III. SYSTEM DESIGN

devices

"room 1"

"lights"

"lamp"

"state"

"on"

In this section, we introduce the system design that we use

to explore the feasibility of LLM-driven smart home control.

"tvs"

"r"

255

We first assume the use of an LLM like GPT-3 that provides

responses to user prompts written in natural language. These

"room

"g"

255

LLM models are not task-specific, rather, they are trained

user

"location"

on an immense amount of cross-domain textual information

"room_1"

"b"

255

and, depending on the structure of the prompt, can provide

Fig. 1: Data structures for expressing smart home device and

responses suited to a variety of different use cases (e.g., writing

user context in prompts to an LLM.

a poem, writing code in response to a high-level program

description, etc.). We opt to adapt the model's outputs to our

task using zero-shot learning through prompt engineering.

color values. This overall structure is depicted in Fig.

1

and

Our challenge is therefore to package relevant context and

illustrated by the example in the following:

user commands into a concise prompt issued to the model,

such that its responses include concrete, machine-parseable

{

changes to device state that can be passed off to the appropriate

"user" {

smart device APIs. Qualitatively, we want these courses of

"location" "living_room"

action to be shaped by the model successfully inferring (1) the

intent behind the user command and (2) the manner in which

}

the state of available devices can be changed to meet the user's

{

intent. To that end, we first define an abstract schema for

"devices": {

capturing smart home context before describing a method for

"bedroom" {

engineering prompts to conversational LLMs that elicit useful,

"lights" {

actionable responses.

"bedside_lamp" {

"state": "off"

A. Context Schema

}

In order for the model to "know" what actions are available

to it, we need to package the available devices, their states, and

},

other relevant information-i.e., the context-into a machine-

"living_room" : {

parseable format. This package effectively describes the action

"lights" {

space available to the model: the knobs it can turn, and

"overhead" : {

information (e.g., which room the user is in) that might

"state" "on"

influence how it turns them. It also provides a hint about

},

how the model should format its response. Representations

'lamp" {

of context can be complex and have been explored in the

"state": "off"

literature 9 1 Since our goal is to conduct an exploratory

}

study rather than design an end-to-end solution, we use a

},

schema that is simple but adequate for our experimental setup.

"tvs" {

We choose JSON for structuring this data since it is the de-

"living_room_tv": {

facto data interchange format for RESTful APIs used by many

"state" "off",

smart home devices 19 8 10 Leveraging a common format

"volume" 20

is also advantageous since there is a high likelihood that the

}

LLM has been trained on source material that contains it,

which benefits the model's ability to converse in it.

At the top level, context is a collection of "key, value" pairs.

There are two relevant contexts: "user" context that contains

}

immutable information about the user's state, e.g., which room

In this example, the user's home has two rooms-a bedroom

they are in, and "device" context, which contains mutable and

and living room-and the user is currently located in the living

immutable information about the devices in the home. Each

room. The bedroom has one light turned off, and the living

top-level key in the collection of device context defines a room

room has two lights (one turned on) and a television.

in the home, and within each room we define collections of devices organized by type, e.g., "lights", "tvs", etc. Within

B. Prompt Engineering

a collection of devices, we define each individual device as

Having developed a structure for storing context, we now

a collection of properties about that device, e.g., for a light

move to the practical challenge of engineering prompts that

we can define its "state" property and its "r", "g", and "b"

elicit useful responses from the model.