Towards Autonomous Trash Collection: Exploring Prompt Caching and Feature-rich Synthetic Trash Data for Improving Integration of Large Language Models with Robotic Systems

Joe Lisk
Minnesota Robotics Institute
University of Minnesota Minneapolis, United States
liski003@umn.edu

ABSTRACT

This work demonstrated a proof-of-concept for utilizing prompt caching and feature-rich trash data for autonomous trash collection. Prompt caching is a technique in which exact or similar prompts are compared to access existing solutions from a large language model (LLM), which can be used with robotic systems. Prompt caching can be helpful when using LLMs in robotic systems by saving on the number of prompts for repeated tasks and reducing the uncertainty associated with the LLM's output. Likewise, it was discovered that using additional descriptors for objects can assist LLMs in determining which objects in a scene are trash. This work was built upon previous iterations of robotic trash collection. In previous iterations, GPT -4 was tasked with determining which objects in a scene were trash and providing code to actuate a Kinova Gen3 Lite robotic arm to pick up and dispose of the trash.

Index Terms—robotics, autonomy, object detection, human-robot interaction, large language models, trash collection

I. INTRODUCTION

A. Problem Motivation

Trash is a global problem that can affect both natural environments and cities. The scale of the trash problem and the difficulty in detecting, collecting, and sorting trash suggests that autonomous trash-collecting robots could one day reduce the growing trash problem. Whether an object is trash depends on several factors, such as the scene in which an object is found,

how long the object has been, and additional social factors. However, a current barrier to using robots and artificial intelligence for trash collection is that trash classification is inherently tricky because " trash " is subjective.

B. Shortcomings of Previous Work

The first iteration of the system [1] tasked GPT-4 with deciding which object was most likely to be trash between two objects in a scene. Once GPT-4 determined which object was trash, it was then asked to provide code to actuate a Kinova Gen 3 Lite robotic arm to pick up and dispose of the trash object. Given sufficient context about the objects, for example, describing one object as an "empty" can vs. a "full" can, GPT-4 was always able to correctly determine which object was most likely to be trash. However, even without variations in the prompt, GPT-4 would occasionally provide code output that would crash the program. One of the lessons learned was that re-prompting GPT-4 introduced cost and risk into the system. Through previous work, [1] [2], it was demonstrated that LLMs could be used to decide between objects in a scene, decide which objects in the scene are trash (if any), and provide code to actuate a robotic arm to pick up the trash object. Although the LLMs demonstrated the potential to perform this task successfully, this work aims to test the robustness of LLMs for trash classification and increase the difficulty in the scene descriptions. In [1], the LLM only had to decide between two objects with different descriptors (i.e., "full" and "empty"). In [2], the LLM was given the labels of objects detected with the YOLOv9 model. The second iteration introduced vision into the system to allow the detected object labels to be fed into GPT-4 to decide which objects were likely to be trash. Adding computer vision made the system less dependent on the human operator. GPT 4 was able to correctly determine which object was trash, even when the object detection model (YOLOv9) did not assign the correct labels to the objects. For example, it labeled the can as a bottle, but GPT-4 could still determine that it would likely be trash. Despite the system's overall success, the problem of the same prompt causing output that crashed the system still persisted.

Additionally, merely using the labels of detected objects can increase the chances of a false trash classification, which is why additional characteristics of an object and scene context are vital for successful trash detection. One of the main issues encountered with previous versions of the system was repeatedly prompting GPT -4 when solving the same problem. Repeated prompting increased the system's total cost by increasing the number of tokens used and the uncertainty of the output.

C. Novelty of This Work

This system introduces prompt caching, a technique where the system can quickly access previously found solutions if a new prompt is sufficiently similar to a stored prompt. Prompt caching was believed to be helpful when using LLMs for robotic systems due to the aforementioned issues with the previous two systems. As shown in the Related Work section, although current methods for prompt caching of LLMs exist, the novelty in this work was using prompt caching in robotic software to avoid re-prompting the LLM entirely.

Additionally, this work demonstrates that object detection is necessary but insufficient for autonomous trash collection. This work further demonstrates that more descriptive features of objects, such as physical characteristics, scene context, and time, can reduce the uncertainty of an object being trash. This work uses an "Indeterminate" category for trash classification that previous works have not explored. As shown in the Related Work section, most works restrict their trash detection tasks to object detection. Regarding autonomous trash collection, few, if any, previous works account for the subjective nature of trash, which this work attempts to do. As there are currently no datasets containing descriptive features for trash objects, Large Language Models were used as proof of concept to demonstrate how additional descriptive features can allow for autonomous trash collection in more environments.

II. RELATED WORK

A. Prompt Caching

Although OpenAI has its own form of prompt caching, there are several problems with relying solely on it. First, OpenAI's prompt caching method relies on an exact match of the prefixes of a prompt, which means that small variations in the prompt can break the cache. Second, although there is a reduced cost for cached prompts, the cost can still add up to a large amount of money with repeated prompts. Additionally, prompt caching does not entirely reduce the risk of an OpenAI LLM's response code crashing the robot program. Risk remains because while OpenAI caches the prompts, the LLM recalculates the response every time [3], which increases uncertainty. Ultimately, this uncertainty can limit the effectiveness and adoption of using LLMs in the robotics applications loop. These issues highlight the need for more control over prompt caching and more sophisticated prompt caching techniques. Further, other prompt caching techniques make prompts more modular through XML tags, which can be helpful when using open-source and open-weight LLMs [4].

B. Trash Detection

Datasets like TrashNet [5] and TACO [6] contain labeled trash objects. However, most trash datasets appear to focus solely on object detection and ignore the broader context of what makes an object "trash." Some datasets get around the fuzziness of trash classification by restricting the data to almost certainly trash objects, such as crushed aluminum cans and cardboard [5]. Some also restrict images to contexts where objects are likely to be trash, such as finding objects lying in natural environments [6]. A commercial robot, Glacier, is attached to a conveyor belt that sorts recyclable trash [7]. Additionally, some datasets contain images in an underwater environment [8] [9], which suggests that domain-specific trash detection is possible. When it comes to deploying trash-collecting robots in diverse public spaces, extra caution will need to be taken because not collecting objects that are not trash is just as important as

collecting objects that are trash. Although there are newer vision-language models that can provide some physical descriptors, such as the vision-language model PG InstructBLIP [10], few can provide the detailed descriptions and context needed to classify a variety of detected objects as trash in varied environments.

III. EXPERIMENT - MATERIALS & METHODS

A. Problem Description

a. Technologies Used

The system utilized the following technologies: Kinova Gen 3 for the simulation, Kinova Gen 3 lite for the simulation and the physical system, Ubuntu 20.04, Ros Noetic, Gazebo, RViz, Ros Kortex, MoveIt, Python 3, Open AI API, GPT-4, YOLO v9, Google COLAB, Logitech C920 webcam (see Fig. 9 for the system architecture).

b. Prompt Caching

The prompt caching experiment aimed to measure how much overlap existed between commands to "pick up trash" and sentences that did not have the same semantic meaning but might be confused as such. Claude AI [11] was used to generate 100 different phrases that have the same semantic meaning as "pick up the trash" and 100 different phrases that had different semantic meanings from "pick up the trash" but contained the word "trash." The "all-MiniLM-L6-v2" model [12], which is based on SBERT [13] from HuggingFace, computes the similarity score of the 200 examples. The all-MiniLM-L6-v2 model creates an embedding for each sentence where the embedding is represented by a vector, where the similarities can be compared by taking the dot product of the two vectors. Finally, the mean number of instances where non-semantically similar sentences had higher embedding similarities than semantically similar sentences was calculated as:

$$\bar{x} = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \mathbb{1}(Score(x_i) < Score(y_j))$$

Where each *xi* is a semantically similar sentence to "pick up the trash," each *yj* is a semantically dissimilar sentence, *n* is the number of semantically similar sentences, and *m* is the number of semantically dissimilar sentences. For the second part of the experiment, which was more of a proof-of-concept, a precomputed object detection solution was placed into a ROS node along with the grasp pose of the object. Both the object detection and grasp pose were from previous works [1] [2] and were obtained from YOLO v9 [14] and Gazebo/RViz modeling, respectively. The algorithm was:

$st \leftarrow 0.30$
$X \leftarrow cachedSentenceEmbeddings$
$p \leftarrow userPromptEmbedding$
$returnVal \leftarrow NULL$
for xi in X **do**
  $si \leftarrow similarityScore(p, xi)$
  if $si \geq 0.30$ **then**
    $returnVal \leftarrow cachedActionSequence$
  end if
end for
return $returnVal$

Figure 1. Prompt Caching algorithm in ROS

The solution of an LLM prompt was stored in JSON memory, and the system was tested live. Due to the quadratic scaling in computational complexity for sentence transformers [15], the prompts were truncated, and only their first part was compared. One-shot prompting was employed, which tends to lead to more accurate code from the LLM [16]. The results of the second part of the experiment are shown in the video. The form of the correct solution was:

```
cachedSolution = {
  "prompts": [
    "pick up the trash",
    "take out the trash",
    "remove the trash",
    "take out the garbage"
  ],
  "actions": [
    'sceneObjects["gripper"]["open"]',
    'sceneObjects["can"]["go_to"]',
    'sceneObjects["gripper"]["close"]',
    'sceneObjects["trash_bin"]["go_to"]',
    'sceneObjects["gripper"]["open"]',
```

Figure 2. Cached prompts and Action Sequence

The "actions" list contains actions that move the Kinova arm to precomputed locations and open/close the gripper.

c. Trash Classification

This experiment aimed to demonstrate the subjective nature of trash, the insufficiency of relying on object labels in environments with a significant amount of human activity, and that richer feature descriptions can make trash classification more certain. The first part of the experiment consisted of twenty trash classification prompts with three layers of difficulty. The first tier was a simple binary problem with only two objects, one of which was trash. The second tier consisted of several objects and was based on how a computer vision system would detect objects and provide labels. The third tier was created jointly with GPT-4o [17] and contained trash and non-trash objects that might be found in an airport. GPT 4o was given examples of problems related to autonomous trash collection with a progressive difficulty scale for all three tiers. GPT-4o then generated 20 prompts based on these examples. In the system prompts for tiers 2 and 3, the LLM Llama 3.2 Instruct 8B [18] was told that it was to act as a trash-collecting robot in an airport and that not taking people's belongings was just as important as detecting trash. An airport setting was chosen because trash classification is more ambiguous and context-dependent, with a penalty for mistakenly collecting personal belongings as trash.

```
System prompt for Llama 3.2 8B Instruct:
(formatted for readability)

”You are a robot in an airport. It’s
important you both collect trash and
avoid taking peoples belongings”
```

Figure 3. System Prompt for Trash Detection

d. Researcher example prompts

```
Tier 1 – Binary choice:

"There are two objects in
the environment: an empty can of
soda and a full can of soda. Which
object is most likely to be trash?"
```

```
Tier 2 – Multiple Objects,
Simple Descriptors:

"Your object detection system
found the following objects
in the scene: 1 person, 1 bottle,
2 bowls, 1 sandwich, 2 chairs,
2 dining tables. Which of these
objects is most likely to be trash?"
```

```
Tier 3 – Multiple Objects,
Realistic Scene,
Rich Descriptors:

"In the airport terminal, there are
scattered sheets of paper with coffee
stains, a set of mittens soaked in coffee,
and a stainless steel coffee mug on its
side, all lying next to an open suitcase
that appears to have fallen over. Which
of these objects is most likely
to be trash?"
```

Due to the increasing complexity of the trash classifications, the focus shifted away from correct/incorrect for tiers 2 and 3 but focused on the number of indeterminate objects classified as trash. First, the human researcher determined which objects in each scene were indeterminate. Next, GPT-4o reviewed the examples and determined which objects were trash, non-trash, and indeterminate. The indeterminate objects for each example were the union between the researcher and GPT-4o. Then, Llama-3.2 8B was tasked with classifying the objects in each scene as trash but was not asked to determine if any objects were

indeterminate. This experiment tested Llama 3.2 for trash classification and GPT-4o for classifying objects as indeterminate. The second part of the experiment, which was more of a proof-of-concept, assessed how human-robot interaction could improve the trash classification of indeterminate objects. A simple example prompt was given in the style of a tier 1 problem, with two cans in the scene without any additional context. GPT-4o was tasked with determining which of the two cans was trash and to ask a human if it could not. After receiving the response from the LLM, the human researcher added additional context about the two objects to improve the trash classification. The results of the second part of the experiment are shown in the video.

B. Results

a. Prompt Caching

For prompt caching, the similarity scores revealed some embedding similarities between trash and non-trash sentences (Fig. 5). Around a similarity score of 0.3, trash sentences became more numerous than non-trash sentences.

Summary Statistics for Embedding Similarities

Stat	Trash	Non-Trash
Min	0.060	0.096
Max	0.836	0.497
Mean	0.425	0.276
StdDev	0.139	0.080

Figure 4. Summary Statistics of Embedding Similarity to the sentence “pick up trash”

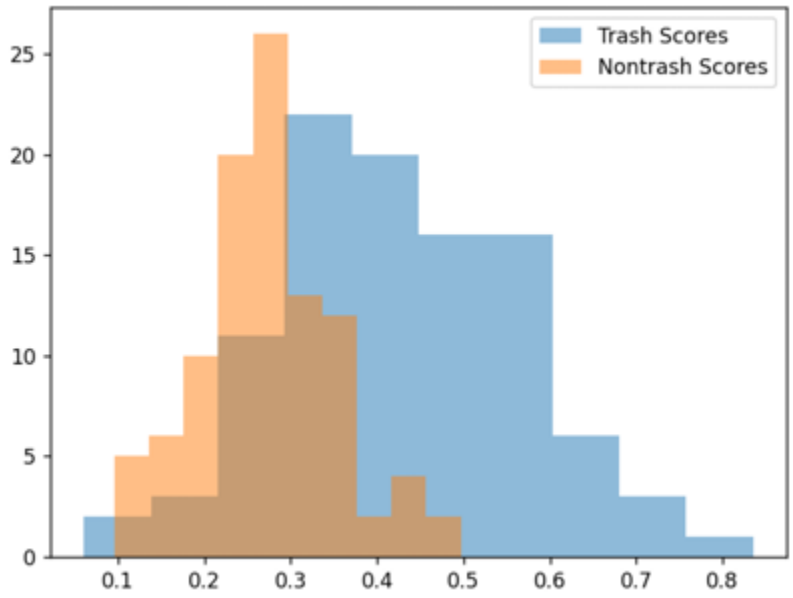

Figure 5. Distribution of Embedding Similarity scores to the sentence “pick up trash”

b. Trash Classification

Total Correct Trash Classification %
Tier 1
95.24

Total Number of Indeterminate Objects Classified as Trash	
Tier 2	Tier 3
21	12

Total Number of Indeterminate Objects	
Tier 2	Tier 3
52	30

Total Number of Objects	
Tier 2	Tier 3
89	75

Proportion of Indeterminate Objects %	
Tier 2	Tier 3
58.43	40.00

Proportion of Indeterminate Objects Classified As Trash / Indeterminate Objects %	
Tier 2	Tier 3
40.38	40.00

Figure 6. Correct Trash Classification % (Tier 1) and Proportion of Indeterminate Objects (Tier 2 and Tier 3)

The proportion of indeterminate objects and indeterminate objects classified as trash was higher for the tier 2 set of questions. A proportion test would need to be performed to determine the statistical significance of the difference. However, the results of the proportion test would be suspect, as the data were primarily generated from GPT-4o (see Discussion). For the second tier of questions, in the absence of more detailed feature descriptions, Llama 3.2 8B Instruct hallucinated its own. For example, the prompt "The detected objects in the environment are: 1 computer, 1 broken mouse, 1 plant pot, and 2 pieces of cardboard. Which object is most likely trash?" yielded the response "...The 2 pieces of cardboard are also likely trash because they are

no longer needed or useful and are likely to be discarded...". In the third tier of questions, although the questions were more complex than the previous tiers, richer descriptors and context for objects led to some examples with no "Indeterminate" classifications. However, there were some examples that both GPT-4 and Llama-3.2 struggled with. For the prompt, "In the airport terminal, there are scattered sheets of paper with coffee stains, a set of mittens soaked in coffee, and a stainless steel coffee mug on its side, all lying next to an open suitcase that appears to have fallen over. Which of these objects is most likely to be trash?" both GPT-4o and Llama 3.2 Instruct 8B stated that the papers were likely to be trash. Neither model considered the papers possibly spilled out of the open suitcase and may be important documents.

IV. DISCUSSION

A. Issue with Evaluating LLM output

Statistical analysis of the output of Llama 3.2 8B was complicated due to most statistical methods relying on the independent and identically distributed (i.i.d.) assumption, which LLM output is not [19]. Though there are nonparametric statistical tests such as the Kruskal-Wallis test, they would have been less helpful in interpreting the LLMs output. Additionally, as the complexity of the scene descriptions in the prompts increased, the number of "indeterminate" objects (i.e., the object could be trash or not trash based on additional context) increased, which made grading the LLM output difficult due to the lack of a "correct/incorrect" dichotomy. Vector similarity measures, like cosine similarity, also would not have provided much information outside of testing how similar the prompts and the outputs were. Because of these issues, the LLM output was evaluated with spot-analysis, such as looking at the number of indeterminate objects in a prompt vs. the number of indeterminate objects that the LLM classified as trash.

B. When Are Two Problems No Longer Similar?

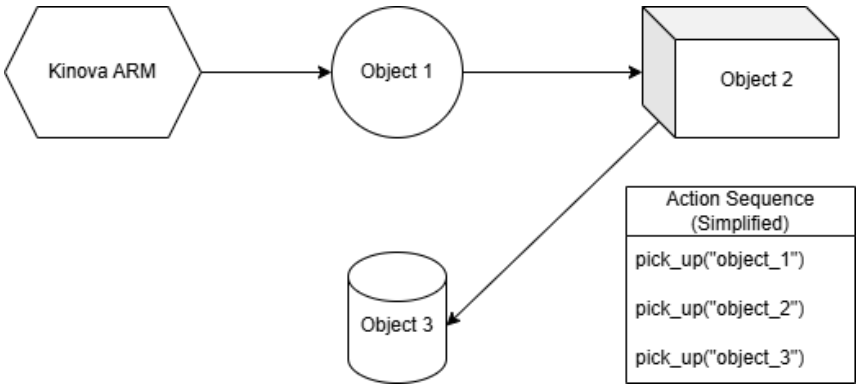

Figure 7. Example layout of objects with Action Sequence to pick up objects

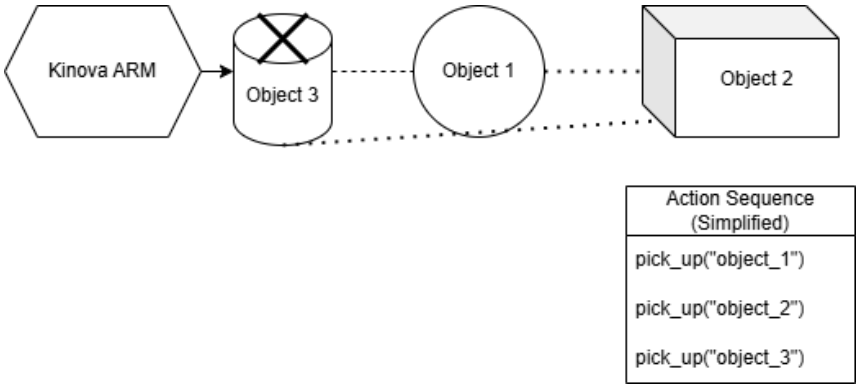

Figure 8. Using the Action Sequence from Figure 7 may cause a collision if the object layout changes

Saving the solved action sequences from cached prompts opens up a potential for error if the same objects are in the scene. However, the layouts are significantly different, so it may be possible that the cached action sequence is no longer a valid solution. For example, suppose there is a scene with three objects. In that case, their layout may be substantially different in another scene, which may necessitate a change in the order in which the objects are picked up due to factors like collision avoidance. Fig. 8 demonstrates the problem, where the action sequence of the first problem (Fig. 7) causes a potential collision in the second problem. Although, in some instances, trash does not need to be handled with care, the potential for

hazardous waste and the possibility of damaging the manipulator's arm necessitates a critical look at the cached action sequence. This issue can be mitigated with strategies like vision-based control for collision avoidance and testing permutations of the action sequence in simulation.

C. Limitations of Current Trash-Detection Datasets

Although this work demonstrated how richer object and scene descriptions could improve trash classification, it currently relied on LLM-generated synthetic data that doesn't yet exist. Another layer of feature descriptors that were not tested may also exist, potentially further improving the ability to perform trash classification on detected objects.

V. FUTURE WORK

Adding a penalty for incorrectly grabbing people's belongings could form a basis for adding feedback into the system with reinforcement learning. A classifier like SVM or MLP can be trained based on the embedding similarity scores to make prompt caching more robust. Additionally, an in-house LLM or one that is open-source or open-weight could be used. For classification methods, additional features beyond vector similarity scores (such as n-grams) would likely need to be added beyond vector similarity scores (such as n-grams) because of difficulties classifying 1-dimensional data when classes overlap.

VI. CONCLUSION

This paper demonstrated how prompt caching can reduce the risk of using LLMs in robotics systems. Additionally, this work revealed how scene context and richer object descriptors can assist in classifying objects as trash. Although the prompt caching mechanism is simple and relies on a single metric (embedding similarity), it can safeguard against the nondeterministic nature of LLM output. This framework of descriptive features could also be useful for other

tasks, such as robot chefs preparing complex recipes that require outside context and interpretation, like distinguishing between several kinds of cheese. Although datasets containing rich object descriptions do not yet exist, this work will likely motivate future dataset creation directions.

.

VII. APPENDIX

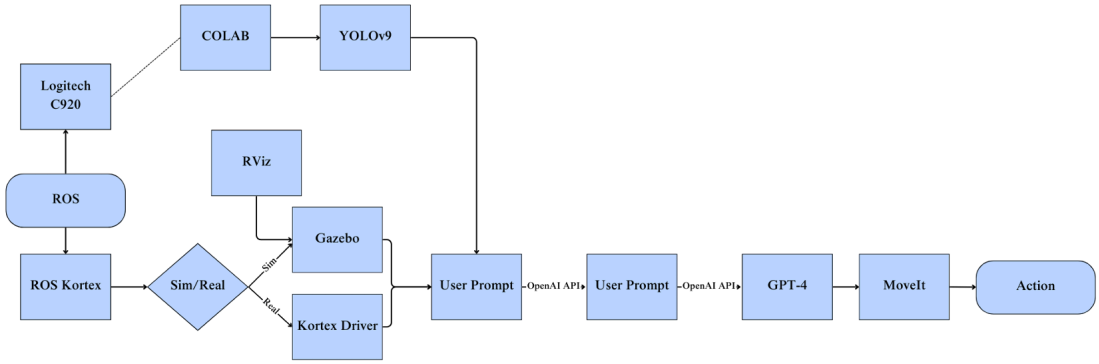

Fig. 9 Control scheme of the full system

REFERENCES

[1] J. Lisk, “Text-to-action: Assessing the role of large language models in high-level robotic control.” None. Accessed: 2024-12-21.

[2] J. Lisk, “Vision-to-text-to-action: Combining vision and language for semi-autonomous trash collection.” None. Accessed: 2024-12-21.

[3] OpenAI, “Prompt caching in the api.” https://openai.com/index/api prompt-caching/. Accessed: 2024-12-02.

[4] I. Gim, G. Chen, S. seob Lee, N. Sarda, A. Khandelwal, and L. Zhong, “Prompt cache: Modular attention reuse for low-latency inference,” 2024.

[5] G. Thung and M. Yang, “trashnet.” Accessed: 2024-12-22.

[6] P. F. Proenc ¸a and P. Sim˜ oes, “Taco: Trash annotations in context for litter detection,” 2020.

[7] Glacier, “Glacier: Recycling robots to end waste.” https://endwaste.io/. Accessed: 2024-12-21.

[8] M. S. Fulton, J. Hong, and J. Sattar, “Trash-icra19: A bounding box labeled dataset of underwater trash,” 2020. Retrieved from the Data Repository for the University of Minnesota (DRUM).

[9] J. S. Walia and P. L. K, “Deep learning innovations for underwater waste detection: An in-depth analysis,” 2024. Fig. 5. Updated control scheme

[10] J. Gao, B. Sarkar, F. Xia, T. Xiao, J. Wu, B. Ichter, A. Majumdar, and D. Sadigh, “Physically grounded vision-language models for robotic manipulation,” 2024.

[11] Anthropic, “Models- anthropic.” Accessed: 2024-12-21.

[12] “sentence-transformers/all-minilm-l6-v2.” Accessed 24.12.2024.

[13] N. Reimers and I. Gurevych, “Sentence-bert: Sentence embeddings using siamese bert-networks,” in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 11 2019.

[14] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, “Yolov9: Learning what you want to learn using programmable gradient information,” 2024.

[15] H. Wang, Y. Fan, M. F. Naeem, Y. Xian, J. E. Lenssen, L. Wang, F. Tombari, and B. Schiele, “Tokenformer: Rethinking transformer scaling with tokenized model parameters,” 2024.

[16] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, “Language models are few-shot learners,” 2020.

[17] OpenAI, “Gpt-4o system card,” 2024.

[18] AI@Meta, “Llama 3 model card,” 2024.

[19] J. Boyd-Graber, “Do iid nlp data exist? [lecture],” 2023.

[20] Kinova®, Kinova® Gen3 lite robot user guide.