关于rasa的学习心得

Rasa介绍

它是一个对话机器人的开源框架，稳定性，可读性较好

它的优点：一方面在于能在本地运行，从而获取大量可用训练数据，另一方面在于它灵活性，组件可定制化。

缺点：配置繁琐。

对于目前的智能机器人的五个层级通知、知识问答、上下文、个性化、自组织，Rasa最多达到上下文（多轮对话）层级。

Rasa主要三个组件rasa主体部分、rasa-x(它是一个外部UI)、

rasa action service(它是一个单独的部分，能跟rasa框架进行交互)组成。

Rasa主体部分包括：

NLU:

跟之前的pipline的NLU类似

Core

跟之前的DM的模块类似

Channels and integrations and action server

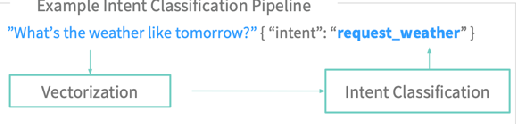
主要用于帮助用户跟第三方工具对接

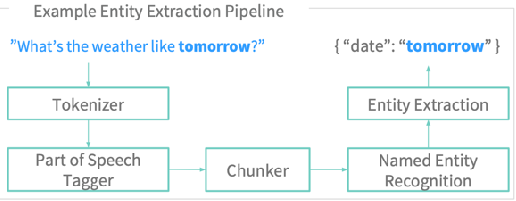
NLG

跟之前的pipline的NLG类似。

NLU:

主要目的是为了生成intend和Slot/Entity 。它的任务类似一个文本、NER、序列标注等等





当我们配置NLU需要在config.yml中的pipline里面

Rasa的训练数据在于data目录下的nlu.yml文件，这个需要我们配置intent和entity（同义词、正则表达式、查表、Role & Group 等等）。

intent注意英文书写。

NLU可以单独训练。

rasa train nlu

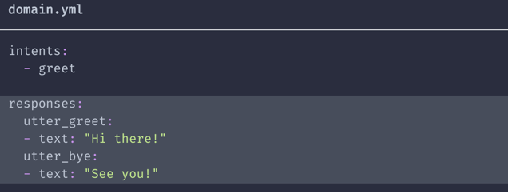
Core:

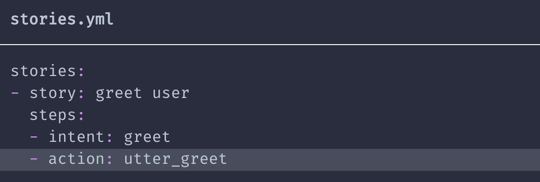
它包括DST跟DP部分，它的核心在于预测 action。它提供了四类action。



它的配置可以在于domain.yml，也可以在stories.yml里面

它是包括intents跟responses。它的action可以使用变量、Channel、rich response（buttons,image）





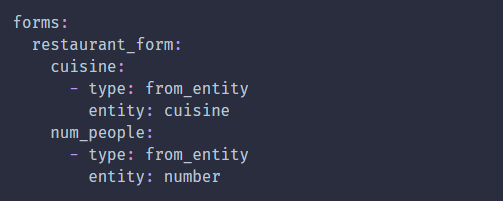


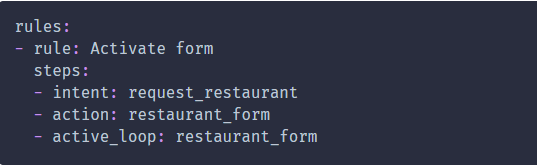


适合任务类型，填写表单，在domain.yml文件里面配置

激活Form需要在config.xml中的polices增加。它可以作为story加入

stories.yml，同时增加active\_loop（null值则为退出）

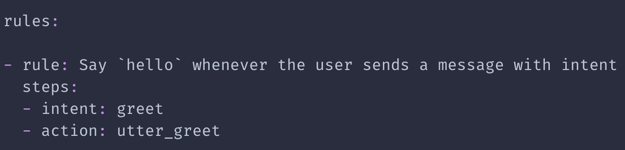






**Rule**

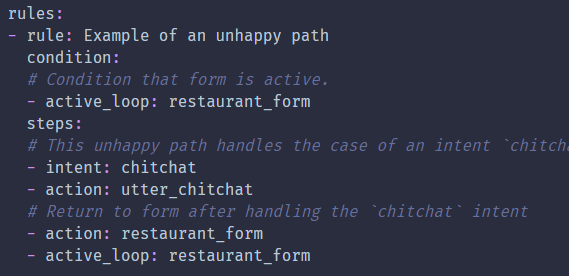
**它也需要在config.xml中的polices增加配置。书写在**domain.yml里面



它与story的区别在于它是完全按照规则来写的，不具有泛化性

story的灵活性较强，具有泛化性。具有步骤性。先执行rule，后story

它可以与带有条件性。



Core也可以用来训练

rasa data validate stories

rasa train core