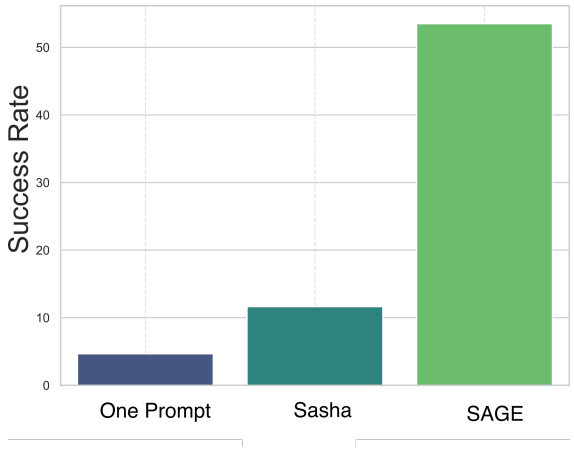

Figure 7: Overall success rates on 43 challenging tasks. SAGE leverages a collection of tools that allow it to integrate a large amount of grounding information into its decision making process, allowing it to outperform other LLM-based methods by a significant margin.

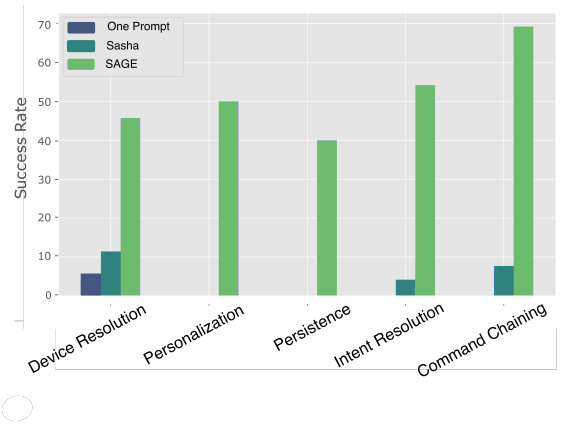

Figure 8: Success rate per task challenge type.

etc). Despite this, the baselines allow the reader to gauge the difficulty of the task set, and to appreciate the extent to which integrating information from a variety of sources can improve the performance of smart home automation systems. We do not provide a baseline with access to the same information as SAGE as, to our knowledge, there isn't any previous work that is capable of integrating all of these information sources.

7 Results

Overall success rates for the three methods, SAGE, one prompt, and Sasha, are presented in Figure 7, and success rates per task challenge type are presented in Figure 8. SAGE achieves an overall success rate of 51%, far beyond either of the baselines, demonstrating that it is indeed capable of integrating a variety of information sources through the use of its tools.

To further understand the reasons for failure in SAGE, we manually analyzed the failing cases, categorizing the failures according to the component causing the issue. These results are summarized in Figure 9, where each bar represents the ratio of tests failing due to failure of the component vs total tests using the component. Note that it is common for components to be reused multiple times within the execution of a single command, but these multiple uses are not reflected in Figure 9. Figure 9 shows that by far the least reliable part of the system is the condition code writing tool, which also helps to explain why persistent commands had the lowest success rate in Figure 8. We believe this to be in part due to issues with prompt length – each time the code writing agent makes a mistake which leads to an exception in the code, a copy of the code is added to the action history. Code takes up a relatively large number of tokens, and as the amount of code in the prompt becomes large the reasoning abilities of the model seem to degrade (as documented in [24]). Device disambiguation could also be improved, for example by using a better CLIP model or by using a captioning-based solution.

8 Conclusion

This article introduced SAGE, a grounded agent targeted at smart home applications. SAGE achieves its grounding by orchestrating the use of tools in a sequential decision making process. To enable SAGE to