智能体实现

总述

语音唤醒智能体，通过语音管理任务列表。

智能体识别并格式化任务，在 DDL 前一段时间提醒。

对于时间占用型任务，需要考虑是否会冲突，整日行程安排

- 整日行程安排
  - 地点变化
  - 交通工具准备
- 全能生活助手
  - 语音打开乘车码
- 多人联机

模块化

Demo1 (ZCX)

- 你有的数据(获取方式见数据库接口)
  1. 历史对话信息. (table: "dialogue", key: 数字, value: "[发送者]:[内容]")
  2. 你自己保存在数据库里的信息.
- 你有的功能:
  1. 你可以在数据中创建 table 并操作你创建的 table, 用于永久保存你所需的信息.
- 你要实现的功能:

1. 一个函数 next_activate, 没有参数, 用于根据数据库里的信息确定当次 "主动唤醒" 的行为和下次 "主动唤醒" 的时间.
   Case1: 根据逻辑判断当前为静默唤醒 (False, None, Time)
   Case2: 根据逻辑判断当前为活跃唤醒 (True, AI主动交流内容, Time)
   eg1. (调用时恰好是中午) -> (True, "喵喵喵, 主人记得吃饭哦.", Time)
   eg2. (调用时发现某个 Task 的开始时间快到了) -> (True , "主人记得去开会哦, 今天下午五点", Time)
   eg3. (调用时发现最近的对话信息中用户情绪不对劲) -> (True, "有什么难过的事情吗?小妍可以听听吗?一个人憋着很难受吧...", Time)
   eg4. (调用时发现没什么可说的) -> (False, None, Time)

Demo2 (JHT)

- 你有的数据(获取方式见数据库接口)
  1. 历史对话信息. (table: "dialogue", key: 数字, value: "[发送者]:[内容]")
  2. 任务安排数据库.
- 你要实现的功能

1. 一个类 (UserTask), 用于表示一个任务.

- 提供一个成员函数将它在数据库中保存或者删除.
- 能够支撑你完成下面的几个功能

2. 一个函数 (try_del_task), 参数是一段文本, 用于判断这段文本是否是一条删除任务的指令, 并根据数据库中的信息确定具体删除哪个任务, 需要考虑对话历史上下文.
   Case1: 不是指令 None
   Case2: 是指令
   SubCase1: 合法指令(对应任务存在) (True, 删掉的任务, AI回复)
   SubCase2: 非法指令(对应任务不存在) (False, None, AI回复)
   eg1. "明晚我要去和同学聚餐, 会议安排取消了" (数据库中找得到关于明晚会议的信息) -> (True, UserTask, "小妍已经把明天的会议从安排中去掉了喵.")
   eg2. "明晚我要去和同学聚餐, 会议安排取消了" (数据库中找不得到关于明晚会议的信息) -> (False, None, "喵? 可是小妍没有找到主人说的会议呢?")
   eg3. "会议安排取消了, 明晚我要去和同学聚餐" (数据库中找不得到关于明晚会议的信息, 但有一个后天晚上的会议) -> (False, None, "喵? 可是小妍没有找到主人说的会议呢, 哦哦, 这里有个后天晚上的研讨会, 主人是把这个会议取消了吗")
   eg4. "嗯" (上下文承接 eg3) -> (False, UserTask, "好哒喵, 已经删了, 主人生活愉快哦喵~")
   eg5. "同学聚餐我不去了" (上下文承接 3.eg2) -> (True, UserTask, "唉, 工作所迫, 小妍只能面为其难的帮你取消了.")
   eg6. "你好呀" -> None
3. 一个函数 (try_add_task), 参数是一段文本, 用于判断这段文本是否是一条加入任务的指令.
   Case1: 不是指令 None
   Case2:
   Subcase1: 任务能正常添加 (True, UserTask, AI回复)
   Subcase2: 任务不能正常添加 (False, None, AI回复)
   eg1. "我明晚有个研讨会" -> (True, UserTask, "好的, 小妍已经安排进时间表了")
   eg2. "我明晚有个研讨会" (数据库中明晚有一个聚餐了) -> (False, None, "主人, 明天晚上的同学聚餐和研讨会冲突了, 再考虑一下吗?")
   eg3. "同学聚餐我不去了" (上下文承接 eg2) -> None

Func1 (ZXJ)

完成一个数据库, 支持多种类型的数据插入, 查询, 删除.

1. 总览
   1. 包含若干个 table, 每个 table 本质是一个字典, 通过唯一的 key 对应 value.
   2. key 的格式由你来规定 (一般规定成**任意**只包含大小写字母和数字的字符串即可), value 的格式一般也是字符串.
   3. 提供一个内置的 "Task" table, 用于存储任务, 此时你可以规定 value 采用一个什么样的格式.
   4. 支持用户自定义 table, value 的格式是一个任意字符串.
2. 插入:
   1. 通过 table, key, value 来插入一项数据.
   2. 如果 table 不存在, 需要返回错误信息.
   3. 如果 key 有重复或者 value 不满足对应格式, 需返回错误信息.
3. 查询:

    1. 通过 table 和 key 来查询特定数据.
    2. 通过 table 来查询这个 table 中所有数据.
    3. 如果 table, key 不存在, 需要返回错误信息.
4. 删除:
    1. 通过 table 和 key 来删除特定数据.
    2. 如果 table, key 不存在, 需要返回错误信息.
5. 创建:
    1. 通过 table 来创建一个自定义 table.
    2. 如果 table 已经存在, 需要返回错误信息.

Func2 (MZX)

实现语音监听唤醒功能.