MusicBot Milestone 2

Members：王耀賢 陳品融 陳聖曄 李俊鈺 謝明勳

Multi-turn Interaction

在這次的milestone 2 中，我們實做了可以multi-turn interaction的系統，系統可以大致分為3個部份，分別為user simulation，language understaing (LU)，以及dialogue management (DM)。

互動由user simulator開始，首先user simulator會先初始此個對話的user goal，接著根據初始的user goal，user simulator便會依序產生句子給dialogue system。dialogue system包含LU以及DM，user simulator產生的句子會先傳給給LU，LU會判斷出句子的各個intent以及slot的機率分佈給DM。根據這個機率分佈以及上一個time step的state，DM會計算各個state的機率，並以rule-based的dialogue policy決定系統下一步的action。接著DM會把action frame交給user simulator，user simulator則會根據DM的回答說出句子以便達成他初始的user goal。

這樣對話的過程會持續到一個對話結束。一個對話結束代表滿足一個user的intent，並且找出所有這個intent所需要的slot。一個對話結束後理應user要再繼續達成他其他的intent，像是他會先要求推薦首歌再說要播放歌曲，只是我們目前的user goal只有一個user intent，所以會達成一個user goal對話就結束。此外，若是一個對話太長(可能因為對話系統不論如何都無法辨識，導致一直循環)，對話系統就會直接結束對話。

Ontology

1. Slot:
    a. Artist: 演出者
    b. Genre: 曲風
    c. Track: 歌曲
2. Intent:
    a. Search [artist,track]:
    僅輸入track，或者artist跟track，再根據artist或者track來搜尋並播放歌曲
    b. Recommend [artist,track,genre]: 可以輸入1~3個slot，根據artis或者track或者genre來推薦與這些slot相關或者類似的音樂
    c. Info [artist or track]: 只能輸入一個slot，根據歌曲提供歌曲演出者、專輯等等資訊，或者根據歌手提供他熱門的歌曲。

System Dialogue Action

DM會做出的行為如下，詳細內容於Dialogue Policy。

1. Confirm: 對於intent或slot的內容進行確認
2. Question: 詢問intent或是slot的內容
3. Response: 系統對於search做出的最終回應
4. Info: 系統對於recommend和info做出的最終回應

User Simulation

會產生的user goal如前述，包含：search, recommend 和 info。

1. Implementation

Initialization

首先，在init的階段，會讀取template的句型。接著，根據給定的intent和 slot

(artist, track, genre)資訊，設定此輪對話的user goal。

實作上，user simulator都是被動的做出回應。因此在user simulator此輪對話的user goal設定完成後，並不會主動開啟對話。要使user simulator開啟對話，需要傳給user simulator { 'action' : 'question' }，user simulator便會從對應 intent 的 templates當中，隨機選取一個，然後將slot填入來做成一個句子。而這個句子也就會是user simulator 所開啟的對話中的第一句話。

Responses

對於DM傳來的不同的action frame (question, confirm, response, info) 分別的處理方式如下：

1. Confirm: 根據DM傳來的intent或slot，和目前所設定 user goal 的 intent 和 slot 進行確認。
    a. 如果全部正確，便會回答 "是的"，"沒錯" 等正面的回應。
    b. 若沒有全部答對：
        i. 要進行確認的slot在初始時的user goal當中並不存在 (根本沒有這項constraint)，回答 "不是", "錯的" 等負面回應。
        ii. 要進行確認的slot都存在，但是slot的內容是錯誤的，便回答 "不是", "錯的"，並給予正確的slot資訊，如 "不是 我想要聽的是林俊傑的歌"。

2. Question: 根據DM詢問的intent或slot進行回應。
    a. 如果僅僅詢問intent，則是從符合user goal intent和slot的 templates當中隨機選一個生成句子回答。
    b. 詢問slot
        i. 先判斷所詢問的slot是否存在於user goal當中，如果沒有的話回答 "不是"，"錯的" 等負面回應
        ii. 如果詢問的slot是存在於user goal當中的，則根據在templates當中找包含此slot 的template，生成句子回應。
3. Response: 對話結束，計算reward。
4. Info: 對話結束，計算reward。

2. Simulator Goal

Search和info這兩個intent的目的非常明確，只要系統有完全整握user goal的
intent 和 slot 即是完成目標。

然而要衡量recommend則較為困難，難以判對系統所推薦歌曲的優劣。為簡化此一問題，我們假設Spotify 的 recommendation API能夠有很好的推薦歌曲，因此將recommend的目標同樣定為，系統若能夠抓取正確的 intent 和 slot 即為成功達成目標。

3. Error Handling

user simulation 會對 confirm 和 question 這兩個action的error進行處理。

Confirm

如果系統confirm某個intent或者slot跟user simulator的goal有所牴觸，user simulator就會先否定這個confirm再說出他所想要的目標
Example:
(User Goal: intent:search, slot:{track:修煉愛情})
DM: confirm[track='可惜沒如果']
Simulator: "不對 我想聽修煉愛情"

如果系統confirm的slot並不存在於user goal當中，user simulator會給予負面的回應
Example:
(User Goal: intent:search, slot:{track:修煉愛情})
DM: confirm[artist='林俊傑']
Simulator: "不是"

Question

如果系統詢問某個user goal不存在的slot，simulator會給予負面回應。
Example
(User Goal: intent:search, slot:{track:修煉愛情})
DM: question[artist]
Simulator: "錯了"

Language Understanding

這次LU的model與milestone1 一樣，仍然是bidirectional lstm的seq2seq model。然而這次可以支援4種不同的intent，分別是search, recommend, info以及neutral，slot總共有artist, track, genre。比較不一樣的事，原本是直接取argmax來predict slots的tags，而現在爲了要輸出機率讓之後的dialog state tracking能據此來作判斷，我們是看LU model predict出來非'0'的slot，把這些slot output出來的logits通過softmax得到機率，這些機率連乘得到的我們便可視爲某個詞屬於哪個tag的一種信心指數。此外，原本直接輸出intent也會變成機率的一個distribution。

Dialogue State Tracking

1. Definition of State:
    a. State: 由上一個time step的action，上一個time step的distribution state，目前user的response，以及confirmed state所決定的。
    b. Distribution state: 是指從對話開始，到目前為止所有intent,以及提過的slot value他們在每個time step LU出來的機率總和，舉例來說"不為誰而做的歌"這個slot出現過兩次，機率分別是0.7,0.5，則在distribution state上他的值就是1.2。
    c. Confirmed State: 被寫入confirmed state的intent或者slot代表系統完全確定這個intent或者slot的值，如果某個類別的slot的值= -1代表系統確認user不需要這個slot。
    Example:
    confirmed state{intent=recommend,track=-1,genre=rock,artist=-1}
    代表系統認為intent是recommend，genre是rock，其他slot都不需要填值
2. State Update:
    a. Distribution state: 每次NLU的結果都會被寫入Distribution State，然而若是他跟user所confirmed的結果是"不是"，則被confirmed的intent或者slot的distribution state會被重設為0
    b. Confirmed State: 被寫入confirmed state有二擇一的條件，第一他的機率必須大於某個threshold，又或者DM已經跟user confirm過這個slot或intent。

我們threshold設1.9，因為LU出來的機率大部分都是0.999，因此設1.9才能比較有機會看到他會做confirm的動作

3. Implement:
   因為沒有data，因此我們使用rule-based來實做Dialogue Manager，更新規則如上所述

Dialogue Policy

1. Action:
   a. Confirm: 若是目前intent或者slot的機率高於某個 lower threshold，但是低於upper threshold就要做confirm，因為他不夠確定這個是否正確
   b. Question: 若是DM所的state上所有intent的機率都低於lower threshold，就會question[intent]，若是已經決定intent，而這個intent所需要的有些slot尚未被填入value，就會question某個類別的slot
   c. Response: 如果DM成功的得到search所有需要的所有slot，他就會根據這些資訊做出response，目前的response只有播放歌曲
   d. Info: 提供推薦的結果；或者info這個intent根據slot搜尋的結果，如果在info這個slot填入歌手他會說歌手資訊，填入歌曲他會說這首歌曲的演唱者專輯

2. Policy:
   a. 如果LU出來的intent或者slot機率高於upper threshold，則DM就會當作確定user的intent或者slot，就不會再confirm或question已經確定的intent或slot
   b. 如果 LU出來的機率低於lower threshold則直接當作沒有這項intent或者slot
   c. 如果LU出來的機率低於upper threshold但又大於lower threshold就做confirm
   d. 如果現在沒有東西需要confirm，然而有intent沒填入，或者intent已經填入，他所需要的slot沒填入，就做question
      Example:
      在系統一開始執行的時候
      User: 你好啊 今天天氣不錯
      Dialogue System: question[intent]
      -> 因為系統不知道”你好啊 今天天氣不錯”的intent，因此他會question user
   e. 如果所有需要的intent以及slot都已經確定他的值了(確認user不需要某個slot也算是確認)，系統就會做出response或者info。Response或者Info代表一個對話的結束，user目前的intent已經被滿足

Reward

Reward 會在整個dialogue結束實際算。基於對話次數越少越好的概念，將 reward 定義為：

$$R = -0.1 * t + 1 \quad if\ user\ goal\ completed$$

$$R = -0.1 * t + 0 \quad if\ user\ goal\ not\ completed$$

R 為 reward，t 為 dialogue 對話的次數。

Performance

因為NLU對或錯的機率都很高，大部分都是0.99以上，因此我們強迫他必須要聽到兩次以上(機率總和大於1.9)或者確認過才會確定某個slot或者intent，雖然這樣子做時常要耗費比較多個turn來完成一個對話，也因此造成reward較低(因為完成對話的turn越多，reward越低)，然而這樣子的機制對於資訊理解的正確率是有很大的幫助

Dialogue

- **User simulator user goal**: intent: search, artist: 林俊傑, track: 他說
  Results: Reward: 0.5, turns:5, success: True
  Final Response: 幫你播 林俊傑 的 他说:她说 金曲组曲
  Dialogue:
  1.

```
input: 播首他說吧
```

```
action:confirm
action intent: search
```

2.

```
input: 對
```

```
action:confirm
action slot: track:他說
```

3.

```
input: 是
```

```
action:question
action slot: artist:
```

4.

```
input: 想要聽林俊傑的歌
```

```
action:confirm
action slot: artist:林俊傑
```

5.

```
input: 對
```

```
action:response
action intent: search
action slot: track:他說 genre:None artist:林俊傑
```

```
[DEBUG] intent:[search], artist:[林俊傑], track:[他說],genre:[None], reward:[0.5], turns:[5], success:[True]
Dialogue System final response: 幫你播 林俊傑 的 他说:她说 金曲组曲
```

- **User simulator user goal**: intent: recommend, track: 轉圈圈
  Results: Reward: 0.5, turns:5, success: True
  Final Response: 為你推薦 A-Lin的溫柔累了 林俊傑的我很想愛他 張惠妹的我恨我愛你 - Remastered 曹格的單數
  Dialogue:

  1.

  ```
  input: 有像是轉圈圈的歌曲嗎
  ```

  ```
  action:confirm
  action intent: recommend
  ```

  2.

  ```
  input: 是
  ```

  ```
  action:confirm
  action slot: track:轉圈圈
  ```

  3.

  ```
  input: 對
  ```

```
action:question
action slot: artist:
```

4.

```
input: 不對
```

```
action:question
action slot: genre:
```

5.

```
input: 不是
```

```
action:info
action intent: recommend
action slot: track:轉圈圈 genre:None artist:None
```

```
[DEBUG] intent:[recommend], artist:[None], track:[轉圈圈],genre:[None], reward:[0.5], turns:[5], success:[True]
Dialogue System final response: 為你推薦 A-Lin的溫柔累了  林俊傑的我很想愛他  張惠妹的我恨我愛你 - Remastered  曹格的單數
```

- **User simulator user goal**: intent: info, track: 馬德里不思議
  Results: Reward: 0.4, turns:6, success: True
  Final Response: 這是蔡依林的歌曲 專輯:J世紀Jeneration 大牌新曲+精選盤2006-2009 歌曲:馬德里不思議
  Dialogue:

  1.

  ```
  input: 想知道有關於馬德里不思議的資料
  ```

  ```
  action:confirm
  action intent: info
  ```

  2.

  ```
  input: 恩
  ```

  ```
  action:confirm
  action slot: track:馬德里不思議
  ```

  3.

  ```
  input: 沒錯
  ```

```
action:question
action slot: track:
```

4.

```
input: 馬德里不思議是誰唱的啊
```

```
action:confirm
action slot: track:馬德里不思議
```

5.

```
input: 是
```

```
action:question
action slot: artist:
```

6.

```
input: 不是
```

```
action:info
action intent: info
action slot: track:馬德里不思議 genre:None artist:None
```

```
[DEBUG] intent:[info], artist:[None], track:[馬德里不思議],genre:[None], reward:[0.4], turns:[6], success:[True]
Dialogue System final response: 這是蔡依林的歌曲 專輯:J世紀Jeneration 大牌新曲+精選盤2006-2009 歌曲:馬德里不思議
```

Demonstration

使用的方法入下所述

目前中文的track、artist、曲風輸入全面支援，英文輸入尚未全部支援。

Spotify genre 有固定的seeds。這些seeds請見於data/genre_seed.txt

User Simulator CLI Demo

直接執行 userSimulator.py

`$ python2 userSimulator.py`

填入中文字串時要加個u (e.g. u'林俊傑')

Init User Goal

得到 ":請設定user intent和slots: ['intent','artist','track','genre'] " 提示。

```
troutman@6ec82c7501c8:~/ICB/MusicBot$ python userSimulator.py

:請設定user intent和slots:  ['intent','artist','track','genre']
<<<
```

設定user goal，格式為 ['intent', 'artist', 'track', 'genre']。artist, track, genre 皆為slot內的質，如果沒有的話就填入空字串('')。

Example :

Search: 可以填 artist 和 track

- 只填track '修煉愛情'
  ['search', '' , u'修煉愛情', '']
- 填artist '林俊傑' 和 track '修煉愛情'
  ['search', u'林俊傑' , u'修煉愛情', '']

Recommend: 可以填 artist , track, 或 track

- 填 track '修煉愛情'
  ['recommend', '' , u'修煉愛情' , '']
- 填 artist '林俊傑'
  ['recommend', u'林俊傑', '', '']
- 填 genre '搖滾'
  ['recommend', '', '', u'搖滾']
- 一次填多個slot： artist '林俊傑', track '修煉愛情'
  ['recommend', u'林俊傑' , u'修煉愛情', '']

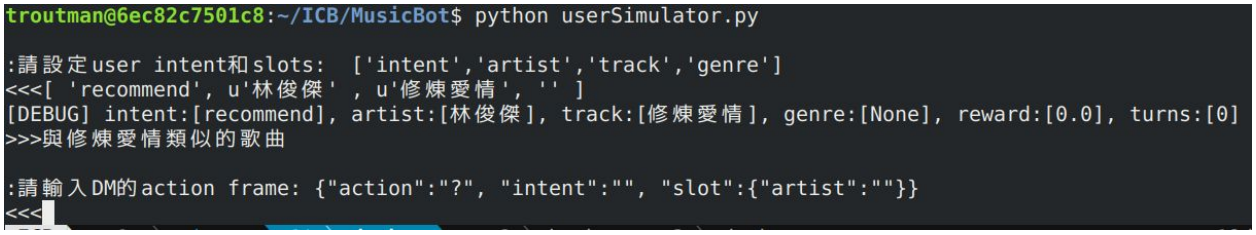
Info: 可以填track或是artist，其中一項

- 填 artist '林俊傑'
  ['info', u'林俊傑', '', '']
- 填track '修煉愛情'
  ['info', '' ,u'修煉愛情', '']

DM Action Frame

設定完後，接著userSimulator會說出一個句子，等待DM的action frame

User goal輸入 ['recommend', u'林俊傑' , u'修煉愛情', ''] 為例，得到下圖。接著，直到dialogue結束為止，都要輸入DM的 action frame跟user simulator互動。



Example:

Action: confirm

- Confirm intent
  {"action":"confirm", "intent":"recommend"}
- Confirm slot
  {"action":"confirm", "slot":{"artist": u"林俊傑"}}
- confirm intent 和 多個 slot
  {"action":"confirm", "intent":"recommend", "slot":{"artist": u"林俊傑", "track": u"修煉愛情"}}

Action: question 一次最多只會問一個slot

- 詢問 intent
  {"action":"question", "intent":""}
- 詢問 slot
  {"action":"question", "slot":{"artist":""}}

Action: response 系統對search最終的回應，結束對話。回傳所有系統讀到的訊息，包含intent和slot。

{"action":"response", "intent":"search", "slot":{"artist": u"林俊傑", "track":u"修煉愛情"}}

Action: info.系統對info和recommend最終的回應，結束對話。回傳所有系統讀到的訊息，包含intent和slot。

{"action":"info", "intent":"recommend", "slot":{"artist": u"林俊傑", "track":u"修煉愛情"}}

Dialogue Examples

1.

['recommend','maroon 5','maps','']
{"action":"confirm","intent":"recommend}
{"action":"confirm", "slot":{"artist":"maroon 5"}}
{"action":"confirm", "slot":{"track":"animals"}}
{"action":"question", "slot":{"track":""}}
{"action":"confirm", "slot":{"track":"maps"}}
{"action":"info", "intent":"recommend", "slot":{"artist":"maroon 5","track":"maps"}}

2.

['info', u'林正','','']

{"action":"question", "intent":""}
{"action":"confirm", "intent":"info"}
{"action":"confirm","slot":{"artist":u"林正"}}
{"action":"info", "intent":"info", "slot":{"artist":u"林正"}}

3.

['search','The Script','Six Degrees of Separation','']
{"action":"confirm", "intent":"recommend"}
{"action":"confirm", "intent":"search"}
{"action":"confirm","slot":{"artist":"The Script"}}
{"action":"question","slot":{"track":""}}
{"action":"response", "intent":"search", "slot":{"artist":"The Script","track":"Six Degrees of Separation"}}

4.

['recommend','Red Hot Chili Peppers','','rock']
{"action":"info", "intent":"recommend", "slot":{"artist":"Red Hot Chili Peppers","genre":"rock"}}

5.

['recommend','',u'馬德里不思議','']
{"action":"confirm","slot":{"track":u"馬德里"}}
{"action":"confirm","slot":{"track":u"馬德里不思議"}}
{"action":"info", "intent":"recommend", "slot":{"track":u"馬德里不思議"}}

Dialogue Manager Demo

有兩種demo的方式，一種是auto-test另外一種是stdin-test

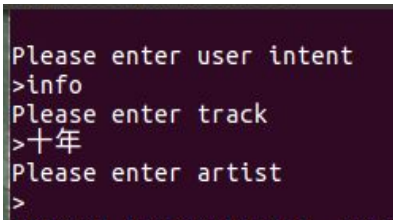
1. Input sentence test: 可以手動輸入input sentence給dialogue system，每輸入一句他回一句
    a. 執行方式:
        $ source ./data/config.sh
        $python2 Dialogue_Manager.py --stdin
    b. 執行demo:
        i. 輸入: hi初次見面
        ii. Machine: question(intent) -> 因為machine不知道user的intent
        iii. 輸入: 推薦些搖滾歌曲
        iv. Machine: confirm(intent=recommend) -> machine要再次確認user是否要推薦歌
        v. 輸入：是 我想要你推薦歌曲

vi. Machine: confirm(track='hi初次見面') -> 以為第一次打招呼的hi初次見面是首歌曲
vii. 輸入: 不是 可以推薦些搖滾風格的歌曲嗎
viii. Machine: confirm(genre=搖滾) -> 確認是否要推薦搖滾樂
ix. 輸入：是啊
x. Machine: question(artist) -> 想知道是否要推薦跟某個artis風格很像的歌曲
xi. 輸入：推薦跟林俊傑很像的歌手
xii. Machine: confirm(artist=林俊傑) -> 確認是否要跟林俊傑風格相似的歌曲
xiii. 是的
xiv. Machine: question(track) -> 有沒有要跟某首歌風格很像的歌曲
xv. 不用了 謝謝
xvi. Machine: 為你推薦 林俊傑的超越無限 Kings of Leon的Pyro Rage Against The Machine的Guerrilla Radio Toto的Hold the Line -> 一個對話的結束，推薦結果每次都會不同

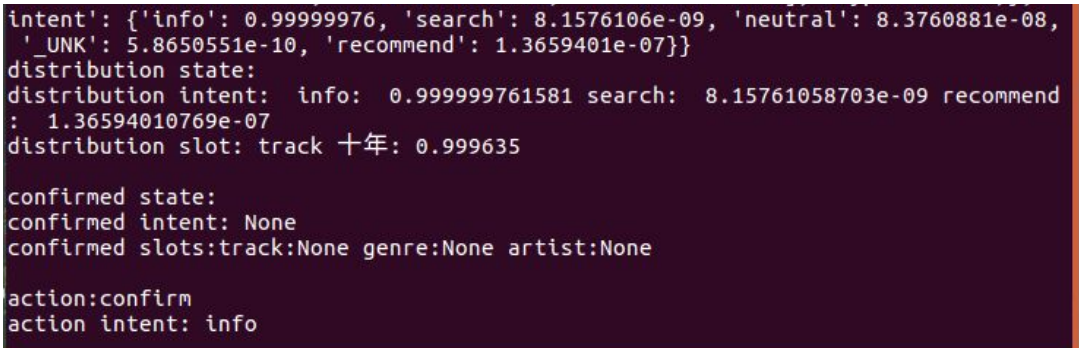
c. 執行demo2:
i. 輸入: 你好啊
ii. Machine: question(intent) -> 因為machine不知道user的intent
iii. 輸入:我想聽歌
iv. Machine: confirm(intent=search) -> machine要再次確認user是否要聽歌
v. 輸入: 對啊 我想聽歌
vi. Machine: confirm(track=歌) -> 因為machine以為"歌"是歌名
vii. 輸入： 不是 我想聽傻子
viii. Machine: confirm(track=傻子) -> 問是否想聽傻子這首歌
ix. 輸入：對啊
x. Machine: question(artist) -> 問是否要填入artist這個slot
xi. 輸入：沒有 就這樣子
xii. Machine: 幫你播 棉花糖 的 傻子 (原唱:林宥嘉) -> 一個對話結束

2. auto-test: 輸入user simulator的goal，user simulator就會自動與dialogue manager對話直到一個intent被達成對話結束，對話結束他會清空user goal，要求你再輸入新的user-goal
a. 執行方式:
$ source ./data/config.sh
$python2 Dialogue_Manager.py --auto_test
b. 執行demo:

i. 這代表user想問十年這首歌曲的資訊

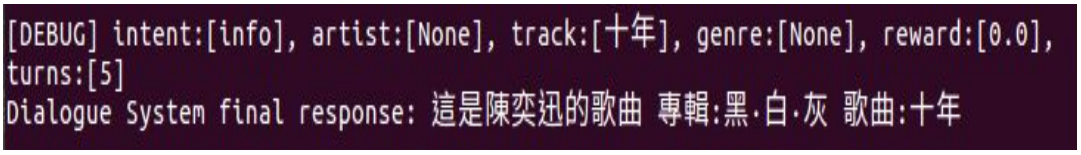


ii. User simulator會先說"你知道十年的相關資訊嗎"，DM再把NLU的結果輸入至distribution state，而因為累積的機率總和要大於1.9才會被寫入confirmed state，而機率總和只大於0.9則要做confirm的動作。因此DM會先confirm他的intent是否是info



iii. 接著再confirm歌曲是否為10年，以及question user是否還要問其他的slot，然後一個對話結束

iv. 結束user simulator會output一個分數，而DM會說十年這首歌曲的相關資，如下圖所示



Web Demo

1. 開啟server:執行 chat/chatdemo.py
2. 連上 http://yourIP:8888
3. 可以在右上角輸入intent和slot，啟動user simulator
   也可以在最下面手動輸入user的input