 potential in giving the model additional context to
851 generate richer prompts. In Example 2, we can
852 see that, despite being able to maintain the *red*
853 *apples*, the model makes no explicit reference to
854 their state, chopped up. This is still a limitation,
855 which may lead to a wrongful representation of
856 intact apples. In Example 3, we want to highlight
857 two main aspects of the generation: we can see the
858 model adding the *bowl of soup* from the context to
859 the prompt, maintaining semantic coherence. We
860 can also see that the model kept *lime juice*. This
861 is correct, from the point of view of the task at
862 hand, but shows the lack of understanding of what
863 can be perceived in an image. We reason that this
864 knowledge should come from the pretraining of the
865 model, and not from our fine-tuning to this task.

BIBLIOGRAPHY



Figure 6: Maintaining visual coherence through the use of different memory latent vectors.

Sequence Evaluation

Jamaican Callaloo With Shrimp

Steps

A B C D E

Add the Small Shrimp, stir well and cook for another 3 minutes.

Turn the heat up to medium high and add the Jamaican Callaloo, Scotch Bonnet Pepper, Fresh Thyme, and Sea Salt. After a couple minutes, add the coconut milk and cook until tender.

After about 10-12 minutes, taste for salt and adjust accordingly.

Best Sequence: A B C D E
Second Best Sequence: A B C D E
Third Best Sequence: A B C D E
 No good sequence

Observations: _____

Submit

Figure 7: Annotation of the comparison between visual coherence methods.

Sequence Evaluation

Jamaican Callaloo With Shrimp

Steps

A B C D E

Add the Coconut Oil in a wide pan over a medium flame, then add the Scallion, and Ground Black Pepper. Reduce the heat to low for about 1-2 minutes.

Turn the heat up to medium high and add the Jamaican Callaloo, Scotch Bonnet Pepper, Fresh Thyme, and Sea Salt. After a couple minutes, add the coconut milk and cook until tender.

After about 10-12 minutes, taste for salt and adjust accordingly.

Best Sequence: A B C D E
Second Best Sequence: A B C D E
Third Best Sequence: A B C D E
 No good sequence

Observations: _____

Submit

Figure 8: Annotation of the comparison between different heuristic thresholds.

868
869
870
871
872
873
874
875

Example 4 shows an example of a depiction that is mostly correct, but misses a step of the sequence. The representation of the saucepan with black tea in it is plausible, but step_n indicates the saucepan should be removed from the heat. Finally, in Example 5, we see a very long step, with various actions. In this case, we consider it plausible for the model to pick one of these actions. This is a better re-

sult than attempting to represent them all, which would lead to an inadequate prompt. Despite this, this specific generation lacks some context, as the word *vegetables* is generic; it is important for the generated prompts to be specific, containing the ingredients mentioned in the context.

876
877
878
879
880
881

.1. GENERATING CONSISTENT IMAGE SEQUENCES FOR REAL-WORLD MANUAL TASKS

Example 1: The Sequence Context Decoder was able to dereference the objects from caption_{n-2} and create the correct caption for step_n.

Caption_{n-2}	A wooden cutting board with ingredients for a salad including avocado , crab meat , and grapes
Step_{n-1}	Dress all ingredients except the arugula with one fourth cup dressing, dress the arugula separately with two tablespoon dressing.
Step_n	Plate the arugula first. Combine the rest of the ingredients, and add on top.
Output	A plate of salad with avocado , crab meat , and grapes on top.

Example 2: The state of objects is not always preserved in the generated caption.

Caption_{n-1}	A purple plastic cup is sitting on a wooden table. Inside the cup, there is a mixture of black pepper, olive oil, brown sugar, curry powder, salt, and white vinegar. There are also some sultanas floating in the mixture. The cup is covered with plastic wrap.
Step_n	Get White Rice in a big bowl. Add in Celery and Scallion. In order to add a bit of color and some sweetness, chop up Red Apple and dust that up. Toss them together.
Output	A bowl of white rice is being prepared with celery, scallions, and red apples

Example 3: Some objects are correctly added to the generated caption.

Caption_{n-1}	In the image, there is a pot of soup on the stove with chicken and vegetables in it. There is also a person's hand reaching into the pot to add more ingredients to the soup.
Step_n	Serve with White Rice on the bottom and Lime Wedge on the side. Or with tortillas for dipping into the soup.
Output	In the image, there is a bowl of soup with a tortilla in it. The soup has chicken, vegetables, and lime juice in it.

Example 4: Sequence of actions was not fully captured by the model.

Step_{n-1}	Bring Water to boil in a medium saucepan.
Step_n	Once water has boiled, remove saucepan from heat and add the Black Tea. Steep for 15 minutes.
Output	a saucepan with black tea in it on a stove top

Example 5: When steps provide plenty of context, using a single step is sufficient.

Step_n	Halve fennel, remove stalk, and cut into fine strips. Finely dice onion. Peel garlic. Clean zucchini and bell pepper and cut into bite-sized pieces. Combine vegetables with rosemary, thyme, and some of the olive oil in a plastic bag. Close, shake, and marinate for approx. 20 min.
Output	A woman is cutting up vegetables on a cutting board with a knife and a peeler.

Table 8: Qualitative analysis of the Sequence Context Decoder results.

BIBLIOGRAPHY

Sequence Evaluation

Jamaican Callaloo With Shrimp

Steps

Heat the Coconut Oil in a wide pan over a medium flame, then add the Onion, Garlic, Scallions, and Black Pepper. Reduce the heat to low for about 3-4 minutes.



Add the Small Shrimp, stir well and cook for another 3 minutes.



Turn the heat up to medium high and add the Jamaican Callaloo, Tomato, Scotch Bonnet Pepper, Fresh Thyme, and Sea Salt. After a couple of minutes, stir the mixture and cook until tender.



After about 10-12 minutes, taste for salt and adjust accordingly.



Best Sequence:

- A
- B
- Equivalent

- No good sequence

Observations

Submit

Sequence Evaluation

Jamaican Callaloo With Shrimp

Steps

Heat the Coconut Oil in a wide pan over a medium flame, then add the Onion, Garlic, Scallions, and Black Pepper. Reduce the heat to low for about 3-4 minutes.



Add the Small Shrimp, stir well and cook for another 3 minutes.



Turn the heat up to medium high and add the Jamaican Callaloo, Tomato, Scotch Bonnet Pepper, Fresh Thyme, and Sea Salt. After a couple of minutes, stir the mixture and cook until tender.



Rating for A:

- 1
- 2
- 3
- 4
- 5

Rating for B:

- 1
- 2
- 3
- 4
- 5

- No good sequence

Observations

Submit

Figure 9: Annotation of the comparison between our method and the best visual coherence method.

Instructions

We will present you two side-by-side sequences of illustrations for a recipe. On the left of the sequences, you will see the step instruction.

We ask you to rate each sequence on a Likert scale of 1-5.

Your rating should based on two factors:

- How well does each illustration represent the step instruction?
 - Note: Generation problems should not affect the rating if the image still clearly illustrates the step.
- How coherent is the sequence of illustrations, as a whole?
 - Example: A blue pan remains blue in the following images.
 - Example: The background scene is preserved between images.

If none of the sequences correctly illustrates the recipe, you can check the option: No Good Sequence.

You can also leave an observation, before submitting your answer.

Figure 11: Annotation guidelines for the comparison between our method and the ground-truth images.

.1. GENERATING CONSISTENT IMAGE SEQUENCES FOR REAL-WORLD MANUAL TASKS

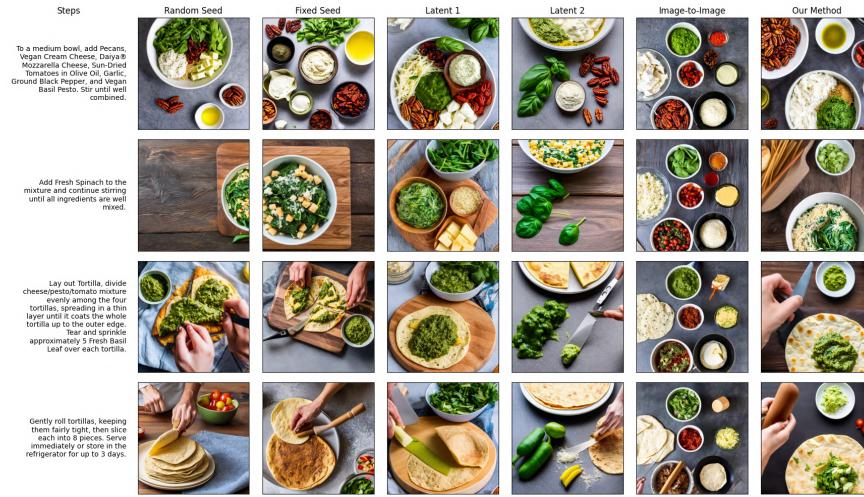


Figure 12: Examples of recipe illustrations with different methods for maintaining visual coherence.



Figure 13: Examples of recipe illustrations with different methods for maintaining visual coherence.

BIBLIOGRAPHY



Figure 14: Examples of recipe illustrations with different methods for maintaining visual coherence.

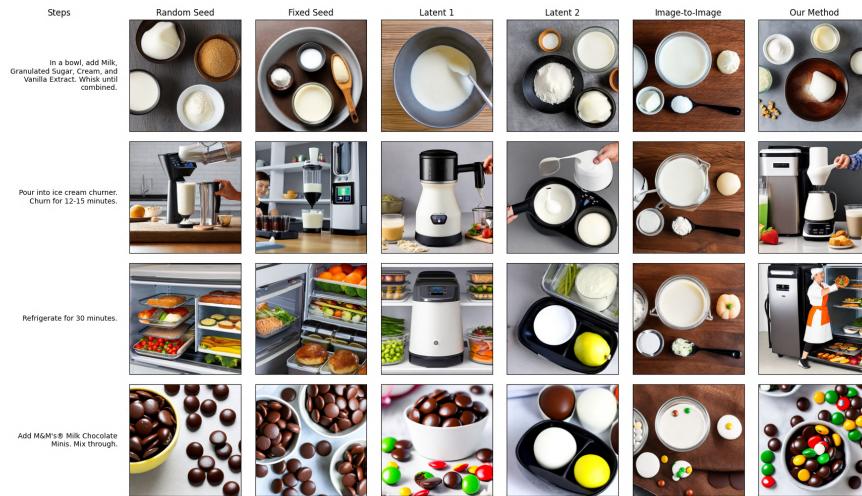


Figure 15: Examples of recipe illustrations with different methods for maintaining visual coherence.

.1. GENERATING CONSISTENT IMAGE SEQUENCES FOR REAL-WORLD MANUAL TASKS

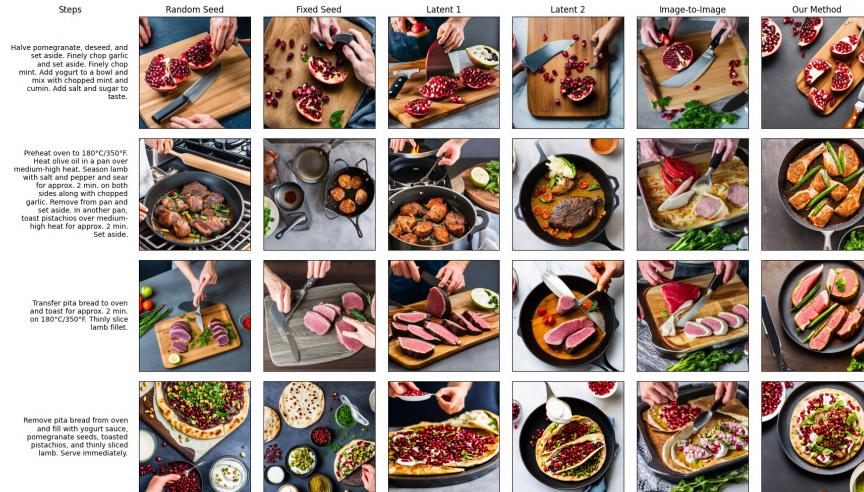


Figure 16: Examples of recipe illustrations with different methods for maintaining visual coherence.



Figure 17: Examples of recipe illustrations with different methods for maintaining visual coherence.

BIBLIOGRAPHY



Figure 18: Examples of task illustrations with different methods for maintaining visual coherence.



Figure 19: Examples of task illustrations with different methods for maintaining visual coherence.

.1. GENERATING CONSISTENT IMAGE SEQUENCES FOR REAL-WORLD MANUAL TASKS

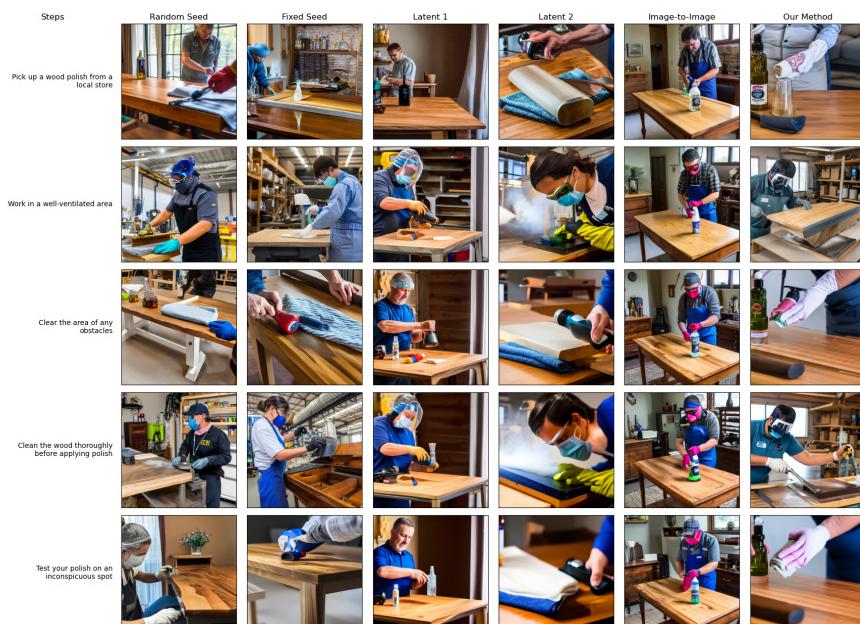


Figure 20: Examples of task illustrations with different methods for maintaining visual coherence.

BIBLIOGRAPHY



Figure 21: Example of a task that is very challenging to illustrate. We can see how the generated images still capture some of the more challenging elements of the steps, such as "Make a note of the longitude and latitude" in step 4, with the images showing a pen.

.2 Aligning Natural Prompts and Image Generation

Verified Alignment of Natural Prompts with Image Generation

Anonymous EMNLP submission

Abstract

001
002
003
004
005
006
007
008
009
010
011
012
013
014
015
016
017
018
019
020
021
022
023
024
025
026
027
028

Text-to-image generation methods (T2I) are widely popular in generating art and other creative artefacts. While visual hallucinations can be a positive factor in scenarios where creativity is appreciated, such artefacts are poorly suited for cases where the generated image needs to be grounded in complex natural language without explicit visual elements. In this paper, we propose to strengthen the consistency property of T2I methods in the presence of *natural complex language*, which often breaks the limits of T2I methods by including non-visual information, and textual elements that require knowledge for accurate generation. To address these phenomena, we propose a **Natural Language to Verified Image generation approach (NL2VI)** that simultaneously (i) re-writes the natural prompt into a *visual prompt*, which is more suitable for image generation, and (ii) generates verification visual questions for assessing the visual consistency of the generated images. Experimentally, aligning natural prompts with image generation can improve the consistency of the generated images by up to 11% over the current state-of-the-art. Moreover, improvements can generalize to challenging domains like cooking and DIY tasks, where the correctness of the generated image is crucial to illustrate actions.

029

1 Introduction

030
031
032
033
034
035
036
037
038
039
040
041

Text-to-image generation (T2I) methods [Ramesh et al., 2022, Rombach et al., 2022, Saharia et al., 2022, Chang et al., 2023] are able to map textual prompts to latent image representations in order to represent objects, actions, scenes, or emotions mentioned in the prompt. Yet, these models still often produce inconsistencies between the prompt and the image [Park et al., 2021, Leivada et al., 2022], as well as hallucinations that escape visual common sense knowledge, e.g. the left side of Figure 1. Visual and common sense inconsistencies are further exacerbated when the input is a natural

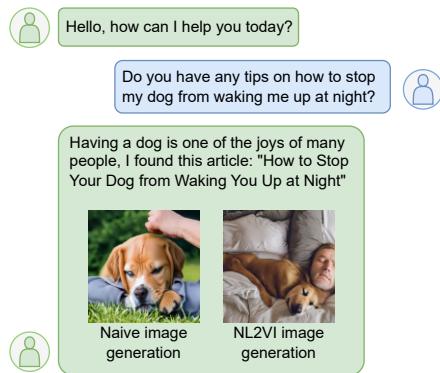


Figure 1: NL2VI can transfer the visual attributes of natural language into generated images and measure its consistency score.

text, as opposed to direct explicit drawing instructions. Natural texts in dialogs and documents, often include information that extends beyond the visual realm, such as emotions and domain knowledge inference, e.g., “*how to buy company shares*”. This creates a reliability problem when deploying T2I methods and automatic image verification methods in applications such as illustration of dialog utterances, DIY or recipes.

In this paper, we propose a **Natural Language to Verified Image generation approach (NL2VI)** to *simultaneously align image generation with natural prompts and verify the consistency of the generated image with respect to the natural prompt*. NL2VI moves beyond synthetic and caption-based prompts and explicitly converts natural language prompts into visually plausible text, which we refer to as a *visual prompt*. This conversion is achieved by employing a few-shot LLM that rewrites the natural text, removing non-visual aspects and providing details for textual information that would require common or domain knowledge inference to accurately generate the image.

042
043
044
045
046
047
048
049
050
051
052
053
054
055
056
057
058
059
060
061
062
063
064

.2. ALIGNING NATURAL PROMPTS AND IMAGE GENERATION

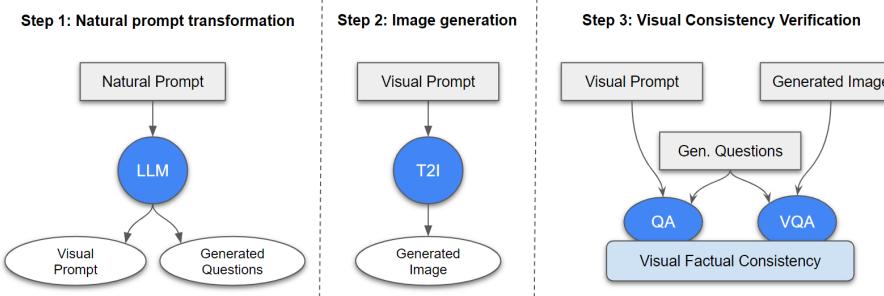


Figure 2: Natural language to verified image generation (NL2VI).

A key novelty of NL2VI, is that the few-shot LLM also generates verification questions when the prompt is rewritten, step 1 of Figure 2. This is a clear advance over previous methods [Cho et al., 2022, Gokhale et al., 2022] that relied on heuristics extracted from image metadata, together with an object detector, or on questions generated from synthetic prompts [Hu et al., 2023]. In NL2VI we use a T2I method to generate candidate images for the visual prompt (step 2 of Figure 2), and rank the candidate images by a consistency verification process based on the generated questions, step 3 of Figure 2.

We demonstrate that our method significantly surpasses previous T2I methods when utilizing real-world natural prompts. Our experiments show that the visual prompt accurately represents the visual elements in the natural prompt, achieving an AUC of over 94%. Furthermore, our method excels at producing images that closely match real-world prompts and exhibits superior accuracy in predicting image consistency. We attain an enhancement of 7.8% and 11.0% in alignment accuracy for the recipes and DIY domains, respectively, compared to the existing state-of-the-art.

2 Aligning Natural Prompts with Image Generation

To improve the alignment between *natural prompts* and generated images, we propose the NL2VI approach to transform the natural prompt into a *visual prompt*, in which all the visual elements present in the natural prompt are well-identified and complete, while non-visual elements are removed. We hypothesize that visual prompts would narrow the gap that T2I models need to bridge between the natural text input and the resulting image, and reduce

visual hallucinations, or the generation of implausible artefacts.

Concretely, we propose the process depicted in Figure 2, consisting of three phases that leverage recent advancements in LLMs and VQA algorithms. In the initial phase, a large language model, such as PaLM [Chowdhery et al., 2022] or GPT-3.5, distills a visual prompt from an input natural prompt. The LLM also explicitly indicates the main visual aspects that must be verified in a generated image, presenting them as a list of question-answer pairs. In the second phase, a conditioned T2I model generates the image, taking as input the visual prompt. Finally, in the third phase, a Visual Question Answering (VQA) model answers the textual questions generated in the first phase, based on the generated images. These VQA answers are then compared to the expected textual answers, to identify any inconsistencies between the natural prompt and the generated image, ensuring image consistency.

When all these aspects are taken into account, we obtain a method that offers guarantees of generating an image aligned with a natural prompt, even when that prompt provides no visual clues. In the following Sections, we elaborate on each of these phases.

2.1 Visual Prompt and Question Generation for Visual Consistency

Modern LLMs have been trained on very large corpora, and across a wide range of tasks. Leveraging this rich training data, LLMs are able to detect which elements of a textual passage are visually transferable into an image. Figure 3 illustrates how “*purchasing company shares*” is transformed to “*computer screen with market shares*”. We build on this property of LLMs to translate a natural

065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
100

101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136

BIBLIOGRAPHY

137 prompt into a visual prompt, through in-context
138 few-shot learning, see Table 1. Additionally, we
139 aim to verify that a generated image, conditioned
140 on the generated visual prompt, faithfully follows
141 the prompt instruction. To this end, we also instruct
142 the LLM with in-context few-shot examples to gen-
143 erate a set of question/answer pairs that are used
144 to verify the alignment between the natural prompt
145 and the generated image, see Table 1. We have two
146 types of questions: binary questions, which serve
147 to verify the presence of objects from the prompt
148 in the image, and open-ended questions, which
149 are more general. The distribution of questions is
150 further detailed in Table 6.

151 Lastly, it’s important to acknowledge that, given
152 the generative nature of this phase, there is a poten-
153 tial for anomalies, such as hallucinations and incon-
154 sistencies. To address potential inconsistencies in
155 the generative process, we adopt a robust approach,
156 recognized widely for its efficacy: the application
157 of generated question/answer pairs [Honovich et al.,
158 2021, Dagan et al., 2006, Honovich et al., 2022].
159 These question/answer pairs are used to validate
160 the consistency of the generated images, by eval-
161 uating whether the VQA answers, derived from
162 the generated images, align with the QA answers,
163 grounded exclusively in the visual prompt. This
164 topic is explored in further detail in Sections 2.3
165 and 4.4.

2.2 Text-to-Image Generation

In this phase, a T2I model is conditioned on the visual prompt, with more visual details and fewer ambiguous descriptions, as depicted in step 2 of Figure 2. The visual prompts were shorter than the input limit of the T2I methods we tested, hence no truncation was needed, in contrast to the natural prompts, which often exceeded these limits.

2.3 NL2VI Consistency Verification

The final phase in NL2VI involves verifying the consistency of the generated image, in relation to the natural prompt. The rationale is to check the consistency of the visual prompt with respect to the natural prompt, followed by the generated questions and lastly, the generated image. Inspired by Honovich et al. [2022] and Hu et al. [2023], we leverage the questions generated in phase 1 to probe both the image and the prompt, evaluating the consistency of the answers. To filter the questions which are not relevant to the prompt, we use a Question Answering (QA) model [Khashabi et al.,

As an AI Image Verification Specialist, your primary responsibility is to create a text2img prompt and examine its accuracy assuming an associated image. Your task involves two main steps:

Construct a text2img prompt: You will be provided with a description. It’s crucial that your text2img prompt incorporates all visual aspects mentioned in the description. The text2img prompt must be a detailed visual description that accurately represents the visual attributes of the description. Non-visual attributes should not be included.

Formulate a series of questions: Your questions must be related to the visible components within the image. Questions must be simple, unambiguous, and answerable based on the observable content in the image. Questions must be about elements that exist in the image and are clearly visible. Questions must be about a single element.

Follow the examples below and complete:

Description: "Mango-Black Bean Salsa. This fiery, flavorful salsa is excellent with tortilla chips, spooned over roasted meat or fish, or as a topping for quesadillas. Made with mango, avocado, no-salt-added black beans, red onion, jalapeño pepper, cilantro, lime, salt."

text2img prompt: A bowl of mango and black bean salsa with tortilla chips. The salsa also has onions, jalapeño peppers and cilantro.
Q: is there a bowl of food? A: yes
Q: is there salsa? A: yes
Q: are there black beans in the salsa? A: yes
Q: Are there mangos in the salsa? A: yes
Q: are there tortilla chips? A: yes
Q: is there cilantro? A: yes
.....
(in context learning examples)
.....
(unseen example)
Description: "Garlic Parmesan Pasta. The hardest part is chopping the parsley. Made with: parsley, garlic, butter, chicken broth, milk, parmesan cheese, salt, ground pepper."

text2img prompt: A bowl of garlic parmesan pasta with parmesan cheese and parsley.
Questions:
Q: what is in the bowl? A: pasta
Q: is there a bowl of food? A: yes
Q: is there cheese? A: yes
Q: is there cheese on the pasta? A: yes
Q: is there parsley? A: yes

Table 1: In-context instructions used to transform a natural prompt into a visual prompt and a set of visual consistency evaluation questions.

2020] in conjunction with a Natural Language In-
ference (NLI) model [Nie et al., 2020]. Although
we investigated various methods to verify the con-
sistency between the natural prompt and the visual
prompt, the fact is that the performance of the LLM
has an AUC of over 90%, which makes it suffi-
ciently robust to consider that they are correct, in
general.

We are then tasked with verifying the consis-
tency of the generated images against the LLM-
generated questions. To accomplish this, we em-
ploy VQA models capable of eliciting open-ended
responses. This can lead to semantic differences
between the ground truth and model-generated an-
swers. Hence, to compare these answers, we eval-
uated several text-matching algorithms, including
string equality, BERTScore [Zhang et al., 2019a],
and NLI [Dagan et al., 2006] models.

Figure 3 illustrates the NL2VI method. On
the Recipes domain, the images from the natural
prompt miss important visual ingredients, like pars-

187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207

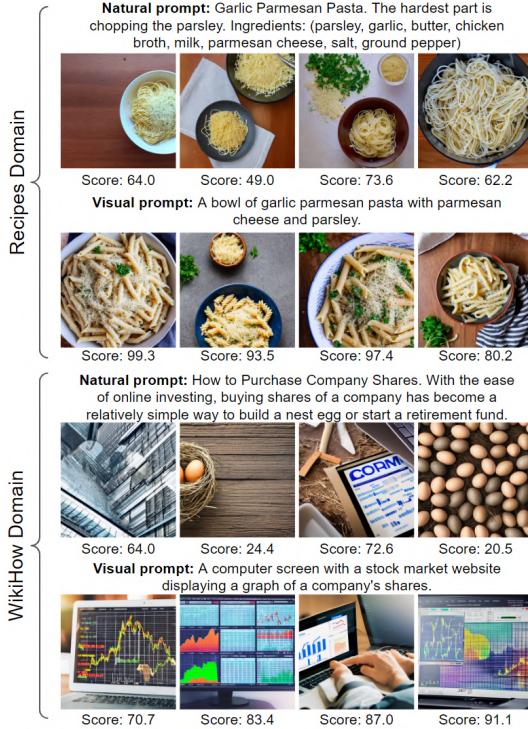


Figure 3: Comparison of natural and visual prompt for image generation across different domains.

ley. The visual prompt improves the consistency and consists only of relevant visual information. On the WikiHow domain, the natural prompt is penalized for being mostly non-visual information, which degrades image generation. However, the visual prompt is able to overcome these problems.

3 Experimental Methodology

In this Section, we present the NL2VI implementation details, and describe the experimental setup used to evaluate it.

3.1 Implementation

The implementation of NL2VI, as shown in Figure 4, relies on many pre-trained models that are used throughout the various pipeline stages. Initially, the natural prompt is transformed into a visual prompt, and the consistency verification questions are generated, utilizing in-context learning. Our experiments involved two large language models: gpt-3.5-turbo from OpenAI and

PaLM 540B [Chowdhery et al., 2022]. For reproducibility purposes, we release the visual prompts and generated questions, detailed in Appendix B. Secondly, in the image generation step, we employ the Stable Diffusion 2.1 model, conditioned on the computed prompts, to generate the target image. While SD 2.1 currently defines the state-of-the-art, in the future, NL2VI can be easily adapted to any other image generation method. Finally, the generated questions are answered by both QA and VQA methods based on the visual prompt and on the image together with the visual prompt, respectively. For QA filtering, our tests included the widely popular QANLU [Namazifar et al., 2021] and the UnifiedQA model [Khashabi et al., 2020], followed by the NLI model RoBERTa-NLI [Nie et al., 2020]. Regarding VQA, our experiments involved the following models: BLIP [Li et al., 2022b], GIT [Wang et al., 2022a], OFA [Wang et al., 2022b], PaLI [Chen et al., 2022] and mPLUG [Li et al., 2022a].

227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247

BIBLIOGRAPHY

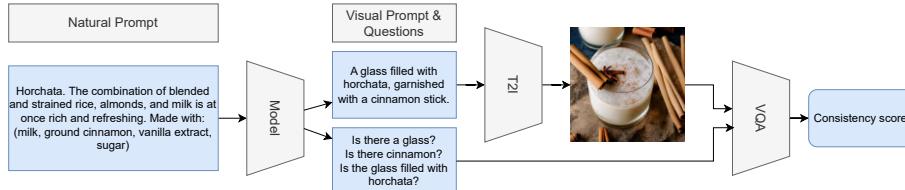


Figure 4: The NL2VI method implements an in-context few-shot learning approach: an LLM is responsible for generating the visual prompt and the verification questions for the VQA method.

3.2 Baselines

NL2VI is a general approach technique that supports any T2I model for the image generation step, such as DALL-E2 [Ramesh et al., 2022] or Imagen [Saharia et al., 2022]. In the following experiments, we used Stable Diffusion 2.1. Results with other T2I models can be found in the supplementary material. Moreover, the consistency verification step can also be done using other consistency verification algorithms, such as the CLIP-Score [Hessel et al., 2021] or the TIFA model [Hu et al., 2023].

3.3 NL2VI Public Dataset¹

To study the generalization of T2I methods, we benchmark their performance in settings with natural prompts in the Recipes and WikiHow domains. The **NL2VI natural prompts and questions** dataset comprises 3000 curated natural prompts, where the correct illustration of an action and correct composition of overlapping objects is both challenging and critical. It was designed to allow for the realistic, yet controllable, research of image generation methods conditioned on real-world natural prompts. See Annex B for details.

4 Results and Discussion

Next, we present and discuss the NL2VI method’s experimental results, highlighting the key takeaways. All the experiments were run on a 1x NVIDIA A100 GPU.

4.1 Verified Image Generation Results

In this Section, we compare the performance of NL2VI against other verified image generation approaches. We consider several LLMs and VQA methods, as well as the CLIPScore [Hessel et al., 2021] metric and the recent TIFA model [Hu et al., 2023].

¹Available after publication.

Models	Recipes	Wikihow
CLIPScore	57.4	53.8
TIFA	72.5	64.9
NL2VI	80.3	76.0

Table 2: Human evaluation of the different visual factual consistency methods.

Table 2 summarizes the overall experimental results. The LLM and VQA models displayed have the highest accuracy for each method. For the full results, please refer to table 5. We observed that CLIPScore has a very low variance across all generated images, with a mean value of $\sim 32\%$. We also noticed that this metric is not correlated with the visual consistency. This might be explained by two factors: first, CLIPScore is already used by some T2I methods as the metric to be optimized, and second, CLIPScore does not capture fine-grained information between the image and the prompt, as discussed by related works [Yuksekgonul et al., 2023]. TIFA performs better than CLIPScore, achieving good results with explicit prompts, but fails when presented with more challenging real-world natural language prompts. NL2VI is clearly superior to the other methods, specially with the combination of GPT-3.5 and PaLI. NL2VI was able to solve common inconsistencies present in other methods, w.r.t the lack of visual common sense knowledge.

4.2 Visual Prompt Consistency

In this Section, we analyse the consistency of the visual prompt with respect to its alignment with the natural prompt. The objective is to understand how well the visual elements of a natural prompt are unambiguously captured in the visual prompt. A human annotation task was set up for this purpose. Figure 5 presents the precision curve over the annotated corpus, and Table 2 presents summarized metrics of the curves. In the Recipes do-

248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283

284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314

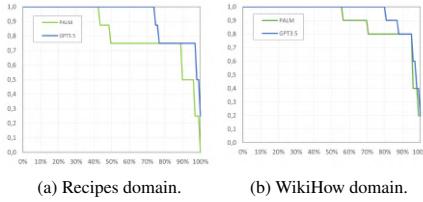


Figure 5: Visual prompt consistency with respect to the natural prompt.

Models	Recipes		Wikihow	
	AUC	P@1	AUC	P@1
PaLM	82.5	42.0	90.2	56.0
GPT-3.5	92.8	74.0	94.9	80.0

Table 3: Alignment between natural language prompts and visual generation prompts.

main, both LLMs perform exceedingly well, with GPT-3.5 achieving an average precision of 92.8% and PaLM 82.5%. This performance is even better in the WikiHow domain, with 94.9% and 90.2% average precision for GPT-3.5 and PaLM, respectively. In terms of precision at 1–prompts that are fully correct–precision decreases in both models, with GPT-3.5 being able to generate visual prompts that could capture the visual elements of natural prompts in 74.0% of the cases in the Recipes domain and 80.0% of the cases in the WikiHow domain. This is a highly positive result, given that natural language prompts can depict a wide range of situations and concepts, which may not have an obvious visual representation. Appendix ?? illustrates some challenging examples.

4.3 Ablation Studies

When comparing the VQA generative answers with the QA extractive answers, there are several discrepancies that need to be bridged due to the differences in the methods. Computing the correspondence between answers with string matching algorithms would be rather limiting, which is why we investigated NLI [Dagan et al., 2006] and BERTScore [Zhang et al., 2019a], as alternatives. Figure 6 provides a visualization of the score distribution for each method. TIFA images, which are based on natural prompts have, on average, lower scores than our images generated from visual prompts. Therefore, based on NL2VI, our images are more likely to be consistent with the prompt.

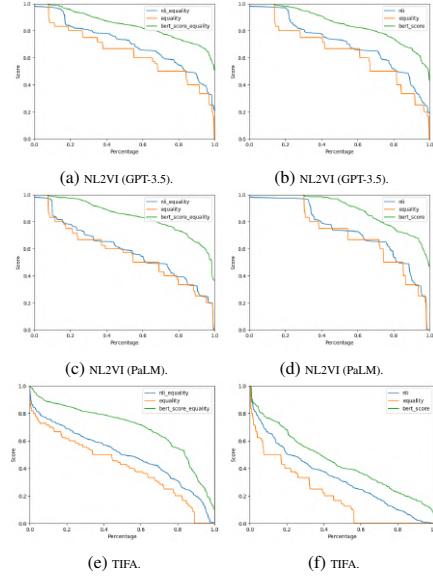


Figure 6: Image consistency on the Recipes (top row) and WikiHow (bottom row), according to the QA and VQA answers verification with string matching, NLI and BERTScore.

Tables 4 and 5 illustrate the comparison between answer matching algorithms, as judged by human annotators. The equality strategy is used for simple answers, like *yes* and *no*. NLI is used for longer answers, where entailment properties need to be checked. BERTScore achieved the best performance on the Recipes domain, while NLI was the best in the WikiHow domain. The first fact that stands out is that, independently of the method we use to validate the images, we observed that visual prompts (GPT-3.5 and PaLM) generate images that are better aligned with the natural prompt than images generated with the original natural prompt. This is due to the linguistic ambiguities that may exist in the natural prompt and need to be solved by T2I algorithms. With visual prompts, these linguistic ambiguities were solved by an LLM, which is better at handling linguistic idiosyncrasies. Furthermore, the visual prompt is more concise than the natural prompt and is therefore unaffected by the token limit of some T2I algorithms.

4.4 Impact of QA and VQA performance

In this Section, we report the results of an ablation study concerning the importance of the QA and

347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370

BIBLIOGRAPHY

	Equality	NLI	BERT-Score
TIFA			
BLIP	64.7	68.9	72.0
GIT	62.4	67.3	72.5
OFA	61.8	65.4	<u>72.2</u>
mPLUG	64.3	67.4	71.1
PaLI	61.9	66.5	68.3
NL2VI w/ GPT-3.5			
BLIP	73.7	76.2	79.8
GIT	72.1	74.7	79.2
OFA	70.3	73.0	79.9
mPLUG	74.3	75.9	79.9
PaLI	75.5	78.3	80.3
NL2VI w/ PaLM			
BLIP	76.5	77.3	78.2
GIT	72.8	74.1	78.4
OFA	70.1	70.9	78.4
mPLUG	75.4	76.3	77.9
PaLI	75.7	76.5	77.7

Table 4: Ablation study on the Recipes domain: analysis of VQA and answer validation methods.

	Generated Questions	Verified Questions
Recipes		
Binary	392	386
Open-Ended	233	174
Wikihow		
Binary	403	388
Open-Ended	222	158

Table 6: Distribution of generated questions for the Recipes and WikiHow datasets before and after filtering (UnifiedQA).

371 VQA methods in the image consistency verification
 372 process. Table 6 presents the statistics of the image
 373 verification questions that need to be answered by
 374 the QA and VQA methods. The key fact to note in
 375 this table is that the manual annotations of the gen-
 376 erated questions show that 89.6% and 87.4% of the
 377 questions in the Recipes and WikiHow domains,
 378 respectively, are valid, thus confirming the overall
 379 quality of the generated questions. Moreover, we
 380 can see that many questions are open-ended, which
 381 forces the use of generative VQA methods. Table 7
 382 presents the results of two QA methods. QA meth-
 383 ods were restricted to span-extraction methods and
 384 multiple-choice approaches, hence the reason for
 385 using the QANLU [Namazifar et al., 2021] and
 386 the UnifiedQA [Khashabi et al., 2020] algorithms.
 387 UnifiedQA was superior both in terms of preci-

	Equality	NLI	BERT-Score
TIFA			
BLIP	56.1	61.6	64.2
GIT	56.1	62.7	64.8
OFA	56.3	62.7	64.2
mPLUG	58.4	64.3	64.9
PaLI	58.3	65.1	64.8
NL2VI w/ GPT-3.5			
BLIP	73.1	75.8	73.8
GIT	71.0	73.9	73.9
OFA	71.6	73.6	73.6
mPLUG	73.5	76.1	73.8
PaLI	74.0	<u>76.0</u>	74.1
NL2VI w/ PaLM			
BLIP	73.0	73.5	69.8
GIT	70.8	71.3	69.7
OFA	72.5	73.6	69.8
mPLUG	72.3	<u>72.7</u>	69.3
PaLI	<u>72.7</u>	72.6	70.0

Table 5: Ablation study on the WikiHow domain: anal-
 ysis of VQA and answer validation methods.

sion and recall, and was the model chosen for our
 experiments.

With respect to VQA methods, we ran an ex-
 tensive ablation study as presented on Table 4 for
 the Recipes domain and Table 5 for the WikiHow
 domain. We cross-examined five VQA genera-
 tive algorithms with natural prompts (TIFA) and
 visual prompts (PaLM and GPT-3.5) and three
 answer comparison methods: equality, NLI and
 BERTScore. From these results, we can observe
 that mPLUG was always the best, or the second
 best performing method, in all settings. PaLI was
 the best, or second best, in four of the six exper-
 imental settings. A key insight that we get from
 these results is that the performance of the VQA
 method is *directly connected* to the performance of
 the visual consistency verification of the generated
 image.

5 Related work

Assessing the consistency of language or image
 generation methods is still an unsolved problem,
 despite having been addressed in the NLP field un-
 der different formulations, i.e., entailment [Li et al.,
 2018, Falke et al., 2019], counterfactual informa-
 tion [Zhang et al., 2019b], and question-answer
 approaches [Honovich et al., 2021, Gupta et al.,
 2022]. Early work was done in the related tasks
 of face image hallucination [Wang et al., 2014]
 and image forensics to detect deepfakes and image

Question filtering	% of valid questions		Precision		Recall	
	Recipes	Wikihow	Recipes	Wikihow	Recipes	Wikihow
QANLU	50.4	50.8	22.2	33.3	28.6	28.6
Unified-QA	89.6	87.8	90.9	84.2	71.4	76.2

Table 7: Evaluation of the question filtering stage.

tampering [Nowroozi et al., 2021]. However, in the image generation domain, only after the publication of DALL-E [Ramesh et al., 2021] and Stable Diffusion [Rombach et al., 2022] models, has the community started to take the first steps towards assessing the visual consistency of T2I algorithms [Rassin et al., 2022, Leivada et al., 2022, White and Cotterell, 2022, Gokhale et al., 2022, Petsiuk et al., 2022, Park et al., 2021, Russo, 2022]. The hallucinations and errors of T2I methods were recently discussed by Rassin et al. [2022], Leivada et al. [2022], White and Cotterell [2022], Petsiuk et al. [2022], shedding some light on the lack of visual consistency between the generated image and the prompt. An inspiring step is taken by Russo [2022], discussing possible methods to evaluate the artistic value of image generation methods. However, better methods for evaluating visual consistency are still lacking. For example, traditional image quality metrics are not fine-grained enough, e.g., the inception score [Salimans et al., 2016] and FID [Heusel et al., 2017] are intended to measure the realism of the generated images and fail to catch inconsistencies [Park et al., 2021]. End-to-end evaluation of the similarity of image-text embeddings obtained with a dual-encoder multimodal model like CLIP [Radford et al., 2021] fails to encode compositional information, which is crucial for visual consistency evaluation [Yuksekgonul et al., 2022]. There have been attempts to improve such models, like CLIP-R Park et al. [2021] and CLIPScore [Hessel et al., 2021], but results are still suboptimal.

Another line of work proposed several synthetic benchmarks and metrics to measure the visual consistency properties of T2I methods. Early approaches relied upon object detection models and heuristics. Gokhale et al. [2022] proposed the VISOR metric, which evaluates the spatial relationship between objects detected in the scene. Similarly, Cho et al. [2022] proposed the PaintSkills dataset to assess object presence, properties and the spatial relationships described in the prompt. Drawbench is another benchmark introduced by Imagen [Saharia et al., 2022]. A more generic ap-

proach was proposed by Hu et al. [2023], which follows the literature of natural language factual consistency with QA [Honovich et al., 2022, Durmus et al., 2020]. In this method, several questions are generated from the prompt and verified in the image with a VQA method. Concurrent to our work, Yarom et al. [2023] introduces the SeeTRUE benchmark for meta-evaluation of image-text alignment. While all these methods can measure T2I average correctness with synthetic or visually descriptive prompts, they are not designed to correctly generate images conditioned on natural prompts.

6 Conclusions

Contributions. Generating images from natural language that is non-visual is, in many cases, an impossible task for text to image methods. In this context, the key contributions of this paper are twofold.

First, the NL2VI method to transfer the visual attributes of a natural language into a visual prompt that will generate a verified image. The visual prompt is correctly aligned with the natural prompt in over 90% of the cases and is also an enabler of a verification process based on VQA methods. The image verification step is the final safeguard to ensure that the image was correctly generated, according to the initial text.

Second, a *public dataset* with natural prompts, visual prompts and verification questions to benchmark image generation methods in the presence of natural language. The curated dataset aggregates a series of natural language prompts that are from the instructions domain with descriptions that are not always visual.

Broader Impacts. By improving the consistency of an image generation process, we are trying to more faithfully depict what is described in the original prompt. The most obvious adverse impact, is the malicious use of generative image generation for disinformation or deceiving. We are against the applications of generative AI non-ethical uses and argue for a responsible and accountable use of these algorithms.

BIBLIOGRAPHY

504	7 Limitations	
505	Although our method improves consistency, image	
506	generation works in an open-world setting, e.g. the	
507	prompt "an ice-cream scoop" will force the NL2VI	
508	to make an under-specified decision. To this end,	
509	the verification is limited to the information present	
510	in the prompt, which sets an upperbound on the ver-	
511	ification process. As we elaborate in Section D, this	
512	work assumes a closed-world setting when verify-	
513	ing the consistency of an image. In other words, we	
514	follow a prompt-based verification, which will miss	
515	some detail of visual consistency, i.e. the ice-cream	
516	color/flavor of the example above.	
517	Furthermore, when generating visual prompts,	
518	some elements may be added, which were not	
519	present, originally. This can lead to some differ-	
520	ences between the expected result and NL2VI's	
521	output. There is still important work to be done to	
522	be able to support a more broad verification of gen-	
523	erated images, in particular the inclusion of visual	
524	commonsense.	
525	References	
526	H. Chang, H. Zhang, J. Barber, A. Maschinot, J. Lezama,	
527	L. Jiang, M.-H. Yang, K. Murphy, W. T. Freeman,	
528	M. Rubinstein, et al. Muse: Text-to-image generation	
529	via masked generative transformers. <i>arXiv preprint</i>	
530	<i>arXiv:2301.00704</i> , 2023.	
531	X. Chen, X. Wang, S. Changpinyo, A. Piergiovanni,	
532	P. Padlewski, D. Salz, S. Goodman, A. Grycner,	
533	B. Mustafa, L. Beyer, et al. Pali: A jointly-scaled	
534	multilingual language-image model. <i>arXiv preprint</i>	
535	<i>arXiv:2209.06794</i> , 2022.	
536	J. Cho, A. Zala, and M. Bansal. Dall-eval: Prob-	
537	ing the reasoning skills and social biases of text-	
538	to-image generative transformers. <i>arXiv preprint</i>	
539	<i>arXiv:2202.04053</i> , 2022.	
540	A. Chowdhery, S. Narang, J. Devlin, M. Bosma,	
541	G. Mishra, A. Roberts, P. Barham, H. W. Chung,	
542	C. Sutton, S. Gehrmann, P. Schuh, K. Shi,	
543	S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes,	
544	Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du,	
545	B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Is-	
546	ard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghe-	
547	mawat, S. Dev, H. Michalewski, X. Garcia, V. Misra,	
548	K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan,	
549	H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Do-	
550	ohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pil-	
551	lai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child,	
552	O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta,	
553	M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-	
554	Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel.	
555	Palm: Scaling language modeling with pathways,	
556	2022.	
I. Dagan, O. Glickman, and B. Magnini. The pascal	557	
recognising textual entailment challenge. In <i>Machine</i>	558	
<i>Learning Challenges. Evaluating Predictive Uncer-</i>	559	
<i>tainty, Visual Object Classification, and Recognising</i>	560	
<i>Tectual Entailment: First PASCAL Machine Learn-</i>	561	
<i>ing Challenges Workshop, MLCW 2005, Southamp-</i>	562	
<i>ton, UK, April 11-13, 2005, Revised Selected Papers</i> ,	563	
pages 177–190. Springer, 2006.	564	
E. Durmus, H. He, and M. Diab. FEQA: A question	565	
answering evaluation framework for faithfulness assess-	566	
ment in abstractive summarization. In <i>Proceedings of</i>	567	
<i>the 58th Annual Meeting of the Association for Com-</i>	568	
<i>putational Linguistics</i> , pages 5055–5070, Online,	569	
July 2020. Association for Computational Linguistics.	570	
doi: 10.18653/v1/2020.acl-main.454. URL https://aclanthology.org/2020.acl-main.454 .	571	
https://aclanthology.org/2020.acl-main.454 .	572	
T. Falke, L. F. R. Ribeiro, P. A. Utama, I. Dagan,	573	
and I. Gurevych. Ranking generated summaries by	574	
correctness: An interesting but challenging applica-	575	
tion for natural language inference. In <i>Proceedings</i>	576	
<i>of the 57th Annual Meeting of the Association for Com-</i>	577	
<i>putational Linguistics</i> , pages 2214–2220, Flo-	578	
<i>rence, Italy, July 2019. Association for Computa-</i>	579	
<i>tional Linguistics</i> . doi: 10.18653/v1/P19-1213. URL	580	
https://aclanthology.org/P19-1213 .	581	
T. Gokhale, H. Palangi, B. Nushi, V. Vineet, E. Horvitz,	582	
E. Kamar, C. Baral, and Y. Yang. Benchmarking spa-	583	
tial relationships in text-to-image generation. <i>arXiv</i>	584	
<i>preprint arXiv:2212.10015</i> , 2022.	585	
P. Gupta, C.-S. Wu, W. Liu, and C. Xiong. DialFact:	586	
A benchmark for fact-checking in dialogue. In <i>Pro-</i>	587	
<i>ceedings of the 60th Annual Meeting of the Associa-</i>	588	
<i>tion for Computational Linguistics (Volume 1: Long</i>	589	
<i>Papers)</i> , pages 3785–3801, Dublin, Ireland, May	590	
2022. Association for Computational Linguistics.	591	
doi: 10.18653/v1/2022.acl-long.263. URL https://aclanthology.org/2022.acl-long.263 .	592	
https://aclanthology.org/2022.acl-long.263 .	593	
J. Hessel, A. Holtzman, M. Forbes, R. L. Bras,	594	
and Y. Choi. Clipscore: A reference-free evalua-	595	
tion metric for image captioning. <i>arXiv preprint</i>	596	
<i>arXiv:2104.08718</i> , 2021.	597	
M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler,	598	
and S. Hochreiter. Gans trained by a two time-scale	599	
update rule converge to a local nash equilibrium. <i>Ad-</i>	600	
<i>vances in neural information processing systems</i> , 30,	601	
2017.	602	
O. Honovich, L. Choshen, R. Aharoni, E. Neeman,	603	
I. Szpektor, and O. Abend. Q2: Evaluating factual	604	
consistency in knowledge-grounded dialogues via	605	
question generation and question answering. <i>arXiv</i>	606	
<i>preprint arXiv:2104.08202</i> , 2021.	607	
O. Honovich, R. Aharoni, J. Herzig, H. Taitelbaum,	608	
D. Kukliansky, V. Cohen, T. Scialom, I. Szpektor,	609	
A. Hassidim, and Y. Matias. True: Re-evaluating	610	
factual consistency evaluation. <i>arXiv preprint</i>	611	
<i>arXiv:2204.04991</i> , 2022.	612	

.2. ALIGNING NATURAL PROMPTS AND IMAGE GENERATION

613 614 615 616	Y. Hu, B. Liu, J. Kasai, Y. Wang, M. Ostendorf, R. Krishna, and N. A. Smith. Tifa: Accurate and interpretable text-to-image faithfulness evaluation with question answering, 2023.	670 671
617 618 619 620 621 622 623 624	D. Khashabi, S. Min, T. Khot, A. Sabharwal, O. Tafjord, P. Clark, and H. Hajishirzi. UNIFIEDQA: Crossing format boundaries with a single QA system. In <i>Findings of the Association for Computational Linguistics: EMNLP 2020</i> , pages 1896–1907, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.171. URL https://aclanthology.org/2020.findings-emnlp.171 .	672 673 674 675 676
625 626 627	E. Leivada, E. Murphy, and G. Marcus. Dalle-2 fails to reliably capture common syntactic processes. <i>arXiv preprint arXiv:2210.12889</i> , 2022.	677 678 679 680 681 682
628 629 630 631 632 633 634 635 636 637	C. Li, H. Xu, J. Tian, W. Wang, M. Yan, B. Bi, J. Ye, H. Chen, G. Xu, Z. Cao, J. Zhang, S. Huang, F. Huang, J. Zhou, and L. Si. mPLUG: Effective and efficient vision-language learning by cross-modal skip-connections. In <i>Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing</i> , pages 7241–7259, Abu Dhabi, United Arab Emirates, Dec. 2022a. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.488 .	683 684 685 686 687
638 639 640 641 642 643 644 645	H. Li, J. Zhu, J. Zhang, and C. Zong. Ensure the correctness of the summary: Incorporate entailment knowledge into abstractive sentence summarization. In <i>Proceedings of the 27th International Conference on Computational Linguistics</i> , pages 1430–1441, Santa Fe, New Mexico, USA, Aug. 2018. Association for Computational Linguistics. URL https://aclanthology.org/C18-1121 .	688 689 690 691
646 647 648 649 650	J. Li, D. Li, C. Xiong, and S. Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In <i>International Conference on Machine Learning</i> , pages 12888–12900. PMLR, 2022b.	692 693 694 695
651 652 653 654 655 656	M. Namazifar, A. Papangelis, G. Tur, and D. Hakkani-Tür. Language model is all you need: Natural language understanding as question answering. In <i>ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)</i> , pages 7803–7807. IEEE, 2021.	696 697 698 699 700
657 658 659 660 661 662	Y. Nie, A. Williams, E. Dinan, M. Bansal, J. Weston, and D. Kiela. Adversarial NLI: A new benchmark for natural language understanding. In <i>Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics</i> . Association for Computational Linguistics, 2020.	701 702 703 704
663 664 665 666	E. Nowroozi, A. Dehghantanha, R. M. Parizi, and K.-K. R. Choo. A survey of machine learning techniques in adversarial image forensics. <i>Computers & Security</i> , 100:102092, 2021.	705 706 707 708 709 710
667 668 669	D. H. Park, S. Azadi, X. Liu, T. Darrell, and A. Rohrbach. Benchmark for compositional text-to-image synthesis. In <i>Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)</i> , 2021.	711 712 713 714
670 671	V. Petsiuk, A. E. Siemann, S. Surbehera, Z. Chin, K. Tyser, G. Hunter, A. Raghavan, Y. Hicke, B. A. Plummer, O. Kerret, et al. Human evaluation of text-to-image models on a multi-task benchmark. <i>arXiv preprint arXiv:2211.12112</i> , 2022.	715 716 717 718
672 673 674 675 676	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 <i>International conference on machine learning</i> , pages 8748–8763. PMLR, 2021.	719 720 721
677 678 679 680 681 682	A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. Zero-shot text-to-image generation. In <i>International Conference on Machine Learning</i> , pages 8821–8831. PMLR, 2021.	722 723 724 725 726 727
683 684 685 686 687	A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents. <i>arXiv preprint arXiv:2204.06125</i> , 2022.	728 729 730 731 732
688 689 690 691	R. Rassin, S. Ravfogel, and Y. Goldberg. Dalle-2 is seeing double: flaws in word-to-concept mapping in text2image models. <i>arXiv preprint arXiv:2210.10606</i> , 2022.	733 734 735 736 737
692 693 694 695	R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In <i>Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition</i> , pages 10684–10695, 2022.	738 739 740 741
696 697 698 699 700	I. Russo. Creative text-to-image generation: Suggestions for a benchmark. In <i>Proceedings of the 2nd International Workshop on Natural Language Processing for Digital Humanities</i> , pages 145–154, 2022.	746 747 748 749 750
705 706 707 708 709 710	C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. <i>Advances in Neural Information Processing Systems</i> , 35:36479–36494, 2022.	755 756 757 758 759 760
711 712 713 714	T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. <i>Advances in neural information processing systems</i> , 29, 2016.	765 766 767 768
715 716 717 718	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. <i>arXiv preprint arXiv:2205.14100</i> , 2022a.	769 770 771 772
719 720 721	N. Wang, D. Tao, X. Gao, X. Li, and J. Li. A comprehensive survey to face hallucination. <i>International journal of computer vision</i> , 106:9–30, 2014.	773 774 775

BIBLIOGRAPHY

722	P. Wang, A. Yang, R. Men, J. Lin, S. Bai, Z. Li,	A	NL2VI Qualitative Examples	755
723	J. Ma, C. Zhou, J. Zhou, and H. Yang. Unifying			756
724	architectures, tasks, and modalities through a sim-			757
725	ple sequence-to-sequence learning framework. <i>arXiv</i>			758
726	<i>preprint arXiv:2202.03052</i> , 2022b.			759
727	J. C. White and R. Cotterell. Schr"\{o} dinger's			760
728	bat: Diffusion models sometimes generate poly-			761
729	semous words in superposition. <i>arXiv preprint</i>			762
730	<i>arXiv:2211.13095</i> , 2022.			
731	M. Yarom, Y. Bitton, S. Changpinyo, R. Aharoni,			
732	J. Herzig, O. Lang, E. Ofek, and I. Szpektor. What			
733	you see is what you read? improving text-image			
734	alignment evaluation, 2023.			
735	M. Yuksekgonul, F. Bianchi, P. Kalluri, D. Jurafsky,			
736	and J. Zou. When and why vision-language models			
737	behave like bag-of-words models, and what to do			
738	about it? <i>arXiv preprint arXiv:2210.01936</i> , 2022.			
739	M. Yuksekgonul, F. Bianchi, P. Kalluri, D. Jurafsky,			
740	and J. Zou. When and why vision-language models			
741	behave like bags-of-words, and what to do about it?,			
742	2023.			
743	T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and			
744	Y. Artzi. Bertscore: Evaluating text generation with			
745	bert. <i>arXiv preprint arXiv:1904.09675</i> , 2019a.			
746	Y. Zhang, J. Baldridge, and L. He. PAWS: Paraphrase			
747	adversaries from word scrambling. In <i>Proceedings</i>			
748	<i>of the 2019 Conference of the North American Chap-</i>			
749	<i>ter of the Association for Computational Linguistics:</i>			
750	<i>Human Language Technologies, Volume 1 (Long and</i>			
751	<i>Short Papers)</i> , pages 1298–1308, Minneapolis, Min-			
752	nesota, June 2019b. Association for Computational			
753	Linguistics. doi: 10.18653/v1/N19-1131. URL			
754	https://aclanthology.org/N19-1131 .			

2. ALIGNING NATURAL PROMPTS AND IMAGE GENERATION

Natural prompt:	Visual Prompt (GPT 3.5)	Visual Prompt (PaLM)
Beef Tacos de Lengua (Beef Tongue Tacos). Classic Mexican tacos de lengua, beef tongue which has been braised with garlic and onions, finely chopped, and served with salsa verde and avocados. Made with: (bay leaves, garlic, corn tortillas, cilantro, oil, avocado, white onion, red onion, radishes)		
Kid-Friendly Instant Apple Crisp. Make this easy, no-cook recipe with kids for breakfast, dessert or just an afternoon snack. Warm it in the microwave for just a few seconds, if you like. Made with: (applesauce, ground cinnamon, granola)		
Chimichurri Sauce. Fresh herbs, garlic, olive oil and red wine vinegar come together in a tangy sauce that works wonders on steak, contrasting the heavier flavors and textures inherent to beef. Made with: (garlic, red wine vinegar, red pepper flakes, extra virgin olive oil, oregano, parsley, salt)		
Gulab Jamun (Indian Donuts). Soaked in sweet syrup, these Indian-style doughnuts can be enjoyed warm, hot or cold! Made with: (milk, sugar, flour, oil, butter, salt, baking powder)		
Tangerine-Roasted Tuna with Green Olives. Serve this one-pan dish with brown rice, red onions and green olives over cooked brown rice for a quick and easy meal. Watch our how-to video. Made with: (pitted green olives, red onion, tuna steak, ground pepper, salt, tangerine)		
No-Bake Cheesecake With Freeze-Dried Fruit. Here's how to transform no-bake cheesecakes with freeze-dried fruit. Made with: (sugar, cream, lemon, butter, salt)		
Grilled Shrimp Skewers with Chimichurri. An herby, garlicky chimichurri sauce doubles as both the marinade and dipping sauce for these easy shrimp skewers. Made with: (Water, tap, drinking, garlic, red wine vinegar, red pepper flakes, lemon, oil, oregano, parsley, salt)		
Unicorn Swiss Roll. How to make a magical AF unicorn roll cake! Made with: (egg, flour, sugar, oil)		

Figure 7: Example of natural prompts and visual prompts on the recipes domain.

BIBLIOGRAPHY

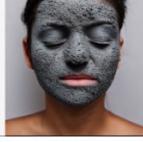
Natural prompt:	Visual Prompt (GPT 3.5)	Visual Prompt (PaLM)
How to Clean Your Room in Less Than Thirty Minutes. Oh no! It's time to face the biggest mess in the house - your room. Are you among the hundreds of people out there needing to clean your room? Then you came to the right place!	 A messy room with clothes and items scattered on the floor and bed.	
How to Use Charcoal Scrubs. Some people swear by charcoal products for skin care. While there is limited evidence on the effectiveness of such products, they are generally safe to use. You can use charcoal to enhance your appearance. You can also use it to address skin issues, like acne.	 A face with charcoal scrub on it.	
How to Make Money in School. Being in school, whether high school or college, doesn't mean you don't need money. If you are looking for ways to make money while attending school full time, you may not be able to find a job that requires regular hours that are likely to interfere with school. [...]	 A student sitting at a desk with a laptop and a stack of textbooks, surrounded by items for sale.	
How to Stop Itching from Diabetes. Diabetics frequently experience horrible itching. It is a common side effect of elevated blood glucose levels, which is the defining factor of diabetes. If you suffer from uncontrollable itchiness, this quick-fix article explains ways that you can soothe your irritated skin.	 A person scratching their skin due to diabetes-related itching.	
How to Buy an Out of State Car. You're browsing cars for sale online and you finally find your dream car at a fraction of the price for similar cars in your area. The only problem is the car is 4 states away. While buying a car out-of-state does require extra legwork, the process is straightforward.	 A laptop with a car listing website open, showing a car for sale in another state.	
How to Do Tile Painting. Many people believe that the only way to color a tile is to glaze it in a kiln, but you can actually paint tile yourself at home! If you do the right prep work, it's a quick, easy task that will let you add a decorative element to your bathroom, or even add a decorative element to your floors, counters, or mantel. [...]	 A hand painting a tile with a brush, while wearing protective gloves.	
How to Let Go of Something You Cannot Change. Admitting you didn't get what you wanted or that things didn't go your way can be difficult, yet moving from disappointment can be even more challenging. It's hard to let go of something outside of your control, know that it can be done. [...]	 A person looking at a sunset, hands held up in surrender.	
How to Be a Better Person. Life is a constant exercise in self-improvement, and while some of that focus lands squarely on becoming more educated or rising in the ranks of the workplace, sometimes we forget to improve how we treat ourselves and those around us. [...]	 A person looking at themselves in a mirror with a thoughtful expression.	

Figure 8: Example of natural prompts and visual prompts on the WikiHow domain.

763
764
765
766
767
768
769**B NL2VI Public Dataset**

To study the generalization of T2I methods, we benchmark their performance in settings with natural prompts in the Recipes and WikiHow domains. This dataset was designed to allow for the realistic, yet controllable, research of image generation methods conditioned in natural prompts.

770

B.1 Statistics771
772
773
774
775

The **NL2VI natural prompts and questions** dataset comprises 3000 curated natural prompts from the instructions domain, where the correct illustration of an action and correct composition of overlapping objects is critical.

776

B.2 Annotations777
778
779

In order to collect human data to use as a baseline, we created multiple annotation jobs using Amazon Mechanical Turk.

780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796

For the image-prompt alignment annotation task, the annotation job requires a qualification test. The qualification test is similar to the main annotation job, both require the annotation of the image quality and of the image-prompt alignment on a Likert scale from 1-5. Due to the subjectiveness of this task, we compare the annotator’s responses against the average of the responses of the authors. We give two points for exact matches and 1 point for off-by-one. Annotators only have access to the main annotation task if their score is greater than 80%. The main annotation task is shown in Figure B.2. Like on the qualification test, the users have detailed instructions (Figure 10) and many examples (Figure 11). To mitigate the randomness of stable diffusion we annotate 4 images per prompt for each method.

797
798
799
800
801

Similarly, to study the relevance of VQA errors and the accuracy of transformation from natural prompts to visual prompts we did two more annotation jobs, Figure 12 and Figure 13 illustrate the respective jobs.

802
803**C Ablation Study: Accuracy of VQA models**804
805
806
807
808
809

To assess the influence of the VQA errors in the final output we ask annotators to rank question/answers pairs from the VQA models. The results are shown in tables 8. The recipes domain is an easier domain for VQA models with an accuracy between 81%-84% depending on the model.

VQA	Acc. Recipes	Acc. WikiHow
BLIP	83.3%	77.7%
GIT	84.4%	78.2%
mPLUG	84.6%	76.3%
OFA	81.7%	75.2%
PaLI	84.1%	77.1%

Table 8: VQA accuracy in the Recipes and WikiHow domain.

WikiHow is a slightly harder domain with a VQA model accuracy between 75%-78%.

810
811**D Hallucinations, Open-World Assumption and Visual Common Sense**812
813

Prompts fed to image generation models contain *limited information*. It is not reasonable to assume that all the information that will be present in the final image was originally present in the prompt. Some common missing aspects in prompts are the image background, and object colours and texture. Examples can be seen in appendix in Figure 14. The lack of visual descriptions in the prompts, is filled in by model hallucinations, resulting in an image with more information than the original prompt. Hence, **hallucinations are needed** and unavoidable in the context of image generation, creating an **open-world setting** where multiple valid images can be generated from the original prompt. Verifying image consistency in an open-world setting is a challenging task. In the present work, instead of verifying the visual consistency in an open-world setting, we chose to verify consistency based on what is present in the visual prompt, thus adopting a conditional closed-world assumption during verification. LLMs are responsible to transform the open-world setting into a closed-world setting. A key concern is guaranteeing that these hallucinations are aligned with **visual common sense**, as some hallucinations are plausible while others invalidate the consistency of an image. In the Recipes domain, most generated images correctly depict the prompt, even when they are complex, such as in Figure 14 in the appendix. However, T2I algorithms sometimes lack common sense [White and Cotterell, 2022], especially when the meaning of the words is not clear, as in Figure 14. This issue falls out-of-scope of the present work, as we focus on verifying whether the elements present in the prompt show up in the final image.

814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848

BIBLIOGRAPHY

Please read the instructions carefully before working on this HIT.

[View instructions](#)

Pay attention to the following image: "A slice of quiche with spinach, cheese, and a flaky crust on a plate."



Q1. How do you rate the overall quality of this image?
 Very poor Poor Acceptable Good Very good

Q2. How well does the image match the description above?
 Does not match at all Has significant discrepancies
 Has several minor discrepancies Has few minor discrepancies
 Matches exactly

Check this box only when you feel you cannot answer Q2 for any reasons, such as abstract / non-visual descriptions or when there is missing information.
 Unable to answer

Offensive image

Submit

Figure 9: Image-prompt alignment annotation task.

Instructions

1. Read the description of the image carefully.
2. Check the content of the image carefully.
3. (Q1) Rate the overall quality of the image.
4. (Q2) Rate how well the image matches the description.

Q1. Rate the overall quality of this image.

When you answer Q1, please rate if the image looks like a real image. A high quality image is one that is hard to tell if it is real or AI-generated. Some AI-generated images may have artifacts such as irregular textures, shapes or noise as in the examples below. Rate such images as lower quality. **The description is irrelevant for this question.**

Q2. Rate how well the image matches the description.

When you answer Q2, please **judge only the matching of the description and the image**. An image that has some irregular textures and shapes but has all the important content, e.g. all the ingredients, should be given a high score.

Figure 10: Instructions given to annotators for the alignment qualification test and annotation task.

2. ALIGNING NATURAL PROMPTS AND IMAGE GENERATION

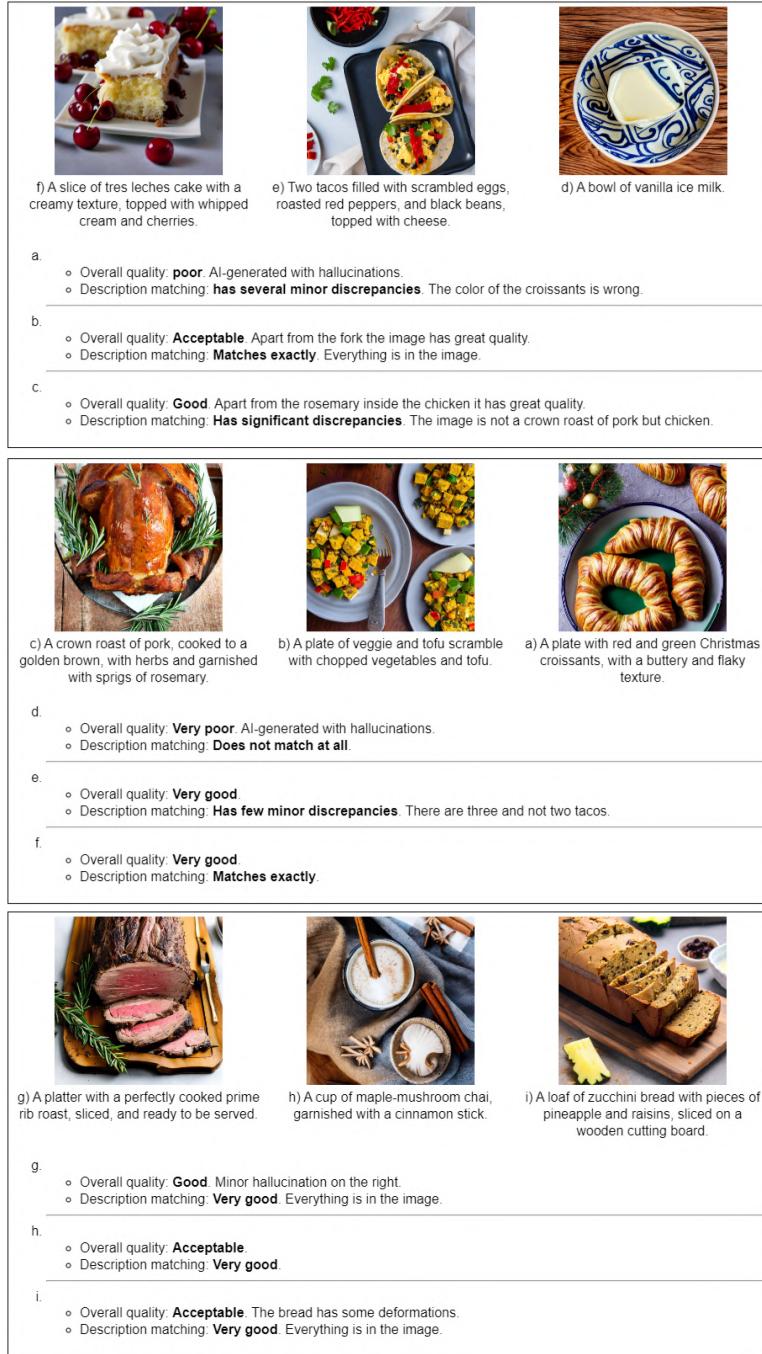


Figure 11: Examples given to annotators for the alignment qualification test and annotation task.

BIBLIOGRAPHY

Pay attention to the following image:



Annotate the question/answer pair with respect to the image

Question 1: Is there a bowl of food?
Answer: Yes
 Incorrect Correct Not related to the image

Question 2: What is in the bowl?
Answer: Shrimp
 Incorrect Correct Not related to the image

Question 3: Is there shrimp?
Answer: Yes
 Incorrect Correct Not related to the image

Question 4: What is on top of the shrimp and grits?
Answer: Chives
 Incorrect Correct Not related to the image

Question 5: Is there grits?
Answer: Yes
 Incorrect Correct Not related to the image

Submit

Figure 12: VQA error annotation task.

How aligned is prompt 2 in relation to prompt 1?

A prompt is considered aligned with another prompt if it **has all the visual information of the other prompt**.

Prompt 1: "Grilled Japanese Eggplant with Tahini Sauce. Tender long Japanese eggplants quickly grilled, served with sesame tahini sauce."
Prompt 2: "A plate of grilled japanese eggplants with a tahini sauce."

Not aligned (most of the visual information of prompt 1 is missing in prompt 2)
 Somewhat aligned (some visual information of prompt 1 missing in prompt 2)
 Aligned (all of the visual information of prompt 1 is in prompt 2)

Submit

Figure 13: Natural prompt to visual prompt error annotation task.



Figure 14: Images generated for "A bowl of ice-cream with a spoon". There is no information about the colour or type of ice-cream, and the model must fill in the missing properties. The same can be observed for the bowl itself. Another aspect is quantity, with the image on the left having two balls of ice-cream, which is information not present in the prompt, and the other images showing a single ball. We can also see how the spoons appear deformed.

.3 TWIZ 2022 - Report

TWIZ-v2: The Wizard of Multimodal Conversational-Stimulus

Rafael Ferreira, Diogo Tavares, Diogo Silva, Rodrigo Valério, João Bordalo,
Inês Simões, Vasco Ramos, David Semedo, Joao Magalhaes

NOVA University of Lisbon, NOVA LINCS

{rah.ferreira,dc.tavares,dmgc.silva,r.valerio
j.bordalo,ir.simoes,vcc.ramos}@campus.fct.unl.pt
{df.semedo,jm.magalhaes}@fct.unl.pt

Abstract

In this report, we describe the vision, challenges, and scientific contributions of the Task Wizard team, TWIZ, in the Alexa Prize TaskBot Challenge 2022 [1]. Our vision, is to build TWIZ bot as an *helpful, multimodal, knowledgeable, and engaging* assistant that can guide users towards the successful completion of complex manual tasks. To achieve this, we focus our efforts on three main research questions: (1) Humanly-Shaped Conversations, by providing information in a knowledgeable way; (2) Multimodal Stimulus, making use of various modalities including voice, images, and videos; and (3) Zero-shot Conversational Flows, to improve the robustness of the interaction to unseen scenarios. TWIZ is an assistant capable of supporting a wide range of tasks, with several innovative features such as creative cooking, video navigation through voice, and the robust TWIZ-LLM, a Large Language Model trained for dialoguing about complex manual tasks. Given ratings and feedback provided by users, we observed that TWIZ bot is an effective and robust system, capable of guiding users through tasks while providing several multimodal stimuli.

1 Introduction

Helping users in real-world manual tasks is a complex and challenging paradigm [11, 6, 1], where it is necessary to leverage multiple information sources, provide several multimodal stimuli, and be able to correctly ground the conversation in a helpful and robust manner. In this work, we build upon the success of TWIZ [10] and introduce new features along with expanding existing ones. This results in an assistant capable of guiding a user through a task while keeping it engaging and stimulating.

With the aim of advancing Multimodal Conversational AI, we explore three main research questions encompassing several contributions:

- **RQ1: Humanly-Shaped Conversations** - Conversations should be fun and rewarding, yet reach a successful completion. To achieve this, we take an all-encompassing approach and propose novel ways of finding/creating a task, getting task highlights in the overview, and handling the task execution dialogue with TWIZ-LLM, a Large Language Model (LLM) trained specifically for supporting a robust interaction in the TaskBot domain.
- **RQ2: Multimodal Conversational Stimulus** - Interactions between a user and an assistant should make use of various stimuli to keep the conversation engaging. To this end, we leverage both text and visual content. Particularly, we expand the curiosities paradigm by generating more fun and contextual curiosities. On the visual side, we explore several image-generation methods and propose ways to make them more consistent with the target

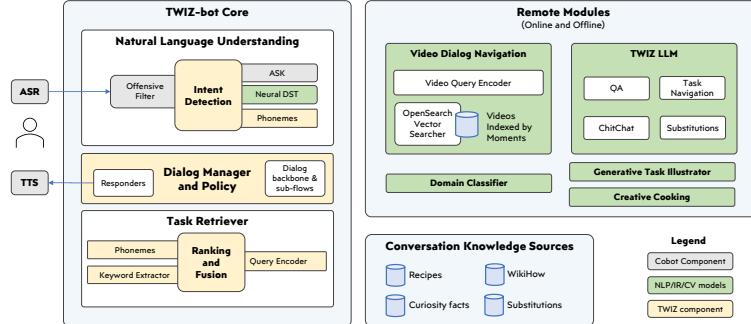


Figure 1: Architecture diagram of the TWIZ bot.

task. Finally, we present a novel Video Navigation feature, to allow for interactive video navigation by voice commands.

- **RQ3: Zero-shot Conversational Flows** - Given that users are unpredictable and may not follow the expected interaction, we need a robust dialogue framework. The goal is to steer the user through a pleasant and natural conversation while supporting these conversation detours. For this, we leveraged various LLMs and prompting strategies to extend TWIZ-LLM to converse about unseen topics with answers that are both in-scope and meaningful to the conversation.

2 TWIZ Bot Modular Architecture

An overview of our architecture is presented in Figure 1. It is built upon Amazon’s CoBot [17], a framework made available by Amazon for developing conversational agents. The agent operates within AWS Lambda, while all machine learning algorithms run on external modules, resulting in a highly efficient architecture with optimal performance. Leveraging the framework’s layered architecture, we adopt a shared database pattern using Amazon’s DynamoDB.

2.1 Intent Detection

Understanding the user’s intent is crucial to keeping a smooth flow in the dialogue interactions. Given its success, we use an approach similar to the previous year [10] by combining three methods: 1) phonemes-based matching; 2) rule-based corrections; and 3) a BERT-based model [34] for intent detection. These models work in an in-domain setting. However, they struggle to handle unseen or long-tail intents, requiring re-training to introduce new intents. To tackle this, we introduce a zero-shot intent detection method (Section 5).

2.2 Dialogue Manager

The dialogue manager keeps track of the current state and flow of the conversation. Our dialogue manager, adopts an event-driven state machine pattern. The progression through its different states is triggered by events associated with the detected intent of each user utterance, allowing for the context of a state to be used in order to provide relevant responses. With it, it is possible to keep track of the conversation’s progress and leverage state transitions to maintain a correct conversation flow.

2.2.1 Dialogue Backbone-Flow

The dialogue manager provides graceful guidance to the user through the task at hand, by means of a *backbone-flow*, which is a set of states that are necessary to go through in order to progress through a task. In TWIZ, the backbone states are as follows:

BIBLIOGRAPHY

1. **Greeting** - The starting state, in which TWIZ's functionalities are presented and task suggestions are made (e.g., Summer suggestions).
2. **Grounding** - State in which the selected suggestions or search results (Section 3.1) are shown to the user, so they can make a choice.
3. **Task Overview** - State after the selection of a particular task, where its overview – e.g. rating, duration, and others – is presented. This enables the user to either initiate the task or return to grounding in order to alter their selection. We also add a generated description to provide a brief task summary and further entice the user to start the task (Section 3.3).
4. **Task Execution** - The user can browse through the different steps of the task. Multimodal devices provide visual depictions of the steps, presented as either original or generated images (Section 4.2), or videos that the user can interact with through touch and voice (Section 4.3).
5. **Task Completed** - This is the last state of the dialogue, after the execution of the task. The agent offers more task suggestions, and the user has the option to either select one of these tasks, conduct a new search, or conclude the interaction.

2.2.2 Dialogue Sub-flows

Subsequently, the dialogue *backbone-flow* is enriched with *sub-flows*, comprised of a set of states associated with an additional feature, such as answering questions and engaging in chit-chat (Section 3.4), sharing curiosities (Section 4.4), and navigating through a video (Section 4.2).

The defined sub-flows are organized in a self-contained fashion, allowing for easy insertion, modification, or removal. The possibility to seamlessly integrate or remove sub-flows or states without causing disruptions to the rest of the state machine allows different developers to focus on various features without having to worry about conflicts with states, associated events, and response generators. The usage of different sub-flows also provides support for dialogue flexibility, allowing the user to stray from the main conversation in a contained way. By keeping track of the conversation's current state, the dialogue manager employs a stack-like checkpoint mechanism to offer seamless fallback options, guiding the user back to the *backbone-flow*.

While the primary objective of the dialogue manager is to assist the user in carrying out a task, it also accommodates the user's ability to switch tasks midway and provides the option to pause and resume the task at a later time, preserving all the progress made in that particular task.

3 RQ1: Humanly-shaped Conversations

Making sure that users select and successfully complete a task is one of the main objectives of a TaskBot. Therefore, the research goal is to deliver a conversation flow in an open, yet resilient manner. Our quest is to strike the delicate balance between the *task-knowledge* and a *humanly-shaped dialog* to ground the conversation on a complex manual task and guide the user through it.

3.1 Frictionless Task Suggestions and Search

TWIZ needs to swiftly ground the conversation in a frictionless way, by effectively and collaboratively finding the right task while also allowing for an exploration paradigm. Consequently, on the home page, we present a range of demonstrative tasks for users to select. These examples not only provide a clear overview of TWIZ's capabilities, but also offer users insights into the type of supported tasks. Moreover, from our analysis, a large portion of the time, users follow the suggestions provided, highlighting the value of these examples in guiding the users. Expanding upon these insights, we incorporated time-sensitive suggestions, to further improve the relevance of the tasks shown, as well as seasonal tasks relevant to the current time of the year. All the tasks suggested on the home page were manually selected to ensure a high-quality user experience.

The task search pipeline follows a multi-query, multi-ranking, and rank fusion approach. First, user queries are processed and the most relevant terms are extracted [10]. Secondly, we apply a ranking algorithm. Our approach involves carrying out both lexical (text) and semantic (embedding) searches on an OpenSearch index, with semantic searches conducted using embeddings generated by the MPNet model [31]. The results are then combined and re-ranked using the cosine similarity against

the user query, and an additional step of heuristic-based task quality parameters, such as the presence of a video or the number of ratings. This two-step approach enhances our ability to deliver the most relevant and quality-assured tasks.

3.2 Creative Cooking: What's in your fridge?

With regards to cooking, a helpful TaskBot should be able to guide users through cooking deadlocks (e.g. missing an ingredient, no matching recipe). To this end, we introduce a novel *Creative Cooking* feature that seeks to push the limits of users' creativity, TWIZ's knowledge, and cooking. In particular, it seeks to boost the users' creativity by letting them create their own unique and personalized tasks, adjusted to their preferences and available ingredients.

For example, if the user has in the fridge a particular set of ingredients (e.g. zucchini and eggplant) and a particular cooking style in mind (e.g. vegetarian) the user should be able to prompt TWIZ with a recipe that has these particular constraints, resulting in a recipe such as "grilled veggies". To do this, we first extract ingredients and cooking styles from the user utterances, using a rule-based approach against a set of curated tags. This way, we can guarantee that only valid tags are used. After this, we provide the user with a set of recipes from the API that satisfies the user's requests, and we add a recipe generated using an LLM considering the user's specifications. In particular, we use a Vicuna [4] model, which we prompt to generate a title, a list of ingredients, and a list of steps, given the aforementioned set of tags. An example of the creative cooking feature can be seen in this video¹. In Figure 7, we show an example of the creative cooking feature with a generated recipe, illustrated with our image generation methods (Section 4.2).

The creative cooking feature seamlessly integrates with all other TWIZ features (e.g. curiosities and image generation pipeline). As future work, we aim to create a user study for comparing manually created to generated recipes.

3.3 Task Overview: Task Promoter

When analyzing our interactions, we noticed that there is a direct correlation between users who start a task and higher ratings (Section 7). With this in mind, we looked for new ways to entice users to start a task during the *Task Overview* phase. Consequently, in the previous year, this led to the development of a 3D visual illustrator of the recipe [10] and manual templates to highlight certain features of a recipe [10], which resulted in low diversity responses. This year, we developed a Task Promoter, whose purpose is to generate, for any given task, a short and appealing description that highlights the best it has to offer.

3.3.1 Evaluation

We conducted a human evaluation, considering the recipes domain, and manually assessed for 100 recipes the preferred description: a prompted Vicuna-7B [4] or a fine-tuned GPT-2 [26] based model which we call RePro (for model training details refer to Appendix B.2). Additionally, we ask annotators to identify non-sensical descriptions and/or comprising ingredient hallucinations. To help assess the hallucinations, we provide annotators with both the recipe name and the ingredients, as in Table 7. We collect two annotations for each pair of descriptions.

Model	Win %	Hallucination %	Broken %	# Params
RePro (GPT-2)	20	57	10	770 M
Vicuna-7B	61	29	11	7000 M

Table 1: Results of manual evaluation of the recipe promoter. Ties are not included in *Win %*.

The results in Table 1 show that Vicuna in a zero-shot scenario resulted in preferred generations over a purpose-built smaller model. Although both models present very few non-sensical generations, a key differentiator is the hallucination of ingredients in the recipe, where over half of the RePro

¹https://www.youtube.com/playlist?list=PLC5saXed4eNtMDJPITQ0M4i0SGD83k_gy

BIBLIOGRAPHY

descriptions had at least one hallucinated ingredient. However, we noticed that almost all hallucinated and broken descriptions are easily identified, allowing them to be automatically removed.

Given the success of the Vicuna-7B approach, we expanded this feature to the WikiHow domain in a similar way using the title of the task. To improve model performance, as a next step, we plan to train a *RePro* model based on outputs from an LLM such as GPT-4.

3.4 Task Execution: TWIZ LLM

One of our team’s major focuses this year is to make TWIZ more natural and robust to user dialogues. Due to the emergence of effective LLM-based chatbots, users have higher expectations when interacting with TWIZ. To meet these expectations, we developed a task-oriented LLM-based approach, that seeks to support the *Task Execution* phase². We focused only on recipes, but plan to further expand it to DIY tasks.

3.4.1 Conversational Data Augmentation

During the task execution phase, we want to allow users to fully explore the recipe rather than just advance to the next step, to be able to ask questions about the cooking process, replace an ingredient, and get advice on the tricky parts. To generate data that accounts for all of these sub-flows in a robust manner, we devised a dialogue-generation pipeline that leverages data gathered during our participation in the first edition of the Alexa TaskBot Challenge [11]. We complement this data with several data augmentation techniques, described below, to increase the diversity of the generated dialogues.

Conversation Flow The basis of the dataset is the policy that dictates how dialogues are generated. For this, we extracted user patterns from user interactions and created a directed graph containing all the identified intents and the probabilities of the user transitioning between them, allowing us to accurately model real-user behavior when generating a new dialogue. The intents considered are classified using a combination of rule-based heuristics and a customized Transformer-based intent classifier [10]. The complete list of intents considered and their description can be seen in Table 2. To improve dialogue diversity, we manually increased the transition probabilities of less common intents such as Ingredient Replacement and Questions. While, by doing this, we are making the generated dialogues policy diverge from the real-user policy, we believe the added frequency of such intents can greatly improve the ability of models trained on this data to attend to this type of intents.

As this dataset focuses on task execution, we do not consider the task selection and search phases and simulate the generated dialogues starting when the user asks the assistant to start an already selected task. An example of a generated dialogue can be seen in Appendix Table 8.

Task Selection As, in each dialogue, the user focuses on completing a single recipe. We used Amazon’s provided recipes dataset and extracted 1000 recipes that had between 5 and 10 steps, and a total word count of no more than 350. This helps ensure that the recipes are long enough for a meaningful dialogue, with opportunities to ask questions, while also avoiding overly long recipes that could lead to repetitive or noisy dialogues.

User Queries For the data to closely mimic real user behavior, we used authentic user requests. To achieve this, we collected all user utterances classified for each considered intent and their absolute frequency. To clean up classification errors, we manually reviewed each of the most common user requests for each intent and removed any utterances that did not match the intent. When generating a new dialogue turn, the user utterance is selected from the list of utterances for the current intent using a random weighted selection, where the weight of each utterance is its absolute frequency, ensuring that more common utterances are more likely to be selected. However, some intents are extremely contextual, meaning that their associated user requests can have vastly different answers based on the relevant recipe (e.g., questions regarding a specific recipe step or ingredient replacement). For these cases, we utilize zero-shot LLM prompting and template-based approaches to produce natural-sounding and context-relevant user utterances. More information about this can be found in Appendix C.1.

²When the user has already selected and started a task.

	Intent	Description	Example	# ex.
Navigational	Next Step	Go to the next step of the task.	Next Step.	169
	Repeat	Repeats the last system utterance.	Repeat that.	76
	Stop	Ends the interaction.	Stop	73
	Yes	Confirmation (acts as Next Step).	Sure.	52
	Previous Step	Go to the previous step of the task.	Previous Step.	21
	Resume	Repeats the current step.	Resume.	18
Question	Get Curiosities	Asks for a fun fact.	Can you tell me a fun fact?	13
	Ingredients Replacement	Asks for a replacement.	I do not have sugar.	9
	Definition Question	Definition type question	What is a spatula?	4
	Question	Task specific question.	How much salt do I need?	-
Other	Fallback	Intent is not recognized.	Find a restaurant near me.	2618
	Sensitive	Mentions a sensitive/dangerous topic.	How do you make a nuke?	235
	Chit-Chat	Chit-chat utterances.	How are you today?	131
	Search	Ask for another task.	How to change a tire?	108
	More Detail	Ask for more details about a step.	More details, please.	36

Table 2: Intents list, description, example (manually-crafted) utterances, and the count of unique utterances available.

System Responses The system responses need to be diverse but also accurate and contextual, w.r.t. the recipe and the user request. To achieve this, based on the user intent, we use templates, knowledge bases, and LLM-prompts:

- **Navigational** - For these cases the system response consists only of step text.
- **Definition Question** - We query a dictionary³ for the appropriate meaning of a given concept.
- **General Question** - Given a recipe, generate QA-pairs by prompting OpenAI’s *text-davinci-003*. This way both the questions and system answers are contextual.
- **Ingredient Replacement** - Template responses, filled in using a knowledge base⁴ of valid ingredient replacements (e.g., “{Ingredient A} can be replaced with {Ingredient B} in this recipe”).
- **Fun facts** - We used OpenAI’s *text-davinci-003* to generate curiosities for each recipe step by providing the step text and a relevant Wikipedia paragraph⁵ and prompting the model to generate a fun fact relevant to that step.

For all other cases, a template-based approach was used. The templates were written by hand with up to 5 examples per case. In these cases, to ensure inter-dialogue response diversity, the selected response was randomly sampled from all templates not used in the past 5 dialogue turns. Additionally, every template is written in several different tones of voice, to allow for more diversity amongst the generated dialogues.

Dialogue Generation Pipeline The complete generation pipeline follows the following three steps when generating a new dialogue turn:

1. **Determine User Intent** - Based on the previous turn and on the extracted policy, sample the next user intent.
2. **Retrieve User Utterance** - Based on the selected intent and current recipe step, retrieve a candidate user utterance.
3. **Produce System Response** - Based on the selected intent, user utterance, and recipe, select the appropriate system response.

These steps are repeated until the user reaches the last step of the recipe or a *Stop* intent is selected.

³<https://github.com/wordset/wordset-dictionary>

⁴<https://foodsubs.com/>

⁵using <https://neuml.github.io/txtai/>

BIBLIOGRAPHY

3.4.2 Base LLM Models

We are currently using two models: **Vicuna-7B** [4], a LLaMA-based [36] model fine-tuned on conversational data, and **OPT-1.3B** [42], to understand how different model sizes and architectures behave on this task.

For the model input, there are four key pieces of information that the model needs to be able to attend to (training prompts are shown in Appendix C.2):

- **Tone of Voice:** The tone that should be used by the system during the dialogue. It can be neutral, somewhat polite, polite, or very polite. This allows training models with controllable tone of voice in their responses.
- **Recipe Text:** The recipe title, followed by its steps.
- **Current Step:** The recipe step that the user is currently on. This helps in navigational requests and in keeping the answers grounded.
- **Dialogue context:** The previous t turns + the *current user request*. For our evaluation, we use a $t = 1$.

3.4.3 Experimental Setup

Model Details We trained Vicuna-7B and OPT-1.3B on the same data. Vicuna was trained for 1 epoch, whereas OPT was trained for 10. In addition to SFT training, we also trained OPT under an RLHF paradigm [32, 23], to understand if the models would benefit from this approach.

Dataset We generated 10k dialogues with an 80/10/10 split. To create the negative system responses for RLHF, we employ different methods based on the user’s intent:

- **Navigational** - we introduce a sentence from the previous/next step to simulate model copying mistakes.
- **Definition Questions** - we select the definition ranked lowest using a bag-of-words approach.
- **Ingredient Replacement** - we randomly choose an ingredient from a list to replace the current one.
- **Get a Curiosity** - we retrieve a curiosity with lower similarity using a bag-of-words approach.
- **Sensitive** - provides an answer to a sensitive request given by an uncensored LLM.

If no change is applied, and for all other intents, we introduce response perturbation and grammatical errors.

3.4.4 Automatic Evaluation

	Test Set %	OPT-SFT		OPT-RL		Vicuna	
		METEOR	BScore-F1	METEOR	BScore-F1	METEOR	BScore-F1
Navigational	65.36%	64.39	81.73	59.79	79.90	98.36	99.29
Get Curiosities	4.83%	14.94	57.44	14.52	57.11	18.29	61.96
Ingredients Replacement	1.11%	77.37	87.59	32.92	68.84	79.91	89.19
Definition Question	4.07%	83.76	91.52	83.09	91.23	81.25	91.23
Question	9.41%	54.01	78.52	53.26	78.20	61.80	82.92
Sensitive	4.07%	81.75	90.83	91.00	95.30	68.30	85.52
Other*	11.15%	68.97	83.57	42.66	73.06	84.78	92.65

Table 3: Per intent results for all three trained models. *Sensitive intent was separated from the Other intent group due to outlier results.

To measure model performance, we evaluate the model on all turns of the test set dialogues. For automatic metrics, we considered METEOR [3] and BERT-Score-F1 [43] to measure textual overlap and semantic similarity, respectively. In Table 3, we provide a detailed analysis of the models’ performance, by separating turns with the “more static” intents, which require copying from the task

information (e.g., *Navigational*) or returning a default response (e.g., *Other*), from the turns that require access to some external knowledge.

These early results show that the Vicuna-based model is capable of handling *Navigational* intents. It also outperforms both OPT-based models for almost all intents. For Vicuna, the outliers are the *Curiosity* requests. These answers are difficult to evaluate using automatic metrics since there can be various correct answers given a user's request. RL on the OPT model seems to have little impact in most cases. However, it does improve *Sensitive* requests, while having a negative impact on both *Ingredient Replacement* and on the *Other* intent group, indicating that the model might confuse these requests as dangerous tasks. We also noted that when handling sensitive requests, both OPT and Vicuna recognized the nature of the request and did not respond inappropriately.

3.4.5 Human Evaluation

To more accurately measure the models' performance, we conducted a human evaluation for Ingredient Replacement, Questions, and Curiosity user requests. We focused on these request types due to their reliance on external knowledge and the possibility of multiple correct answers. For the 3 cases, we selected 50, 100, and 30 examples from the test set, respectively. The annotators were asked to rate each one on a scale of 0 (wrong/irrelevant) to 2 (accurate/very relevant). For both ingredient replacement and questions, the criteria used was accuracy, whereas for curiosity requests, the relevancy of the generated curiosity w.r.t. the current recipe step was annotated. Annotators were instructed to rate 0 in cases where the generated utterance is incoherent or a clear hallucination.

Table 4 shows the results of the human evaluation. Vicuna outperforms both OPT models. Using RL on OPT had a negative impact on model performance on two tasks, and improved for *Question* requests. For *Curiosity* requests, OPT produces fairly relevant curiosities, on par with Vicuna, but performance worsens with the RL approach.

Model	OPT-SFT	OPT-RL	Vicuna
Question	1.30	1.48	1.77
Ingredient Replacement	0.84	0.76	1.48
Get Curiosities	1.23	0.90	1.30

Table 4: Human evaluation results of TWIZ LLM models on the Question, Ingredient Replacement, and Fun Fact intents, on a 0 to 2 scale.

In a later analysis of the reward model used with OPT-RL, we noted that the reward model did not learn to model the preferences appropriately. We believe this is due to the low quality of the preference data. However, these results, along with the results observed in automatic evaluation, show that RLHF has the potential to provide meaningful performance improvements but lacks consistency. Thus, future work should focus on improving the quality of the generated preference data, for example, using more diverse LLM prompting methods.

To conclude, TWIZ-LLM presents a step forward in having a model capable of guiding a user through the execution of a task while providing natural and helpful responses.

4 RQ2: Multimodal Conversation-Stimulus

Conversation stimulus is what advances a conversation, i.e., the user's desire to obtain the final outcome of the task and the road that leads to it. TWIZ provides a number of linguistic, visual, and cognitive stimuli to keep conversations natural and engaging.

4.1 UI and APL Templates

The user interface significantly influences the perceptions and experiences of the users. As such, the quality and functionality of the interface are crucial. To address this, we developed a new user interface that is more capable of showcasing TWIZ's features.

Our design philosophy leans towards minimalism, emphasizing a clean theme to promote clarity and ease of use. This approach encourages a seamless user experience, contributing to more efficient

BIBLIOGRAPHY

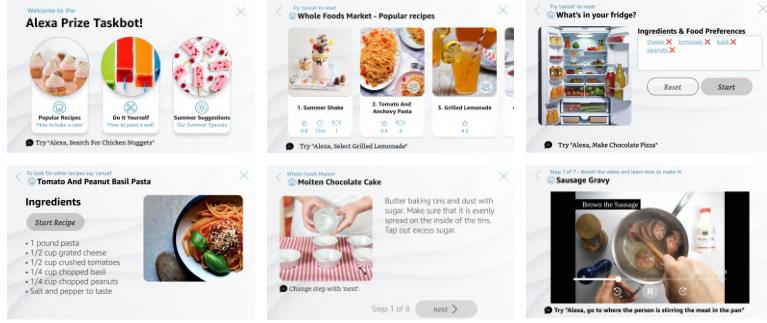


Figure 2: TWIZ’s APL screens: *welcome screen*, *search results*, “*What’s in my fridge?*”, *ingredients list*, *task step*, and *video dialogue*.

interactions. Additionally, on each screen, we provide a rotating set of contextual tips that introduce users to TWIZ’s wide range of features. The complete set of screens can be found in Figure 2.

4.2 Task Visual Enrichment and Dialogue

Task illustrations are fundamental for rich and engaging visual-stimuli interactions. Nonetheless, a significant number of tasks either lack accompanying illustrations or have low-quality ones. Moreover, creative cooking (Section 3.2), offers the possibility to compose a recipe from scratch, requiring a strategy to illustrate it. In this section, we describe how image generation models [29], can provide a solution to generate task and step-specific illustrations.

4.2.1 Generative Task Illustration

Illustrating a task with text-to-image algorithms can be challenging, and requires extra care to ensure the generated images are grounded on the task and have good quality. Our implementation can be divided into three main steps:

1. **Image generation** is a non-trivial task, as it involves generating images from a task that might not be visual in nature. We propose a new method [37], which uses an LLM to convert the task into a visual prompt with high consistency that can then be used in text-to-image models. We generate multiple images for each task using the Stable Diffusion XL model [24].
2. The **image score** is given by an alignment score. This alignment score is calculated by the NL2VI metric [37], which measures the alignment of the image to the prompt. The alignment is calculated using VQA algorithms [20, 19, 41, 40]. The questions for VQA are generated using an LLM with in-context learning and are filtered using question answering [16] and NLI [22] models. This score ensures that the image is grounded on the task. The results of Table 5 show that our method achieves better results than the TIFA [14] baseline.
3. Lastly, we **rank the generated images**, as well as the original image, according to their score and choose the one with the highest score to illustrate the task [37].

4.2.2 Generative Recipe Step Illustration

After having a method to illustrate the main tasks’ content, we now want to illustrate every step of a task using an image-generation approach, with visual continuity guarantees, which we refer to as *Coherent Step Illustration*. As established in Section 4.2.1, generated images must be consistent with the prompts, ensuring the desired elements are present in the image. This is important in task step illustration, as the illustrations must be faithful to the step to steer the user in the right direction.

	Equality	NLI	BERT-Score
TIFA [14]			
Recipes	64.7	68.9	72.0
WikiHow	56.1	61.6	64.2
NL2VI [37]			
Recipes	73.7	76.2	79.8
WikiHow	73.1	75.8	73.8

Table 5: Comparison of the image alignment score between our method NL2VI [37] and the previous SOTA TIFA [14].

The main focus of this method is the sequential nature of step illustrations. Unlike *Task Illustration*, which is independent, *Step Illustrations* have dependencies regarding the previous steps.

Our method takes as input a sequence of steps $\{S_1, S_2, \dots, S_n\}$ and generates each image, I_n , which illustrates step S_n by considering all previous steps. The motivation is to increase the coherence of the sequence of images. The method comprises three phases:

1. **Sequence of Captions.** When generating an illustration for a recipe step, we want the illustration of step S_n to consider the previously generated images more directly. Hence, to address this limitation, we propose to consider the captions [8] of the previously generated images, instead of the text from previous steps. With these captions, our updated input can be written as, $\{C_1, C_2, \dots, S_n\} \rightarrow I_n$.
2. **Sequence-to-Prompt LLM.** To transform the sequence $\{C_1, C_2, \dots, S_n\}$ into an appropriate image generation prompt, we train an LLM with sequences of (captions+step) and the target prompt.
3. **Sequence-to-Prompt Generation.** After training, we can generate the prompts by iteratively prompting the LLM with the current step and the previous captions. These generated prompts contain enough information to increase the coherence of the generated images, with respect to the previous generations.

Given the nature of the task, we only apply this method to the recipes domain. In future work, we will conduct a user study to measure the performance of this approach.

4.3 Video Moment Retrieval

In order to facilitate the interaction with the vast amount of information present in a video, we developed a method to enable users to control videos using voice commands, when the video is in full-screen mode. This allows users to interact with videos effortlessly and intuitively, enhancing their overall viewing experience, as in Figure 4. An example interaction can be seen in this video⁶.

As a first step, we extracted keyframes from all videos. For each keyframe, we generated a caption, using InstructBLIP [8]. Then we used the image and caption of each frame to extract embeddings using CLIP [25]. All of this information is stored in an OpenSearch index, along with the task metadata. The search is performed using the user query (e.g. “When did the chef mix in the flour?”), and uses both a full-text and embedding-based (text and image) search. The result thus has three sets of frames, one for each type of embedding (text and image) and one for plain text. We then perform re-ranking of the potential frames based on rank fusion. After

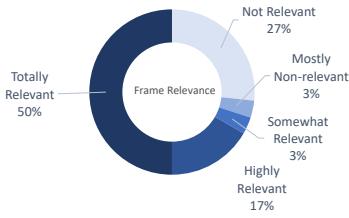


Figure 3: Video moment retrieval results.

⁶https://www.youtube.com/playlist?list=PLC5saXed4eNsebM8C4W5S_BQ9ADEgwH57

BIBLIOGRAPHY



Figure 4: TWIZ users can ask a question about a video and navigate to the correct video moment that answers the question. An example interaction can be seen in this demo video link.

selecting the winning frame, we perform a seek in the video based on the timestamp of the frame on the top of the rank.

To evaluate the proposed method, we ran a test of 30 queries over 10 videos. For each user query, a video frame is returned. We judged the frame’s relevance on a scale from 1 to 5, with 1 indicating a non-relevant frame and 5 indicating a totally relevant frame. As shown in Figure 3, 27% of the results were not relevant to the query, while 50% were considered totally relevant. Given these results, it is important to study ways to reduce the occurrences of lower-quality results by making better use of video information and improving the ranking algorithm.

4.4 Fact-Grounded Curiosity Generation

Curiosity-stimuli, i.e. *fun-facts*, turn conversations into an educational and memorable experience. In the first edition of the TaskBot Challenge, we contextually enriched dialogues with a manually curated set of curiosities [10]. Subsequently, we performed an A/B test with over 1000 conversations, which showed that curiosities increase user engagement and can also provide an average rating improvement [39].

Given these encouraging results, this year we expand this paradigm by using LLMs to generate more contextual and interesting curiosities, removing the bottleneck of manually creating curiosities. In particular, we use GPT-4 as the backbone for our curiosity generation model. Given that LLMs have a tendency to hallucinate [15], we add a relevant Wikipedia passage as context to ground the generation, alongside the task title and current step. We conduct a user study comparing the manually annotated curiosities with the generated ones, according to the following aspects: (1) **relevance** to the step, (2) **truthfulness** according to the provided information, and (3) **fun factor**. These aspects are measured on a scale of 0 (low) to 2 (high). Additionally, we also ask annotators to choose a winner between the two methods. In total, we collect 150 annotations with 3 annotations per comparison. These results are shown in Table 6.

	Win %	Relevance	Truthfulness	Fun Factor
Manually Curated	33.33%	0.71	1.78	0.80
Generated	66.67%	1.56	1.50	0.90

Table 6: Comparison between manually curated curiosities and curiosities generated by GPT-4.

The results show that the annotators have a clear preference for the generated curiosities, despite their truthfulness value being lower than the manually curated ones. Regarding the fun factor, none of the methods exhibits a high score. This suggests that the concept of “fun” in this domain requires further exploration. Nevertheless, the generated curiosities reduce the bottleneck of creating a manual set of curiosities and tend to be more fun and considerably more relevant.

5 RQ3: Zero-Shot Conversation Ramblings

Often, users elaborate their responses or ask for side information, which conventional systems, with a strict dialogue flow, fail to respond. TWIZ aims to improve the user experience by adapting *on-the-fly to conversation ramblings* introduced by the user, through zero-shot approaches.

Zero-shot DST as Reading Comprehension A rapidly evolving dialogue system such as TWIZ requires DST modules that easily support the inclusion of new slots and intents, even with little to no available data. To achieve this, we follow *Namazifar et al.* [21] to cast the typical zero-shot DST task as a reading comprehension one. To train our models, we require QA examples. These are automatically extracted from unlabeled TWIZ data, in a self-supervised setting [35]. Furthermore, during inference time, we require one question per slot and intent. These are created by prompting an LLM (*text-davinci-003*) to generate one question per slot. This contrasts with typical approaches, where questions are either derived from templates or manually written [21]. We find that the strategy of pre-training the model using in-domain dialogues and LLM-based questions significantly improves performance, as detailed in our work [35].

This model is used when new slots and intents are required to support new features, and while the team is gathering relevant, in-domain, data. When enough data is available, we pivot towards full-shot approaches (Section 2.1).

Zero-Shot Responses with TWIZ-LLM Generative Large Language Models trained with Human Feedback [23] possess a great ability to answer a wide range of user ramblings. After the release of ChatGPT and its open-source competitors [4, 33], it became clear that LLMs wrapped in strong guard rails [23, 2] can respond to chit-chat, question-answering, and fallback intents in an appropriate and pleasant way. In this context, we use the responses given by TWIZ-LLM (Section 3.4) to respond to many of these requests.

6 Trustworthy TaskBot Generative Vision and Language

Given the recent advancements in using generative methods for both text [4, 23, 33] and image generation [30], a recent and very important research direction is how to guarantee trustworthy and consistent generations [13, 37]. In our work, we make use of various generative methods and we always ensure that they are as trustworthy as possible through the implementation of several guardrails and verification methods.

In text generation tasks that are grounded in some input constraint, such as the *Creative Cooking* that uses only valid tags for generating new recipes, the *Task Promoter*, and the *Curiosities* which receive a grounding Task/Wikipedia passage, are methods that can be further checked by a verification method such as *True* [12] and *Q2* [13], which use question generation/answering methods to ensure that the output is factually consistent with the input.

In the case of images, we take specific care in generating images that are *Consistent* with the prompt through the use of a novel verification pipeline NL2VI [37] and further expand this work to generate images that are *Coherent* along a sequence of steps. These methods are based on the creation of a *visual prompt* which is more suitable for image generation, followed by a T2I pipeline where images are verified through the use of VQA algorithms.

We believe that these methods are a step forward in diminishing the problems found in generative methods with the aim of providing more relevant and accurate information to the user.

7 User Interaction Analysis

User Behavior Understanding user's interactions allows us to discover possible interaction bottlenecks or new conversational paths.

Looking at device type, we see that on average, ratings from headless interactions are higher than multimodal ones 3.47 vs. 3.27, respectively. The use of multimodal devices also led to the increase of users using the on-screen buttons, rather than their voice, with over 35% of the interaction turns being taps on the screen. Another typical user behavior is asking for commands that our bot cannot

BIBLIOGRAPHY

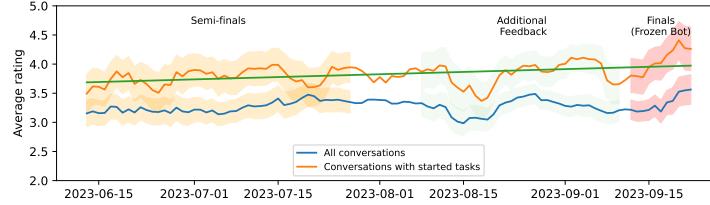


Figure 5: TWIZ’s 7-Day average rating since the semi-finals for conversations with at least 3 turns.

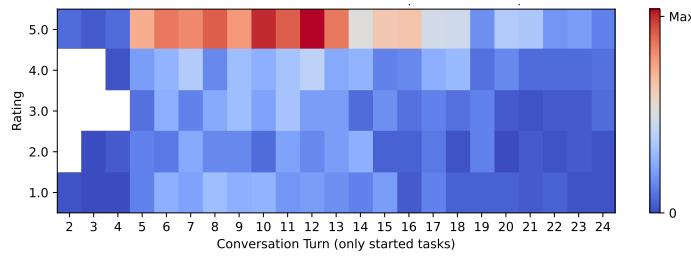


Figure 6: Conversations’ ratings per turn when a task is started.

comply with, such as playing YouTube videos or music, which happens in 8.4% of interactions. We also see that non-rated conversations tend to be very short, with an average of 2.35 vs. 7.46 turns for rated ones. When focusing on ratings and task execution, we see that 32% of users who rate, start a task. From these, 42.8% start a recipe while the remaining 57.2% do a WikiHow task. Additionally, they also rate significantly higher than those who don’t start (3.86 vs. 3.30, respectively). Looking at the task type, users who start a recipe give an average rating of 4.07 whereas users who start a WikiHow have an average rating of 3.70.

Ratings Progression In Figure 5, we present the average 7-Day rating since the semi-finals. There are two major observations: 1) *an increase in the user ratings throughout this period*, and 2) *TWIZ’s high effectiveness in conversations in which users start a task*. Moreover, a positive trend in the ratings can be observed from the start of the semi-finals. In some periods, ratings variability is high, specially in the middle of each stage’s period, which we primarily attribute to intense system modifications.

Ratings per Conversation Length Finally, Figure 6, depicts conversations rating per turn and provides further evidence of TWIZ bot’s effectiveness in delivering the Alexa Prize TaskBot Challenge’s main objective - guiding an user through a complex task. Namely, it can be seen that a large portion of conversations in which users start a task, get the maximum rating (top line). These results highlight TWIZ bot’s consistency and quality during a comprehensive time frame and for users who are using the bot in its full capacity.

8 Conclusions and Future Work

In this paper, we summarized TWIZ’s work on the second edition of the Alexa TaskBot Challenge. We built upon the strong foundation of the previous year and provided several new contributions:

- We focused on ensuring an engaging interaction by supporting *natural and knowledgeable* dialogues. To achieve this, we introduced the “Creative Cooking” feature, empowering users to craft their own custom recipes. Adding to this, we promote tasks in a positive way

to increase user engagement, and provide navigational and contextual responses to user requests using TWIZ-LLM, an LLM trained specifically for the TaskBot domain.

- We provided various *multimodal stimuli* to the user. We expanded the curiosities paradigm and focused on improving multimodal interactions by generating task illustrations. In this last point, we proposed new methods to illustrate the task as well as its steps, with the added care to make illustrations consistent throughout the task. Still, in the multimodal stimuli, we also developed a video moment retrieval pipeline allowing video navigation by voice.
- Finally, we created a *robust* system to allow for more *flexible* user interactions and analyzed the users' interactions with the system resulting in several relevant insights.

In future work, we aim to continue exploring the use of LLMs and expanding their utilization to all stages of the dialogue, as well as introducing new multimodal stimuli, while assessing their impact on the conversation.

References

- [1] Eugene Agichtein, Michael Johnston, Anna Gottardi, Cris Flagg, Lavina Vaz, Hangjie Shi, Desheng Zhang, Leslie Ball, Shaohua Liu, Luke Dai, Daniel Pressel, Prasoon Goyal, Lucy Hu, Osman Ipek, Sattvik Sahai, Yao Lu, Yang Liu, Dilek Hakkani-Tür, Shui Hu, Heather Rocker, James Jeun, Akshaya Iyengar, Arindam Mandal, Saar Kuзи, Nikhita Vedula, Oleg Rokhlenko, Giuseppe Castellucci, Jason Ingyu Choi, Kate Bland, , Yoelle Maarek, and Reza Ghanadan. 2023. Alexa, Let's work together: Introducing the Second Alexa Prize TaskBot Challenge. In *Alexa Prize TaskBot Challenge 2 Proceedings*. <https://www.amazon.science/alexaprize/proceedings/alexaprize-taskbot-challenge>
- [2] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadvath, Jackson Kernion, Tom Conery, Sheer El Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Chris Olah, Benjamin Mann, and Jared Kaplan. 2022. Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback. *CoRR* abs/2204.05862 (2022). <https://doi.org/10.48550/arXiv.2204.05862> arXiv:2204.05862
- [3] Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*. Association for Computational Linguistics, Ann Arbor, Michigan, 65–72. <https://aclanthology.org/W05-0909>
- [4] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality. <https://lmsys.org/blog/2023-03-30-vicuna/>
- [5] Jason Ingyu Choi, Ali Ahmadvand, and Eugene Agichtein. 2019. Offline and Online Satisfaction Prediction in Open-Domain Conversational Systems. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*. ACM, 1281–1290. <https://doi.org/10.1145/3357384.3358047>
- [6] Jason Ingyu Choi, Saar Kuзи, Nikhita Vedula, Jie Zhao, Giuseppe Castellucci, Marcus Collins, Shervin Malmasi, Oleg Rokhlenko, and Eugene Agichtein. 2022. Wizard of Tasks: A Novel Conversational Dataset for Solving Real-World Tasks in Conversational Settings. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*. International Committee on Computational Linguistics, 3514–3529. <https://aclanthology.org/2022.coling-1.310>
- [7] Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free Dolly: Introducing the World's First

BIBLIOGRAPHY

Truly Open Instruction-Tuned LLM. <https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-lm>

- [8] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Jungi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven C. H. Hoi. 2023. InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning. *CoRR* abs/2305.06500 (2023). <https://doi.org/10.48550/arXiv.2305.06500> arXiv:2305.06500
- [9] Rafael Ferreira, David Semedo, and João Magalhães. 2023. Rating Prediction in Conversational Task Assistants with Behavioral and Conversational-Flow Features. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*. ACM, 2314–2318. <https://doi.org/10.1145/3539618.3592048>
- [10] Rafael Ferreira, Diogo Silva, Diogo Tavares, Frederico Vicente, Mariana Bonito, Gustavo Goncalves, Rui Margarido, Paula Figueiredo, Helder Rodrigues, David Semedo, and Joao Magalhaes. 2022. TWIZ: A conversational Task Wizard with multimodal curiosity-exploration. In *Alexa Prize TaskBot Challenge Proceedings*. <https://www.amazon.science/alexaprize/proceedings/twiz-a-conversational-task-wizard-with-multimodal-curiosity-exploration>
- [11] Anna Gottardi, Osman Ipek, Giuseppe Castellucci, Shui Hu, Lavina Vaz, Yao Lu, Anju Khatri, Anjali Chadha, Desheng Zhang, Sattvik Sahai, Prerna Dwivedi, Hangjie Shi, Lucy Hu, Andy Huang, Luke Dai, Bofei Yang, Varun Soman, Pankaj Rajan, Ron Rezac, Michael Johnston, Savanna Stiff, Leslie Ball, David Carmel, Yang Liu, Dilek Hakkani-Tur, Oleg Rokhlenko, Kate Bland, Eugene Agichtein, Reza Ghanadan, and Yoelle Maarek. 2022. Alexa, let's work together: Introducing the first Alexa Prize TaskBot Challenge on conversational task assistance. In *Alexa Prize TaskBot Challenge Proceedings*. <https://www.amazon.science/publications/alexas-lets-work-together-introducing-the-first-alexaprize-taskbot-challenge-on-conversational-task-assistance>
- [12] Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. 2022. TRUE: Re-evaluating Factual Consistency Evaluation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*. Association for Computational Linguistics, 3905–3920. <https://doi.org/10.18653/v1/2022.naacl-main.287>
- [13] Or Honovich, Leshem Choshen, Roee Aharoni, Ella Neeman, Idan Szpektor, and Omri Abend. 2021. Q^2 : Evaluating Factual Consistency in Knowledge-Grounded Dialogues via Question Generation and Question Answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 7856–7870. <https://doi.org/10.18653/v1/2021.emnlp-main.619>
- [14] Yushi Hu, Benlin Liu, Jungo Kasai, Yizhong Wang, Mari Ostendorf, Ranjay Krishna, and Noah A. Smith. 2023. TIFA: Accurate and Interpretable Text-to-Image Faithfulness Evaluation with Question Answering. *CoRR* abs/2303.11897 (2023). <https://doi.org/10.48550/arXiv.2303.11897> arXiv:2303.11897
- [15] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of Hallucination in Natural Language Generation. *ACM Comput. Surv.* 55, 12 (2023), 248:1–248:38. <https://doi.org/10.1145/3571730>
- [16] Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing Format Boundaries with a Single QA System. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics, Online, 1896–1907. <https://doi.org/10.18653/v1/2020.findings-emnlp.171>

- [17] Chandra Khatri, Anu Venkatesh, Behnam Hedayatnia, Raefer Gabriel, Ashwin Ram, and Rohit Prasad. 2018. Alexa Prize — State of the Art in Conversational AI. *AI Magazine* 39, 3 (Sep. 2018), 40–55. <https://doi.org/10.1609/aimag.v39i3.2810>
- [18] Cat P. Le, Luke Dai, Michael Johnston, Yang Liu, Marilyn A. Walker, and Reza Ghanadan. 2023. Improving Open-Domain Dialogue Evaluation with a Causal Inference Model. *CoRR* abs/2301.13372 (2023). <https://doi.org/10.48550/arXiv.2301.13372> arXiv:2301.13372
- [19] Chenliang Li, Haiyang Xu, Junfeng Tian, Wei Wang, Ming Yan, Bin Bi, Jiabo Ye, He Chen, Guohai Xu, Zheng Cao, Ji Zhang, Songfang Huang, Fei Huang, Jingren Zhou, and Lu Si. 2022. mPLUG: Effective and Efficient Vision-Language Learning by Cross-modal Skip-connections. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*. Association for Computational Linguistics, 7241–7259. <https://doi.org/10.18653/v1/2022.emnlp-main.488>
- [20] Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. 2022. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, 12888–12900. <https://proceedings.mlr.press/v162/li22n.html>
- [21] Mahdi Namazifar, Alexandros Papangelis, Gökhan Tür, and Dilek Hakkani-Tür. 2021. Language Model is all You Need: Natural Language Understanding as Question Answering. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*. IEEE, 7803–7807. <https://doi.org/10.1109/ICASSP39728.2021.9413810>
- [22] Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A New Benchmark for Natural Language Understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (Eds.). Association for Computational Linguistics, 4885–4901. <https://doi.org/10.18653/v1/2020.acl-main.441>
- [23] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*. http://papers.nips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html
- [24] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. 2023. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. *CoRR* abs/2307.01952 (2023). <https://doi.org/10.48550/arXiv.2307.01952> arXiv:2307.01952
- [25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, 8748–8763. <http://proceedings.mlr.press/v139/radford21a.html>
- [26] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019).
- [27] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. *CoRR* abs/2305.18290 (2023). <https://doi.org/10.48550/arXiv.2305.18290> arXiv:2305.18290

BIBLIOGRAPHY

- [28] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 3980–3990. <https://doi.org/10.18653/v1/D19-1410>
- [29] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 10674–10685. <https://doi.org/10.1109/CVPR52688.2022.01042>
- [30] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 10674–10685. <https://doi.org/10.1109/CVPR52688.2022.01042>
- [31] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and Permuted Pre-training for Language Understanding. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. <https://proceedings.neurips.cc/paper/2020/hash/c3a690be93aa602ee2dc0ccab5b7b67e-Abstract.html>
- [32] Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. 2020. Learning to summarize from human feedback. *CoRR* abs/2009.01325 (2020). arXiv:2009.01325 <https://arxiv.org/abs/2009.01325>
- [33] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.
- [34] Diogo Tavares, Pedro Azevedo, David Semedo, Ricardo Sousa, and Joao Magalhaes. 2023. Task Conditioned BERT for Joint Intent Detection and Slot-filling. In *Progress in Artificial Intelligence - 22nd EPIA Conference on Artificial Intelligence, EPIA 2023, Faial, Portugal, September 5 - 8, 2023, Proceedings*. Springer.
- [35] Diogo Tavares, David Semedo, Alexander Rudnicky, and João Magalhães. 2023. Learning to Ask Questions for Zero-shot Dialogue State Tracking. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*. ACM, 2118–2122. <https://doi.org/10.1145/3539618.3592010>
- [36] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. *CoRR* abs/2302.13971 (2023). <https://doi.org/10.48550/arXiv.2302.13971> arXiv:2302.13971
- [37] Rodrigo Valerio, Joao Bordalo, Michal Yarom, Yonatan Bitton, Idan Szpektor, and João Magalhães. 2023. Transferring Visual Attributes from Natural Language to Verified Image Generation. *CoRR* abs/2305.15026 (2023). <https://doi.org/10.48550/arXiv.2305.15026> arXiv:2305.15026
- [38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NeurIPS*. <http://papers.nips.cc/paper/7181-attention-is-all-you-need>
- [39] Frederico Vicente, Rafael Ferreira, David Semedo, and Joao Magalhaes. 2023. The Wizard of Curiosities: Enriching Dialogues with Fun-facts. In *Proceedings of the 24rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Association for Computational Linguistics, Prague, CZ. https://sigdialinlg2023.github.io/paper_sigdial75.html

- [40] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. 2022. GIT: A Generative Image-to-text Transformer for Vision and Language. *Trans. Mach. Learn. Res.* 2022 (2022). <https://openreview.net/forum?id=b4tMhpNOJC>
- [41] Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. OFA: Unifying Architectures, Tasks, and Modalities Through a Simple Sequence-to-Sequence Learning Framework. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, 23318–23340. <https://proceedings.mlr.press/v162/wang22a1.html>
- [42] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuhui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: Open Pre-trained Transformer Language Models. *CoRR* abs/2205.01068 (2022). <https://doi.org/10.48550/arXiv.2205.01068> arXiv:2205.01068
- [43] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. <https://openreview.net/forum?id=SkeHuCVFDr>

BIBLIOGRAPHY

A Creative Cooking: *What's in your fridge?* - Interaction

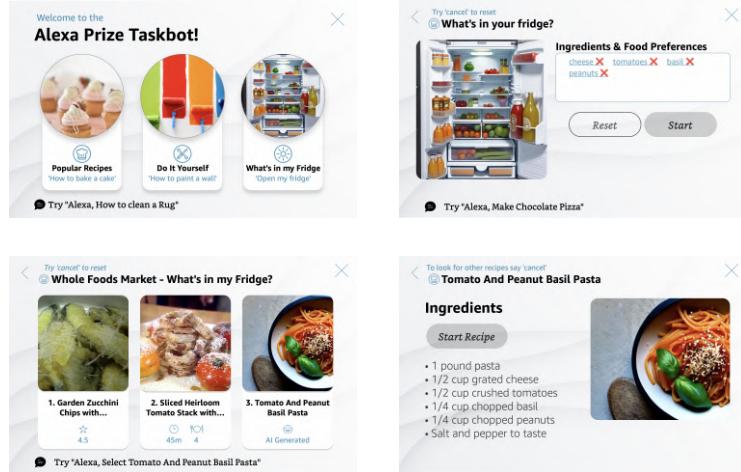


Figure 7: Creative cooking interaction examples.

B Task Promoter

B.1 Example Outputs

Recipe Title	Ingredients	Promoter Output
Lemony Lentil Salad	[cauliflower, green lentils, couscous, lemon, garlic, honey, extra virgin olive oil, parsley, feta cheese]	Lemony Lentil Salad: A zesty, plant-based meal that's packed with flavor and nutrients. Cauliflower, lentils, and couscous come together in a tangy dressing for a satisfying and healthy salad. Top with feta for a pop of flavor and crunch.
Garlic and Cheese Popovers	[butter, chives, lemon, egg, garlic, milk, flour, salt, ground pepper, parsley]	Upgrade your dinner game with these garlic and cheese popovers. Perfect for a cozy night in or a special occasion, they're sure to impress. Trust us, your taste buds will thank you!
Apple Crumble	[apple, sugar, lemon, ground cinnamon, nutmeg, ginger root, salt, flour, oats, dark brown sugar, ground cinnamon, butter]	This is my favorite homemade apple crumble recipe because it's simple to make but comes out perfectly fluffy and light!

Table 7: Promoter example outputs.

B.2 Training Details

To train the *RePro* model, we used 5k recipe descriptions along with their ingredients and steps. For the model, we used a GPT-2-774M [26], trained on instruction data [7]. We then further fine-tuned this model with Direct Preference Optimization (DPO) [27]. Given that DPO requires preference

data, we prompted an LLM [4] for recipe descriptions and discarded any that were hallucinated (e.g. wrong number of ingredients mentioned, or wrong ingredients mentioned).

C TWIZ-LLM

C.1 Dataset

Task Title: Creamy Pesto Chicken Soup with Avocado		
Turn #	User	System
1	-	Step 1: Pour the Chicken Stock in a pan and add Garlic.
2	Got it done.	Step 2: Bring the stock to a boil over high heat and then add the Chicken Breast.
3	I do not have chicken breasts.	No problem, you can also use Turkey.
...
7	Tell me a fun fact.	In the US, on the 31st of July, it's national avocado day.
8	Next.	Step 8: Scoop the avocado chicken soup into bowls.
9	Next step.	We have reached the end of the task.
10	Stop.	Glad I could help you! See you again soon!

Table 8: Example conversation from TWIZ LLM dataset with manually-crafted utterances.

Questions. For the questions, we prompted OpenAI’s API⁷ to generate both question and answer, given the step text.

Ingredients Replacement - To simulate user utterances requesting an ingredient replacement, we apply a set of templates and fill in with an ingredient of a step. The system answer is also template-based, by accessing an external database of ingredient substitutes⁸.

Definition Question - We extract noun phrases from the current step using spaCy and randomly pick one to fill in a “What is?” type question. The answer is extracted from a dictionary⁹, using the intersection between the step’s text and the various definitions to select the best one.

C.2 Example Prompts

Tables 9 and 10 show the prompts used to train the OPT and Vicuna models, respectively.

D Automated Testing

D.1 User Simulator

As TWIZ becomes increasingly more complex and undergoes constant changes, ensuring its correct behavior during testing becomes more challenging. To address this bottleneck in development, we created a testing tool that utilizes CoBot’s interactive mode [17] to simulate user interactions. This user simulation can be conducted using either previously recorded user interactions or generated ones. By making the testing process automatic, the testing time can be shortened, which accelerates the R&D cycle.

While previously recorded user interactions help check if previous features weren’t compromised while adding new ones, the generated user interactions allow to explore previously unseen conversation flows, allowing for the creation of a more robust assistant. In order to generate these interactions, our user simulation tool leverages the dialogue manager state-machine pattern implementation (Section 2.2) and user conversation transcripts. The transcripts provide the necessary data to compute

⁷specifically, *text-davinci-003*

⁸<https://foodsubs.com/>

⁹<https://github.com/wordset/wordset-dictionary>

BIBLIOGRAPHY

Below is a recipe and the steps to make it, the step of the recipe that the user is doing, and part of the dialogue between you and a user about this recipe. Write a response that appropriately responds to the user request. Respond with a *neutral* tone.

Recipe:

Garlicky Broccoli. Step 1: Wash Broccoli in water and shake off excess water. Chop top (head) of the Broccoli into bite-sized pieces. Step 2: Heat Extra-Virgin Olive Oil in a large nonstick skillet over medium-high heat. Step 3: Add Garlic to the pan by crushing with a garlic press. (*other steps omitted*)

Current Step:

Step 1: Wash Broccoli in water and shake off excess water. Chop top (head) of the Broccoli into bite-sized pieces.

Dialog:

User: Is it necessary to wash the broccoli before chopping? System: Yes, it is important to wash the broccoli before chopping to remove any dirt or debris. User: next.

Response:

Heat Extra-Virgin Olive Oil in a large nonstick skillet over medium-high heat.

Table 9: Example of the input prompt to the OPT model.

```
<|prompter|> I will give you a recipe and I want you to help me do it step by step. Please use a neutral tone of voice. Recipe: Garlicky Broccoli. Step 1: Wash Broccoli in water and shake off excess water. Chop top (head) of the Broccoli into bite-sized pieces. Step 2: Heat Extra-Virgin Olive Oil in a large nonstick skillet over medium-high heat. Step 3: Add Garlic to the pan by crushing with a garlic press. (other steps omitted) This is the current step: Wash Broccoli in water and shake off excess water. Chop top (head) of the Broccoli into bite-sized pieces. <|endoftext|> <|assistant|> ok! <|endoftext|> <|prompter|> Is it necessary to wash the broccoli before chopping? <|endoftext|> <|assistant|> Yes, it is important to wash the broccoli before chopping to remove any dirt or debris. <|endoftext|> <|prompter|> next. <|endoftext|> <|assistant|> Heat Extra-Virgin Olive Oil in a large nonstick skillet over medium-high heat. <|endoftext|> </s>
```

Table 10: Example of the input prompt to the Vicuna model.

state transition probabilities and enable the creation of a user utterance bank linked to specific state and event pairs.

The user simulation generation process starts with the tool initiating an interaction with TWIZ. For each turn of the conversation, an event is selected based on the computed transition probabilities and the current state of the dialogue. Then, the user utterances related to the current state and sampled event are ranked by similarity using sentence embeddings [28]. This ranking is performed by calculating the contextual embeddings of the ongoing conversation and determining the cosine similarity with the contextual embeddings of all potential utterances. This makes it so that the most relevant and contextually appropriate utterance is chosen, resulting in compelling and cohesive dialogues. An example is provided in Figure 8.

D.2 Rating Prediction for Conversational Task Assistants

Due to the complex interactions between the user and the system, errors are prone to happen which in turn lead to user dissatisfaction and low ratings. Being able to predict the rating of an interaction is thus a critical step to understand the system’s shortcomings, and act accordingly [18, 5]. Emphasizing this, on average less than 5% of conversations have an associated rating, making it difficult to decide which conversations should be prioritized for analysis. Moreover, rating prediction complements the creation of the simulated interactions (Section D.1), allowing to automatically rate these generated interactions.

Inspired by the work from Choi *et al.* [5] in SocialBot, we developed a rating prediction model specific to the TaskBot setting [9], where the aim is to predict a rating given an entire conversation.

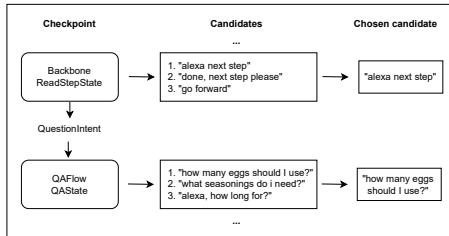


Figure 8: User simulation procedure.

Our model is based on the Transformer [38] architecture and combines both textual features, i.e., the conversation between the user and system, and user-behavior features, such as the number of fallbacks/steps read, and other general and Taskbot-specific features. To evaluate our model, we used conversations and ratings collected during the first edition of the Alexa Prize TaskBot challenge. In this setting, the model achieves an accuracy of 70% in a binary rating classification task. The results demonstrated the utility of automatic rating prediction, revealing insights such as the significance of starting a task and the increased importance of the latter part of the conversation [9].

BIBLIOGRAPHY

A	B	C	D	E
Id	Input	Prompt	Rating: -1: Errors in the generation or in the input 1: Prompt doesn't describe the current step 2: Prompt describes the current step, but some context is missing in the input/output 3: Prompt is a poor description of the current step, and is in accordance with the context 4: Prompt is a good description of the current step, and is in accordance with the context 5: Prompt is a very good description of the current step, and is in accordance with the context	Note (optional) A: copy problem B: step has too many actions C: good prompt DESPITE step with too many actions D: prompt has hallucinations
51	Step 1: In a small saucepan, combine Frozen Spinach and Artichoke Hearts Packed in Water over medium heat, frying for 1 minute.	a small saucepan with spinach and artichoke hearts in it	4	
49	Step 1: Into a blender, add White Onion and Green Bell Pepper. Pulse to finely chop. Step 2: Into a pressure cooker on high, add Olive Oil and chopped veggies. Cook for 10-15 minutes.	a pressure cooker with chopped vegetables in it	4	
85	Step 1: Slice the beef into thin strips. Add half of the soy sauce and juice from half a lime to a bowl. Add beef, tossing to coat, and set aside to marinate. Step 2: In a small bowl, whisk together the beef stock, cane sugar, remaining soy sauce, cornstarch, and remaining lime juice. Set aside. Peel the garlic cloves and ginger and finely chop. Thinly slice the green onions and chili. Transfer to a bowl and set aside. Cut or pull apart the broccoli into small florets.	A person is chopping up garlic, ginger, green onions, and broccoli on a cutting board	4	C

Figure .1: Annotation of the Sequence Context Decoder.

4 Human Annotations

In this Section, we present examples of the annotation tasks.

The human annotation pool consisted of 3 PhD students and 5 MSc students. 25% of the annotators were women and 85% were men.

Figure .1 shows the annotation task for the quality of the Sequence Context Decoder, where annotators were asked to rate the outputs of the model on a 5-point Likert scale. Additionally, they could rate it as -1, in case of error, or leave a **Note** with an observed error. Figure .2 shows the annotation task to choose the best visual coherence maintaining method. The annotators saw 5 sequences: *Random Seed*, *Fixed Seed*, *Latent 1*, *Latent 2*, and *Image-to-Image*. They were then asked to pick the best, second best, and third best sequences. They could also indicate that there were no good sequences, by checking the *No good sequence* checkbox. Additionally, they could leave an observation, in the appropriate text area. Figure .3 shows the annotation task to tune the threshold of our method. The annotators had to pick between 5 sequences, generated with different values of threshold: 0.50, 0.55, 0.60, 0.65, 0.70. They were asked to pick the best, second best, and third best sequences. They could also indicate that there were no good sequences, by checking the *No good sequence* checkbox. They could also leave an observation, in the appropriate text area. Figure .4 shows the annotation task to choose between our method and the winning visual coherence maintaining method. The annotators saw 2 sequences, one generated with *Latent 1* and another with our method. They had to choose the win sequence, or deem them equivalent. If there was no good sequence, they could check the *No good sequence* checkbox. They could also leave an observation. Figure .5 shows the annotation task to rate sequences generated with our method and the ground-truth images, from a scale of 1 to 5. Additionally, the annotators could select that there was no good sequence, or leave an observation. Figure .6 shows the annotation guidelines for the task to rate sequences generated with our method and the ground-truth images.

Sequence Evaluation

Jamaican Callaloo With Shrimp

Steps	A	B	C	D	E
Heat the Coconut Oil in a wide pan over a medium flame, then add the Onion, Garlic, Scallion, and Ground Black Pepper. Reduce the heat to low for about 3-4 minutes.					
Add the Small Shrimp, stir well and cook for another 3 minutes.					
Turn the heat up to medium high and add the Jamaican Callaloo, Tomato, Scotch Bonnet Pepper, Fresh Thyme, and Sea Salt. After a couple minutes, add the Water and cook until tender.					
After about 10-12 minutes, taste for salt and adjust accordingly.					
Best Sequence: <input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D <input type="radio"/> E Second Best Sequence: <input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D <input type="radio"/> E Third Best Sequence: <input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D <input type="radio"/> E <input type="checkbox"/> No good sequence					
Observations <input type="button" value="Submit"/>					

Figure .2: Annotation of the comparison between visual coherence methods.

.5 Additional Examples

In this Section, we show additional examples produced by our method. Figures .7, .8, .9, .10, .11, and .12 show examples for the recipes domain. Figures .13, .14, and .15 show examples for the how-to guides domain. Figure .16 shows a specific manual task that is particularly challenging to illustrate.

.6 Datasets

As mentioned in Sections 2.5, the lack of large scale datasets is a big challenge in multimodal learning and specially in the task of VQA. Recent work has been done in the direction of producing training data for this downstream task.

BIBLIOGRAPHY

Sequence Evaluation

Jamaican Callaloo With Shrimp

Steps	A	B	C	D	E
Heat the Coconut Oil in a wide pan over a medium flame, then add the Onion, Garlic, Scallion, and Ground Black Pepper. Reduce the heat to low for about 3-4 minutes.					
Add the Small Shrimp, stir well and cook for another 3 minutes.					
Turn the heat up to medium high and add the Jamaican Callaloo, Tomato, Scotch Bonnet Pepper, Fresh Thyme, and Sea Salt. After about 10 minutes, add the Water and cook until tender.					
After about 10-12 minutes, taste for salt and adjust accordingly.					

Best Sequence:
 A B C D E

Second Best Sequence:
 A B C D E

Third Best Sequence:
 A B C D E

No good sequence

Observations

Figure .3: Annotation of the comparison between different heuristic thresholds.

.6.1 Dataset Crawling

.6.1.1 WIT

The authors of [64] mention that the main approach of existing work has been to use three datasets, MS-COCO [44], Visual Genome [37] and YFCC100M [80]. Although the first two are high-quality crowd-labeled datasets, they are small for large scale models, with approximately 100,000 training images each. The third dataset has a larger scale, at 100 million photos but the metadata associated with each image is "sparse and of varying quality" [64], and being brought down to around 15 million photos when filtered to keep only data suitable for this type of pretraining. Since existing datasets are of small scale, the authors construct a new dataset consisting of 400 million (image, text) pairs collected

from various "publicly available sources on the Internet" [64]. In an attempt to cover the broadest set of visual concepts possible, the authors add a restriction to the text of the (image, text) pair. The text must contain one of the 500,000 queries present in a query list constructed from all the words which appeared at least 100 times on the English version of Wikipedia. The results are balanced by including up to 20,000 pairs per query. This dataset is referred to as WIT, which stands for WebImageText. This approach has the possibility of achieving large scale and the paper proves that a large model can learn from this type of data.

.6.1.2 PMD

In [71], the authors construct a "corpus out of publicly available sources of image-text data" [71]. Among these sources are the datasets mentioned previously in this section. For YFCC100M, the data is filtered to exclude non-English captions and captions which contain less than two words. That's the only filtering applied to the data. This corpus consists of 70 million text-image pairs retrieved from open datasets which are freely accessible by other researchers, so as to allow reproducibility and facilitate future work.

.6.1.3 HowTo100M

The authors of [57] point out the lack of large scale datasets consisting of video clips paired with captions. Such datasets are too expensive and time consuming to scale to a large number of pairs, since most must be manually annotated. In addition, video annotation "can often be an ill-defined task with low annotator consistency" [57].

They propose learning from video data "video data with readily available natural language annotations in the form of automatically transcribed narrations" [57]. The approach is similar to previous work and consists of searching for tasks on WikiHow and pairing them with videos found on YouTube, for each task. The distinction between this work and previous efforts is the "unprecedented scale both in terms of variety (...) and size" [57]. These type of instructional videos are rising in popularity and often contain "narration with an explicit intention of explaining the visual content on screen" [57], making them a promising target for such an approach. The dataset contains "136 million video clips sourced from 1.22 million narrated instructional videos" [57] spanning over 23,000 tasks from varying domains such as cooking, hand crafting and personal care. The primary focus is on tasks which involve interacting with the physical world. "Each clip is paired with a text annotation in the form of an automatically transcribed narration" [57].

The data was filtered in two steps. The initial WikiHow search filtered out domains which had more abstract tasks as opposed to the desired predominantly visual tasks. The initial set was refined by restricting the primary verb of the task to physical actions, discarding non-physical verbs, such as *be* or *feel*. For videos, videos with English subtitles, uploaded manually or generated automatically, were considered. To improve quality and consistency, the authors removed videos with less than 100 views, less than 100 words

and stick to the top 200 results. There was also an upper bound of 2000 seconds for the length of each video. For creating video-text pairs, the authors "select each line of the subtitles as a caption, and pair it with the video clip from the time interval corresponding to the line" [57].

Some problems with the approach include the fact that, being automatically generated, the captions often have grammatical mistakes, lack punctuation or are incomplete. In addition, the content producer may speak about something which is unrelated to the video or describe something before or after it happens.

After training their model on this dataset, authors found that for instructional video datasets, their model significantly improves over the SOTA models trained on the smaller manually-annotated datasets. On generic videos, the model was still able to be competitive with SOTA models trained on MSR-VTT, a dataset for the open domain video captioning, but was able to outperform them after being fine-tuned on a small subset of this dataset. In this same line of work, fine-tuning on LSMDC, a dataset of movie clips paired with captions extracted from their script, allowed the model to generalize to this domain, despite the "large domain gap" [57].

.6.2 Data Augmentation

A very different approach is taken in [9]. In this paper, the authors propose a method, which they call Visual Question Generation with Question Answering validation or VQ²A, to automatically derive VQA examples at scale, by "leveraging the abundance of existing image-caption annotations" in combination with previously established models for textual question generation. The method consists of three main stages: Candidate Answer Extraction, Question Generation and Question-Answering Filtering. Candidate Answer Extraction is the act of extracting candidate answers from a given caption. To this end, the caption is parsed and the candidates are extracted based on the Part-of-Speech (POS) and dependency parse tree annotations. Since boolean questions are frequent in VQA benchmarks but infrequent in captions, the authors add "yes" and "no" as candidate answers and generate one question per candidate. Also infrequent in captions are mentions of "zero". To correct for this, the authors sample a generated "How many?" question at random from another caption and add it with the answer as "zero". Although this method has the potential to be noisy, it was rare for the answer to the sampled question to be non-zero in the target image. The second stage, Question Generation, consists of a model which takes as input a caption and a candidate answer and generates an adequate question. The answer does not need to appear as is in the caption. The model used is T5-XXL, further fine-tuned on SQuAD1.1 for question generation. The authors use the top-scoring generated question for each caption-answer pair. The final stage is Question-Answer Filtering and it aims to tackle the hallucination problem, in which models generate content which is not consistent with its input source. To mitigate the problem, the generated question is answered with a question answering model. If the answer does not match the answer

candidate offered as input to the question generation model, the generated question is discarded. This match is determined from a token-level F1 score and the discarding happens if the score is below a predetermined threshold. The question answering model used is the T5-XXL model fine-tuned on SQuAD2.0 and Natural Questions.

To assess the performance of the proposed model, the authors generate VQA triplets from two sources of image captions: MSCOCO Captions (COCO-CAP) and Conceptual Captions (CC3M). The former is a smaller human-curated dataset with "cleaner" captions which represent the image content more accurately, with around 120,000 images. CC3M has around 3 million images automatically collected from the web, each associated with an alt-text. The authors point out that using COCO-CAP would "show the potential of training a VQA model using VQ²A in a "cleaner" zero-shot setup" [9] and using CC3M would show the "potential of training on noisy web image alt-text pairs, where scaling up to billions of examples is possible" [9]. The performance is evaluated as a VQA as classification task on three datasets and the authors find that the new models achieve new SOTA results in the zero-shot transfer learning setting. The results "significantly close the performance gap between automatically-generated and manually constructed training sources" [9], suggesting that this method may reduce the need for human-curated VQA datasets. By comparing the two datasets used for evaluation, one sees that the gap between performance is not large, indicating that it doesn't seem to be crucial to have starting captions which are manually annotated, like COCO-CAP, and that the captions provided by CC3M are competitive in zero-shot VQA performance.

This work show that this method results in high-quality data and that VQA models trained on this data pushed the SOTA forward at zero-shot accuracy and improved robustness while, at the same time, reducing the need for expensive human-annotated datasets.

BIBLIOGRAPHY

Sequence Evaluation

Jamaican Callaloo With Shrimp

Steps

A



B



Heat the Coconut Oil in a wide pan over a medium flame, then add the Onion, Garlic, Scallion, and Ground Black Pepper. Reduce the heat to low for about 3-4 minutes.

Add the Small Shrimp, stir well and cook for another 3 minutes.



Turn the heat up to medium high and add the Jamaican Callaloo, Tomato, Scotch Bonnet Pepper, Fresh Thyme, and Sea Salt. After a couple minutes, add the Water and cook until tender.



After about 10-12 minutes, taste for salt and adjust accordingly.



Best Sequence:

A B Equivalent

No good sequence

Observations

Figure 4: Annotation of the comparison between our method and the best visual coherence method.

Sequence Evaluation

Jamaican Callaloo With Shrimp

Steps

<p>A</p>  <p>Heat the Coconut Oil in a wide pan over a medium flame, then add the Onion, Garlic, Scallion, and Ground Black Pepper. Reduce the heat to low for about 3-4 minutes.</p>	<p>B</p>  <p>Add the Small Shrimp, stir well and cook for another 3 minutes.</p>
 <p>Turn the heat up to medium high and add the Jamaican Callaloo, Tomato, Scotch Bonnet Pepper, Fresh Thyme, and Sea Salt. After a couple minutes, add the Water and cook until tender.</p>	
 <p>After about 10-12 minutes, taste for salt and adjust accordingly.</p>	

Rating for A:

1 2 3 4 5

Rating for B:

1 2 3 4 5

No good sequence

Observations

Submit

Figure .5: Annotation of the comparison between our method and the ground-truth images.

BIBLIOGRAPHY

Instructions

We will present you two side-by-side **sequences of illustrations** for a recipe.

On the left of the sequences, you will see the **step instruction**.

We ask you to rate each sequence on a Likert scale of 1-5.

Your rating should based on two factors:

- How well does each illustration **represent the step** instruction?
 - Note: Generation problems should not affect the rating if the image still clearly illustrates the step.
- How **coherent** is the sequence of illustrations, as a whole?
 - Example: A **blue** pan remains **blue** in the following images.
 - Example: The **background scene** is **preserved** between images.

If none of the sequences correctly illustrates the recipe, you can check the option: *No Good Sequence*.

You can also leave an observation, before submitting your answer.

Figure .6: Annotation guidelines for the comparison between our method and the ground-truth images.

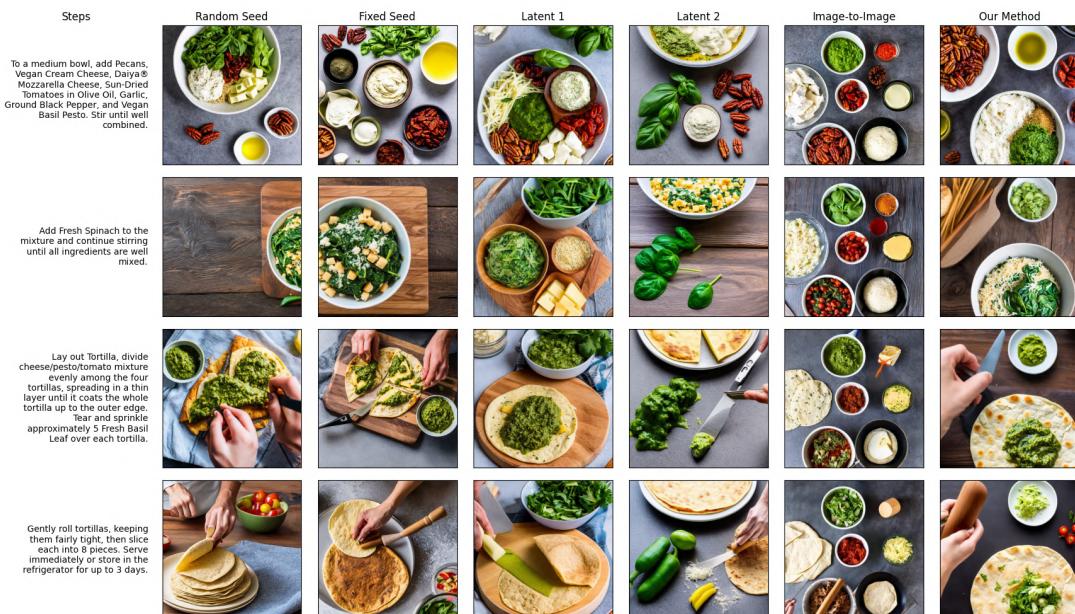


Figure .7: Examples of recipe illustrations with different methods for maintaining visual coherence.

.6. DATASETS

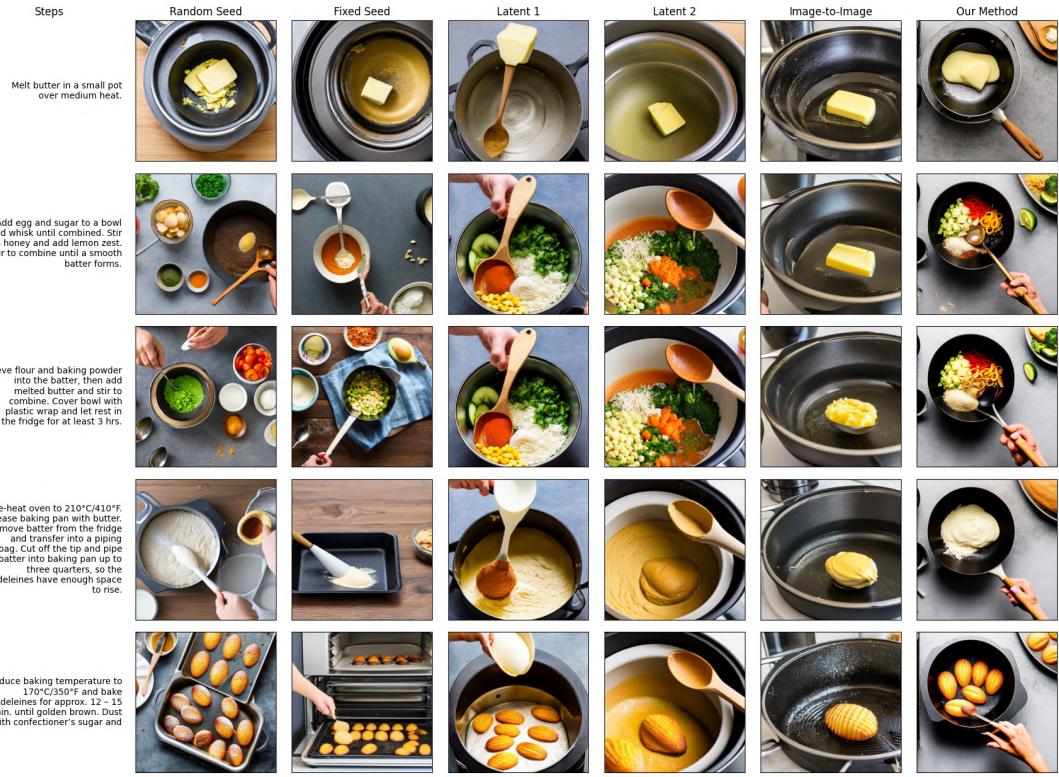


Figure .8: Examples of recipe illustrations with different methods for maintaining visual coherence.



Figure .9: Examples of recipe illustrations with different methods for maintaining visual coherence.

BIBLIOGRAPHY

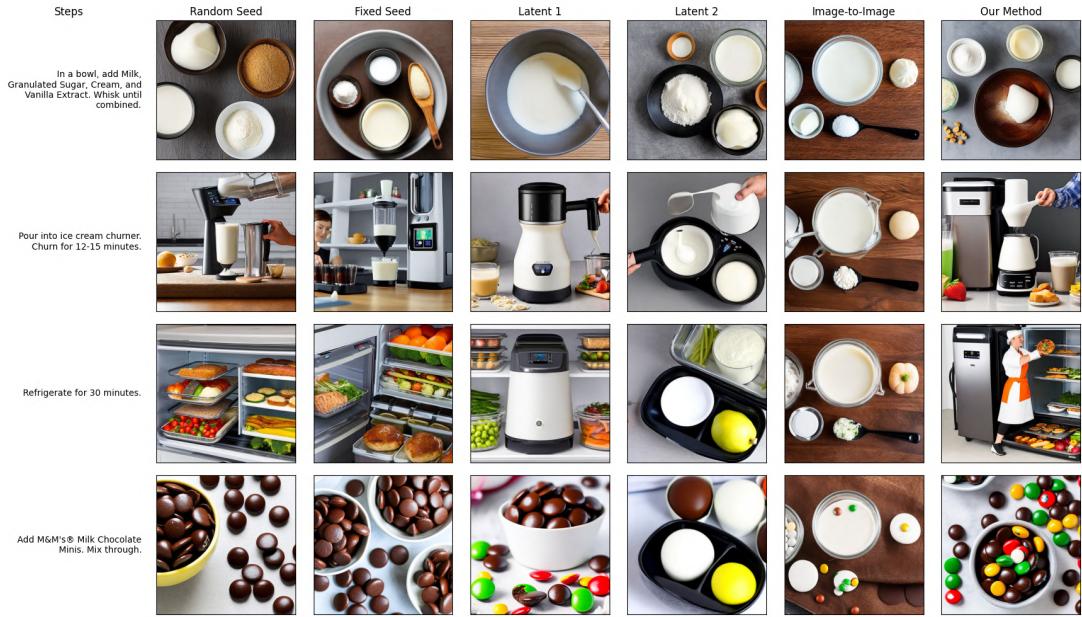


Figure .10: Examples of recipe illustrations with different methods for maintaining visual coherence.

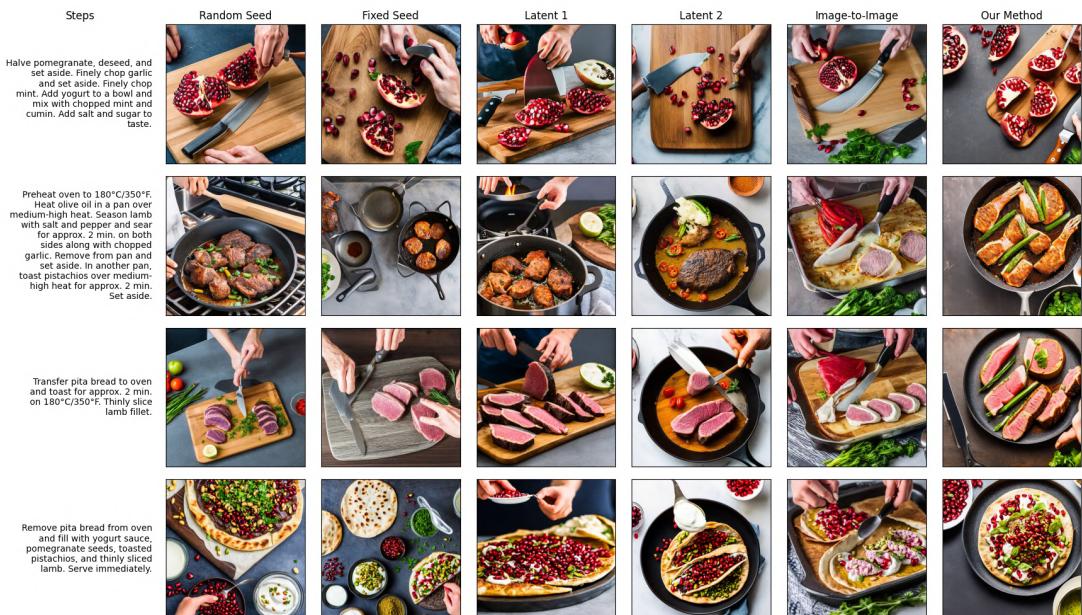


Figure .11: Examples of recipe illustrations with different methods for maintaining visual coherence.



Figure .12: Examples of recipe illustrations with different methods for maintaining visual coherence.

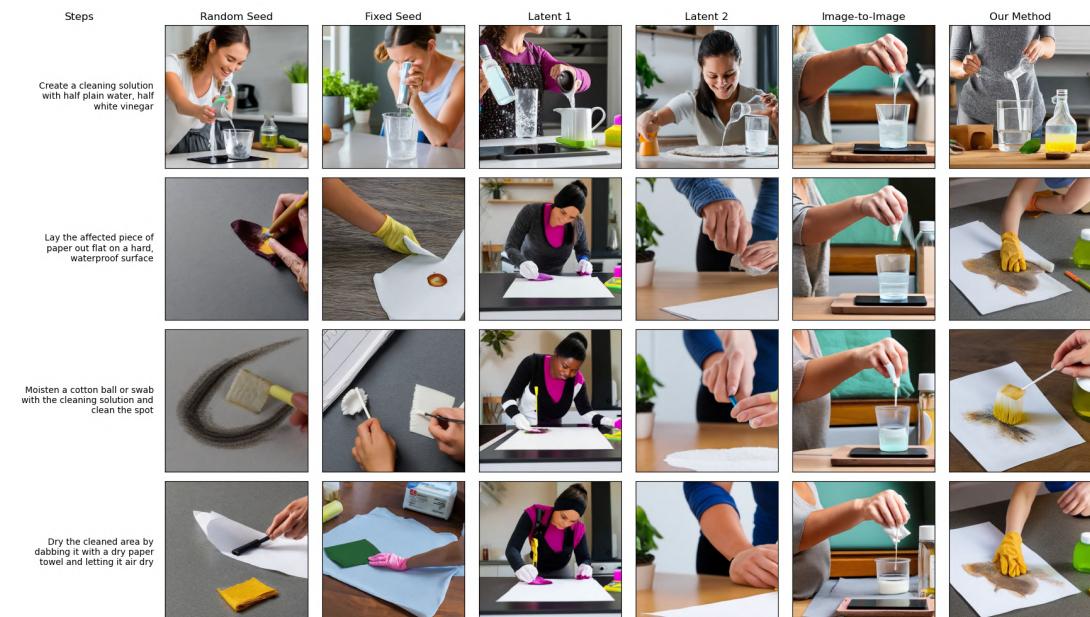


Figure .13: Examples of task illustrations with different methods for maintaining visual coherence.

BIBLIOGRAPHY



Figure .14: Examples of task illustrations with different methods for maintaining visual coherence.



Figure .15: Examples of task illustrations with different methods for maintaining visual coherence.



Figure .16: Example of a task that is very challenging to illustrate. We can see how the generated images still capture some of the more challenging elements of the steps, such as "Make a note of the longitude and latitude" in step 4, with the images showing a pen.



