论文汇报

1. ZeroShotDataAug: Generating and Augmenting Training Data with ChatGPT

2. Visual ChatGPT: Talking, Drawing and Editing with Visual Foundation Models

汇报人：陈鑫宇

时间：2023年4月19日

ZeroShotDataAug: Generating and Augmenting Training Data with ChatGPT

The main contributions of this paper are:

- Evaluation of zero-shot prompting of ChatGPT for data augmentation on three datasets.
- Three methodologies to evaluate the similarity of the data generated from zero-shot prompting of ChatGPT with the training and test sets with the aim of validating and assessing the quality of the data generated.
- Investigation of the marginal returns of data generated from different data augmentation techniques.

三个数据集：

SST-2: 影评数据集（pos,neg）

SNIPS:与音乐、天气和家庭自动化领域的七个意图相关的注释语音query。

TREC: 问题分类数据集

SST-2: 影评数据集（pos,neg）

Class: Positive
Prompt: Generate 20 sentences that are positive reviews to a movie
Class: Negative
Prompt: Generate 20 sentences that are negative reviews to a movie

SNIPS:与音乐、天气和家庭自动化领域的七个意图相关的注释语音query。

Dataset: SNIPS

Class: RateBook
Prompt: Generate 20 sentences in an imperative mood where a human tells a digital assistant to rate a random book and the human provides the numerical rating. Use random book names. Do not mention the name of the digital assistant.
Class: AddToPlaylist
Prompt: Generate 20 sentences in an imperative mood where a human tells a digital assistant to add music to a playlist and the human provides the music name. Use random music and playlist names. Do not mention the name of the digital assistant.
Class: PlayMusic
Prompt: Generate 20 sentences in an imperative mood where a human tells a digital assistant to play a music and the human provides the music name. Use random music and names. Do not mention the name of the digital assistant.
Class: BookRestaurant
Prompt: Generate 20 sentences in an imperative mood where a human tells a digital assistant to book a restaurant and the human provides the restaurant or food name. Use random restaurant and food names. Do not mention the name of the digital assistant.

TREC: 问题分类数据集

Class: Abbreviation
Prompt: Generate 20 questions asking about the meaning of an abbreviation
Class: Entity
Prompt: Generate 20 questions asking about a random example of a noun or entity. Actually use different nouns or entities in each sentence.
Class: Description
Prompt: Generate 20 sentences that are only "what is" questions that query for a definition.
Class: Human
Prompt: Generate 20 questions about random facts about a person or people in history.
Class: Location
Prompt: Generate 20 sentences that are questions that ask the location of a place in history. Use a different place for each sentence
Class: Numeric Value
Prompt: Generate 20 sentences that are questions about a numeric fact in history

TREC: 问题分类数据集

Class: Abbreviation
Prompt: Generate 20 questions asking about the meaning of an abbreviation
Class: Entity
Prompt: Generate 20 questions asking about a random example of a noun or entity. Actually use different nouns or entities in each sentence.
Class: Description
Prompt: Generate 20 sentences that are only "what is" questions that query for a definition.
Class: Human
Prompt: Generate 20 questions about random facts about a person or people in history.
Class: Location
Prompt: Generate 20 sentences that are questions that ask the location of a place in history. Use a different place for each sentence
Class: Numeric Value
Prompt: Generate 20 sentences that are questions about a numeric fact in history

相似度方法

- **Sentence Embedding**: We used the MiniLM model [13] in the sentence transformer library [10] to obtain the embedding of each example. We calculated the cosine similarity between the embeddings of the pair of examples.
- **TF-IDF**: We used TF-IDF vectors to calculate the similarity between examples. First, we removed the stop words from each example and created a corpus by combining the training data, testing data and zero-shot ChatGPT generated data. We used the resulting corpus as the corpus for the TF-IDF vector similarity approach. Finally, we converted each example into a TF-IDF vector and calculated the cosine similarity between each example.
- **Word Overlap**: We calculated the word overlap scores between examples to obtain their similarity. First, we removed the stop words and punctuation from all the examples. Next, for a example pair, we counted the number of unique words that appear in both examples and divided it by the number of unique words that appear on the longer example of the pair. This metric gives us the percentage of overlapping words between the two examples.

	Sentence Embedding	TF-IDF	Word Overlap
ChatGPTzero-shot to SNIPStest	0.553	0.239	0.265
SNIPStrain to SNIPStest	0.593	0.362	0.426
ChatGPTzero-shot to TRECtest	0.528	0.330	0.271
TRECtrain to TRECtest	0.448	0.240	0.202
ChatGPTzero-shot to SST-2test	0.600	0.293	0.271
SST-2train to SST-2test	0.535	0.229	0.211

Table 1: Data augmentation similarity results of ChatGPTzero-shot versus the original training sets relative to the testing sets.

	Sentence Embedding	TF-IDF	Word Overlap
ChatGPTzero-shot to SNIPStrain	0.629	0.341	0.360
ChatGPTzero-shot to TRECtrain	0.634	0.435	0.404
ChatGPTzero-shot to SST-2train	0.635	0.351	0.317

Table 2: Data augmentation similarity results between ChatGPTzero-shot and training sets.

Model	SST-2	SNIPS	TREC
No Aug	52.9 (5.0)	79.4 (3.2)	48.6 (11.5)
EDA	53.8 (4.4)	85.8 (3.0)	52.6 (10.5)
BackTrans.	57.5 (5.6)	86.5 (2.4)	66.2 (8.5)
CBERT	57.4 (6.7)	85.8 (3.5)	64.3 (10.9)
BERTexpand	56.3 (6.5)	86.1 (2.7)	65.3 (6.1)
BERTprepend	56.1 (6.3)	86.8 (1.6)	64.7 (9.6)
GPT2context	55.4 (6.7)	86.6 (2.7)	54.3 (10.1)
BARTword	58.0 (6.8)	86.8 (2.6)	63.7 (9.8)
BARTspan	57.7 (7.1)	87.2 (1.4)	67.3 (6.1)
ChatGPTfew-shot	69.6 (5.8)	**91.3** (1.4)	66.7 (8.0)
ChatGPTzero-shot	**78.1** (5.1)	91.2 (1.3)	**75.3** (4.0)

Table 4: Accuracy (and standard deviation) for each data augmentation method. All results except for ChatGPTfew-shot and ChatGPTzero-shot were the same results reported by Kumar et al. [4].

Visual ChatGPT: Talking, Drawing and Editing with Visual Foundation Models

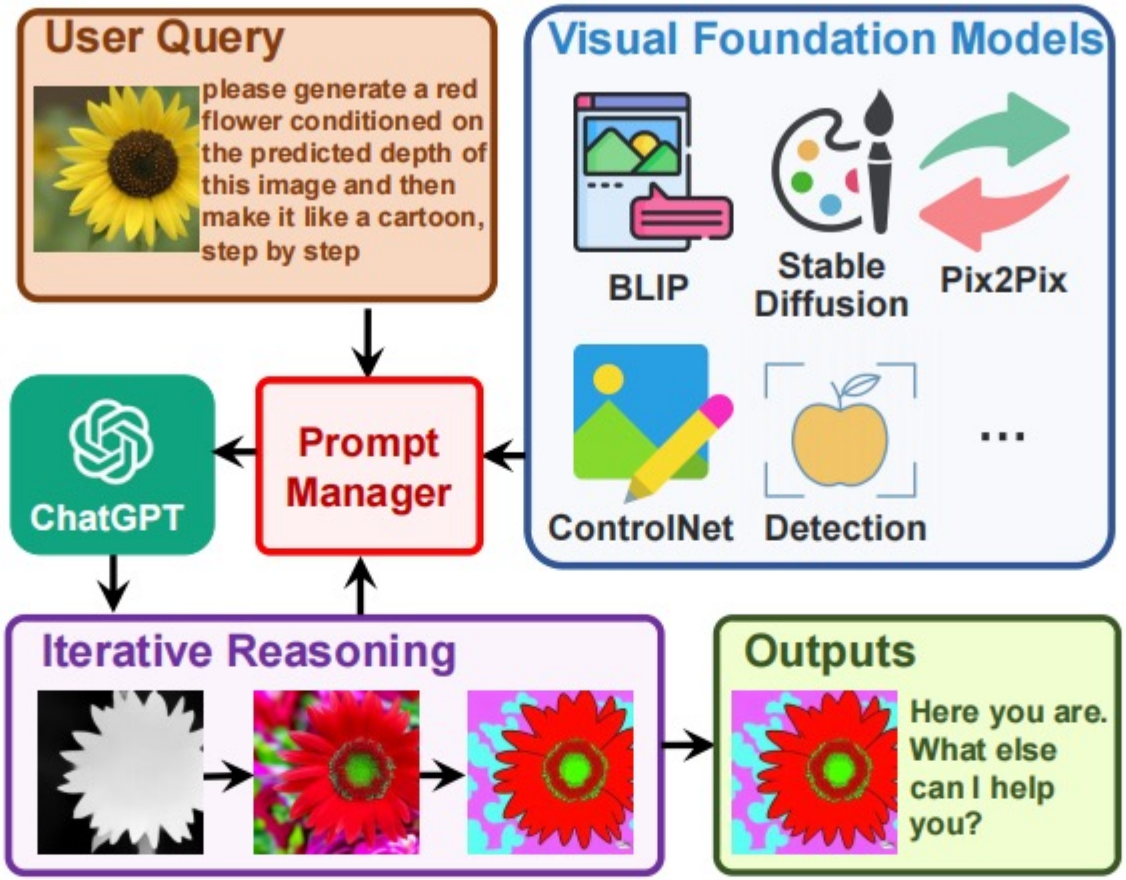


Figure 1. Architecture of Visual ChatGPT.

$S = (Q_1, A_1), (Q_2, A_2), \ldots, (Q_N, A_N)$

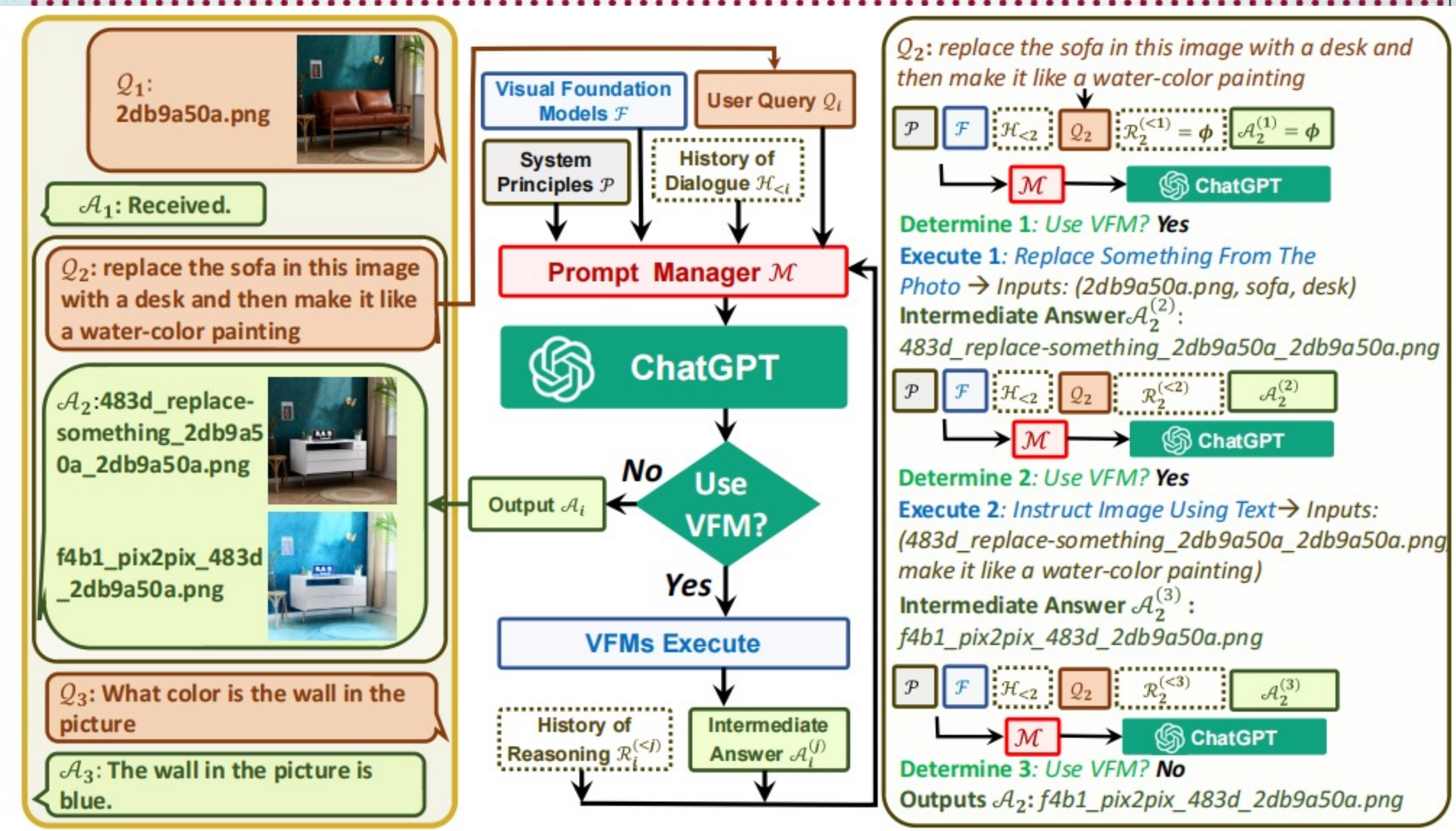
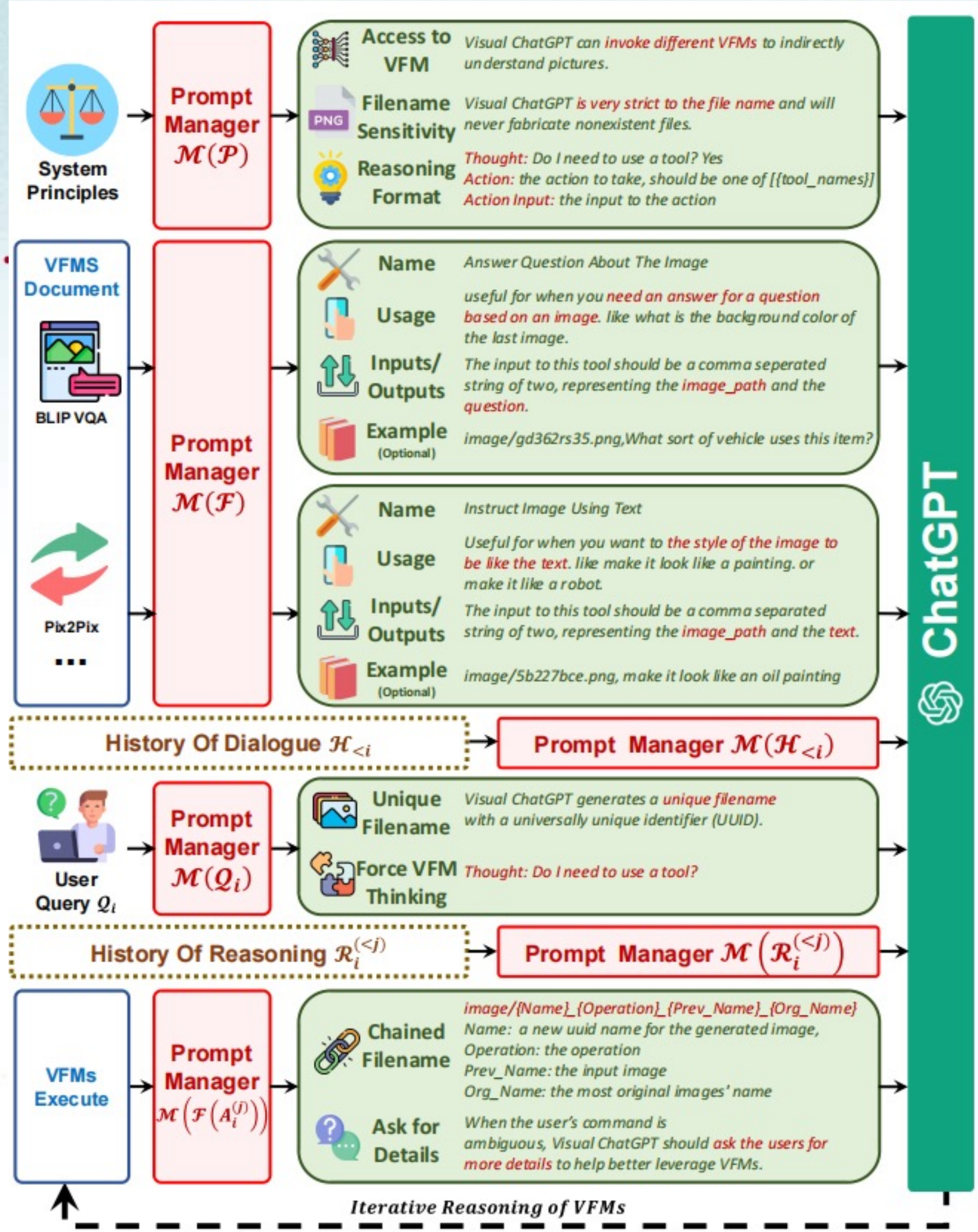
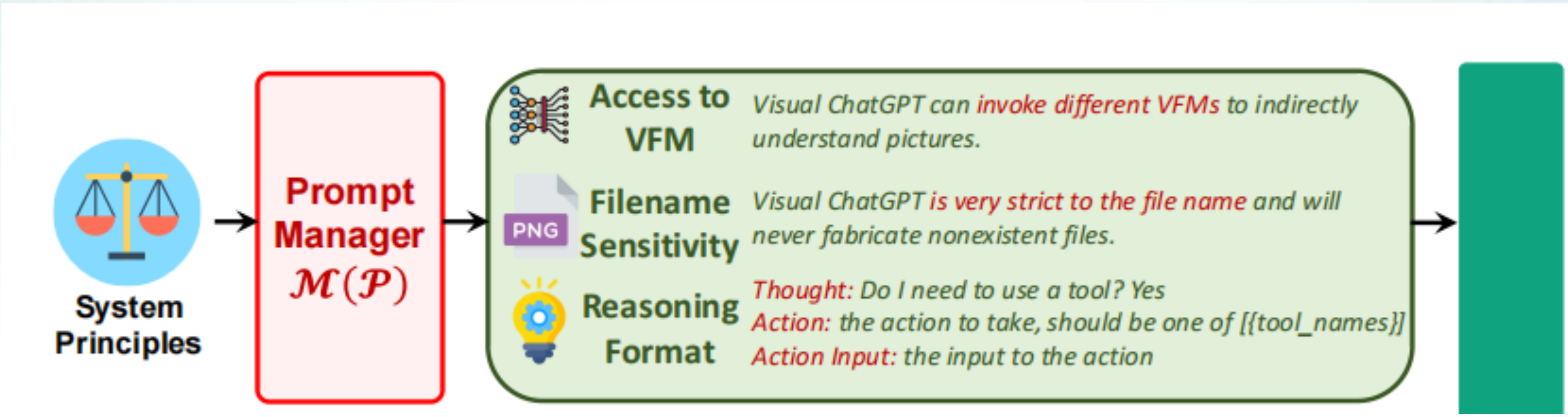
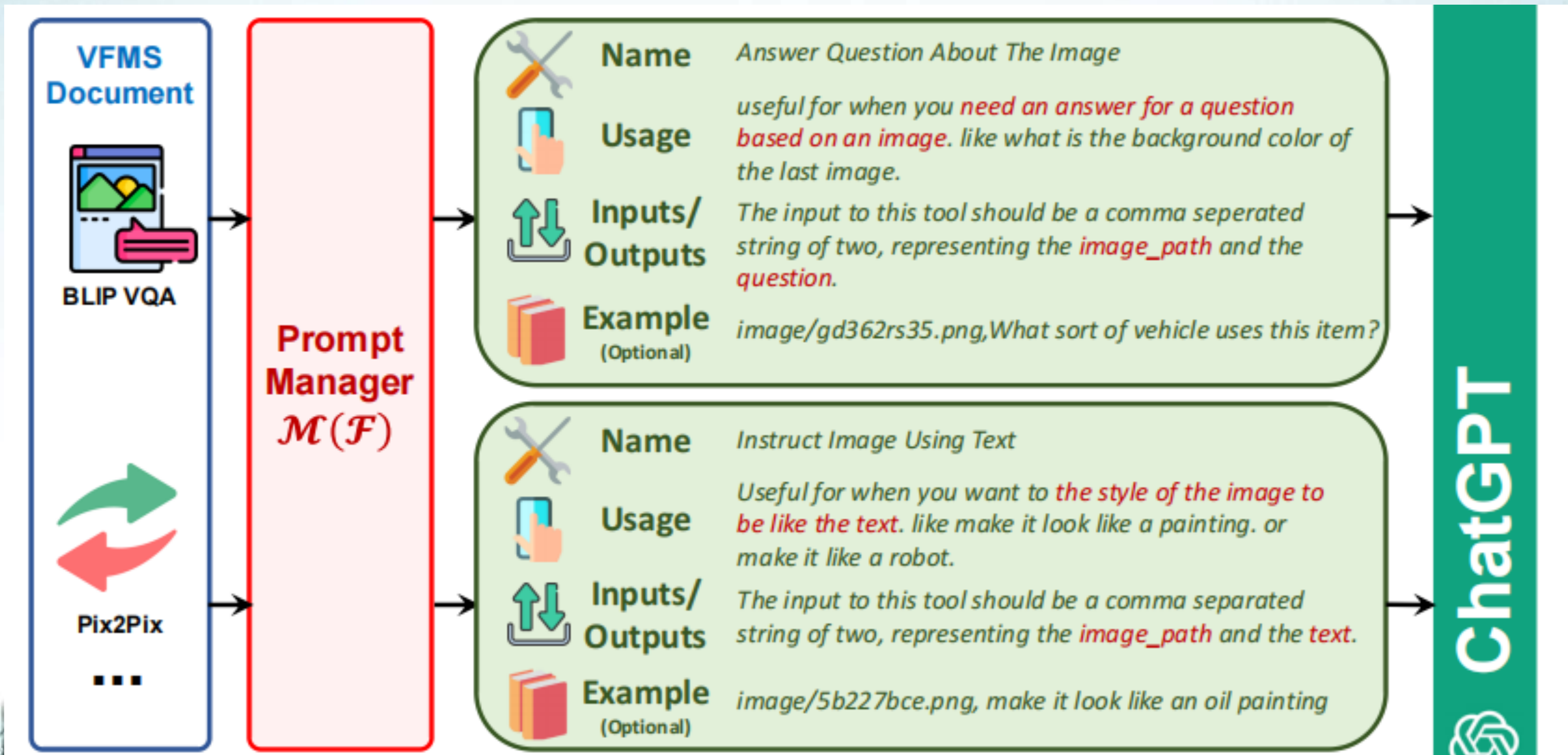


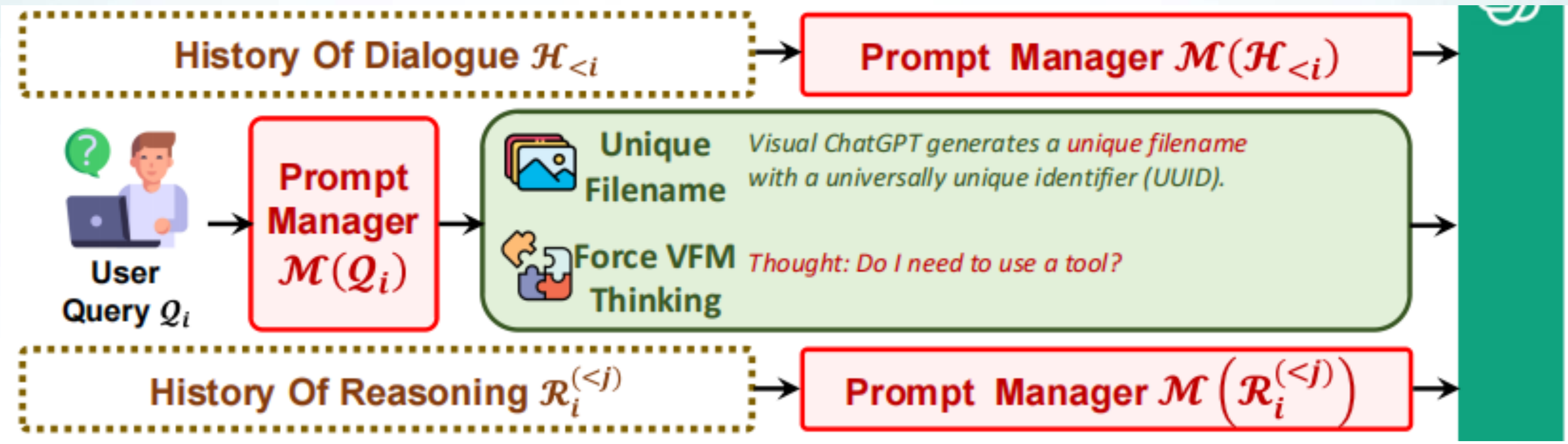
Figure 2. Overview of Visual ChatGPT. The left side shows a three-round dialogue, The middle side shows the flowchart of how Visual ChatGPT iteratively invokes Visual Foundation Models and provide answers. The right side shows the detailed process of the second QA.

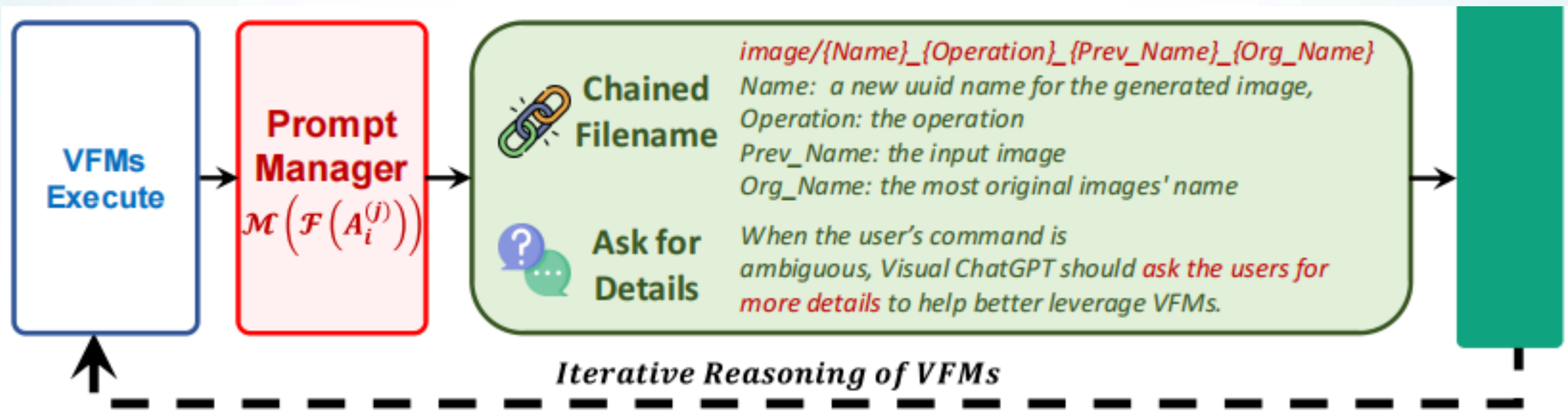

System Principles
Prompt Manager $\mathcal{M}(\mathcal{P})$
Access to VFM
Visual ChatGPT can invoke different VFMs to indirectly understand pictures.
PNG
Filename Sensitivity
Visual ChatGPT is very strict to the file name and will never fabricate nonexistent files.
Reasoning Format
Thought: Do I need to use a tool? Yes
Action: the action to take, should be one of [{tool_names}]
Action Input: the input to the action
VFMS Document
BLIP VQA
Pix2Pix
...
Prompt Manager $\mathcal{M}(\mathcal{F})$
Name
Answer Question About The Image
Usage
useful for when you need an answer for a question based on an image. like what is the background color of the last image.
Inputs/ Outputs
The input to this tool should be a comma seperated string of two, representing the image_path and the question.
Example (Optional)
image/gd362rs35.png,What sort of vehicle uses this item?
Name
Instruct Image Using Text
Usage
Useful for when you want to the style of the image to be like the text. like make it look like a painting. or make it like a robot.
Inputs/ Outputs
The input to this tool should be a comma separated string of two, representing the image_path and the text.
Example (Optional)
image/5b227bce.png, make it look like an oil painting
History Of Dialogue $\mathcal{H}_{<i}$
Prompt Manager $\mathcal{M}(\mathcal{H}_{<i})$
User Query Q_i
Prompt Manager $\mathcal{M}(Q_i)$
Unique Filename
Visual ChatGPT generates a unique filename with a universally unique identifier (UUID).
Force VFM Thinking
Thought: Do I need to use a tool?
History Of Reasoning $\mathcal{R}_i^{(<j)}$
Prompt Manager $\mathcal{M}\left(\mathcal{R}_i^{(<j)}\right)$
VFMs Execute
Prompt Manager $\mathcal{M}\left(\mathcal{F}\left(A_i^{(j)}\right)\right)$
Chained Filename
image/{Name}_{Operation}_{Prev_Name}_{Org_Name}
Name: a new uuid name for the generated image,
Operation: the operation
Prev_Name: the input image
Org_Name: the most original images' name
Ask for Details
When the user's command is ambiguous, Visual ChatGPT should ask the users for more details to help better leverage VFMs.
ChatGPT
Iterative Reasoning of VFMs



System Principles
Prompt Manager $\mathcal{M}(\mathcal{P})$
Access to VFM
Visual ChatGPT can invoke different VFMs to indirectly understand pictures.
PNG
Filename Sensitivity
Visual ChatGPT is very strict to the file name and will never fabricate nonexistent files.
Reasoning Format
Thought: Do I need to use a tool? Yes
Action: the action to take, should be one of [{tool_names}]
Action Input: the input to the action


VFMS Document
BLIP VQA
Pix2Pix
...
Prompt Manager $\mathcal{M}(\mathcal{F})$
Name
Answer Question About The Image
Usage
useful for when you need an answer for a question based on an image. like what is the background color of the last image.
Inputs/ Outputs
The input to this tool should be a comma seperated string of two, representing the image_path and the question.
Example (Optional)
image/gd362rs35.png,What sort of vehicle uses this item?
Name
Instruct Image Using Text
Usage
Useful for when you want to the style of the image to be like the text. like make it look like a painting. or make it like a robot.
Inputs/ Outputs
The input to this tool should be a comma separated string of two, representing the image_path and the text.
Example (Optional)
image/5b227bce.png, make it look like an oil painting
ChatGPT


History Of Dialogue $\mathcal{H}_{<i}$
Prompt Manager $\mathcal{M}(\mathcal{H}_{<i})$
User Query Q_i
Prompt Manager $\mathcal{M}(Q_i)$
Unique Filename
Visual ChatGPT generates a unique filename with a universally unique identifier (UUID).
Force VFM Thinking
Thought: Do I need to use a tool?
History Of Reasoning $\mathcal{R}_i^{(<j)}$
Prompt Manager $\mathcal{M}\left(\mathcal{R}_i^{(<j)}\right)$



VFMs Execute
Prompt Manager
$\mathcal{M}\left(\mathcal{F}\left(A_i^{(j)}\right)\right)$
Chained Filename
image/{Name}_{Operation}_{Prev_Name}_{Org_Name}
Name: a new uuid name for the generated image,
Operation: the operation
Prev_Name: the input image
Org_Name: the most original images' name
Ask for Details
When the user's command is ambiguous, Visual ChatGPT should ask the users for more details to help better leverage VFMs.
Iterative Reasoning of VFMs