# Costruisci un assistente virtuale con Jarvis, BlueXP Copy and Sync e NEMO

NetApp Solutions

NetApp
April 26, 2024

# Sommario

# Panoramica

In questa sezione vengono fornite informazioni dettagliate sull'implementazione di Virtual Retail Assistant.

## Implementazione di Jarvis

Puoi iscriverti a. "Programma Jarvis Early Access" Per accedere ai container Jarvis su NVIDIA GPU Cloud (NGC). Dopo aver ricevuto le credenziali da NVIDIA, puoi implementare Jarvis seguendo questa procedura:

1. Accedi a NGC.
2. Imposta la tua organizzazione su NGC: `ea-2-jarvis`.
3. Individuare le risorse Jarvis EA v0.2: I container Jarvis sono in `Private Registry` > `Organization Containers`.
4. Selezionare Jarvis: Selezionare `Model Scripts` e fare clic su `Jarvis Quick Start`
5. Verificare che tutte le risorse funzionino correttamente.
6. Trova la documentazione per creare le tue applicazioni: I PDF sono disponibili in `Model Scripts` > `Jarvis Documentation` > `File Browser`.

## Personalizza gli stati e i flussi per i casi d'utilizzo retail

È possibile personalizzare gli stati e i flussi di Dialog Manager in base ai casi di utilizzo specifici. Nel nostro esempio di vendita al dettaglio, abbiamo i seguenti quattro file yaml per indirizzare la conversazione in base a diversi intenti.

Se il seguente elenco di nomi di file e la descrizione di ciascun file:

- `main_flow.yml`: Definisce i flussi e gli stati principali della conversazione e indirizza il flusso agli altri tre file yaml, se necessario.
- `retail_flow.yml`: Contiene stati relativi a domande al dettaglio o punti di interesse. Il sistema fornisce le informazioni del negozio più vicino o il prezzo di un dato articolo.
- `weather_flow.yml`: Contiene gli stati relativi alle domande sul meteo. Se non è possibile determinare la posizione, il sistema pone una domanda di follow-up per chiarire.
- `error_flow.yml`: Gestisce i casi in cui gli intenti dell'utente non rientrano nei tre file yaml precedenti. Dopo aver visualizzato un messaggio di errore, il sistema torna ad accettare le domande dell'utente.le sezioni seguenti contengono le definizioni dettagliate per questi file yaml.

### main_flow.yml

```
name: JarvisRetail
intent_transitions:
  jarvis_error: error
  price_check: retail_price_check
```

```
  inventory_check: retail_inventory_check
  store_location: retail_store_location
  weather.weather: weather
  weather.temperature: temperature
  weather.sunny: sunny
  weather.cloudy: cloudy
  weather.snow: snow
  weather.rainfall: rain
  weather.snow_yes_no: snowfall
  weather.rainfall_yes_no: rainfall
  weather.temperature_yes_no: tempyesno
  weather.humidity: humidity
  weather.humidity_yes_no: humidity
  navigation.startnavigationpoi: retail # Transitions should be context
and slot based. Redirecting for now.
  navigation.geteta: retail
  navigation.showdirection: retail
  navigation.showmappoi: idk_what_you_talkin_about
  nomatch.none: idk_what_you_talkin_about
states:
  init:
    type: message_text
    properties:
      text: "Hi, welcome to NARA retail and weather service. How can I
help you?"
  input_intent:
    type: input_context
    properties:
      nlp_type: jarvis
      entities:
        intent: dontcare
# This state is executed if the intent was not understood
  dont_get_the intent:
    type: message_text_random
    properties:
      responses:
        - "Sorry I didn't get that! Please come again."
        - "I beg your pardon! Say that again?"
        - "Are we talking about weather? What would you like to know?"
        - "Sorry I know only about the weather"
        - "You can ask me about the weather, the rainfall, the
temperature, I don't know much more"
      delay: 0
    transitions:
      next_state: input_intent
  idk_what_you_talkin_about:
```

```
    type: message_text_random
    properties:
      responses:
        - "Sorry I didn't get that! Please come again."
        - "I beg your pardon! Say that again?"
        - "Are we talking about retail or weather? What would you like to
know?"
        - "Sorry I know only about retail and the weather"
        - "You can ask me about retail information or the weather, the
rainfall, the temperature. I don't know much more."
      delay: 0
    transitions:
      next_state: input_intent
  error:
    type: change_context
    properties:
        update_keys:
           intent: 'error'
    transitions:
        flow: error_flow
  retail_inventory_check:
    type: change_context
    properties:
        update_keys:
           intent: 'retail_inventory_check'
    transitions:
        flow: retail_flow
  retail_price_check:
    type: change_context
    properties:
        update_keys:
           intent: 'check_item_price'
    transitions:
        flow: retail_flow
  retail_store_location:
    type: change_context
    properties:
        update_keys:
           intent: 'find_the_store'
    transitions:
        flow: retail_flow
  weather:
    type: change_context
    properties:
        update_keys:
           intent: 'weather'
```

```
    transitions:
        flow: weather_flow
  temperature:
    type: change_context
    properties:
        update_keys:
           intent: 'temperature'
    transitions:
        flow: weather_flow
  rainfall:
    type: change_context
    properties:
        update_keys:
           intent: 'rainfall'
    transitions:
        flow: weather_flow
  sunny:
    type: change_context
    properties:
        update_keys:
           intent: 'sunny'
    transitions:
        flow: weather_flow
  cloudy:
    type: change_context
    properties:
        update_keys:
           intent: 'cloudy'
    transitions:
        flow: weather_flow
  snow:
    type: change_context
    properties:
        update_keys:
           intent: 'snow'
    transitions:
        flow: weather_flow
  rain:
    type: change_context
    properties:
        update_keys:
           intent: 'rain'
    transitions:
        flow: weather_flow
  snowfall:
      type: change_context
```

```
      properties:
          update_keys:
             intent: 'snowfall'
      transitions:
          flow: weather_flow
  tempyesno:
      type: change_context
      properties:
          update_keys:
             intent: 'tempyesno'
      transitions:
          flow: weather_flow
  humidity:
      type: change_context
      properties:
          update_keys:
             intent: 'humidity'
      transitions:
          flow: weather_flow
  end_state:
    type: reset
    transitions:
      next_state: init
```

### retail_flow.yml

```
name: retail_flow
states:
  store_location:
    type: conditional_exists
    properties:
      key: '{{location}}'
    transitions:
      exists: retail_state
      notexists: ask_retail_location
  retail_state:
    type: Retail
    properties:
    transitions:
      next_state: output_retail
  output_retail:
      type: message_text
      properties:
        text: '{{retail_status}}'
      transitions:
```

```
        next_state: input_intent
  ask_retail_location:
    type: message_text
    properties:
      text: "For which location? I can find the closest store near you."
    transitions:
      next_state: input_retail_location
  input_retail_location:
    type: input_user
    properties:
      nlp_type: jarvis
      entities:
        slot: location
      require_match: true
    transitions:
      match: retail_state
      notmatch: check_retail_jarvis_error
  output_retail_acknowledge:
    type: message_text_random
    properties:
      responses:
        - 'ok in {{location}}'
        - 'the store in {{location}}'
        - 'I always wanted to shop in {{location}}'
      delay: 0
    transitions:
      next_state: retail_state
  output_retail_notlocation:
    type: message_text
    properties:
      text: "I did not understand the location. Can you please repeat?"
    transitions:
      next_state: input_intent
  check_rerail_jarvis_error:
    type: conditional_exists
    properties:
      key: '{{jarvis_error}}'
    transitions:
      exists: show_retail_jarvis_api_error
      notexists: output_retail_notlocation
  show_retail_jarvis_api_error:
    type: message_text
    properties:
      text: "I am having troubled understanding right now. Come again on
that?"
    transitions:
```

```
      next_state: input_intent
```

### weather_flow.yml

```
name: weather_flow
states:
  check_weather_location:
    type: conditional_exists
    properties:
      key: '{{location}}'
    transitions:
      exists: weather_state
      notexists: ask_weather_location
  weather_state:
    type: Weather
    properties:
    transitions:
      next_state: output_weather
  output_weather:
      type: message_text
      properties:
        text: '{{weather_status}}'
      transitions:
        next_state: input_intent
  ask_weather_location:
    type: message_text
    properties:
      text: "For which location?"
    transitions:
      next_state: input_weather_location
  input_weather_location:
    type: input_user
    properties:
      nlp_type: jarvis
      entities:
        slot: location
      require_match: true
    transitions:
      match: weather_state
      notmatch: check_jarvis_error
  output_weather_acknowledge:
    type: message_text_random
    properties:
      responses:
        - 'ok in {{location}}'
```

```
        - 'the weather in {{location}}'
        - 'I always wanted to go in {{location}}'
      delay: 0
    transitions:
      next_state: weather_state
  output_weather_notlocation:
    type: message_text
    properties:
      text: "I did not understand the location, can you please repeat?"
    transitions:
      next_state: input_intent
  check_jarvis_error:
    type: conditional_exists
    properties:
      key: '{{jarvis_error}}'
    transitions:
      exists: show_jarvis_api_error
      notexists: output_weather_notlocation
  show_jarvis_api_error:
    type: message_text
    properties:
      text: "I am having troubled understanding right now. Come again on
that, else check jarvis services?"
    transitions:
      next_state: input_intent
```

### error_flow.yml

```
name: error_flow
states:
  error_state:
    type: message_text_random
    properties:
      responses:
        - "Sorry I didn't get that!"
        - "Are we talking about retail or weather? What would you like to
know?"
        - "Sorry I know only about retail information or the weather"
        - "You can ask me about retail information or the weather, the
rainfall, the temperature. I don't know much more"
        - "Let's talk about retail or the weather!"
      delay: 0
    transitions:
      next_state: input_intent
```

# Connettersi alle API di terze parti come motore di adempimento

Abbiamo collegato le seguenti API di terze parti come motore di adempimento per rispondere alle domande:

- "API di WeatherStack": restituisce meteo, temperatura, pioggia e neve in una determinata posizione.
- "API Fusion di Yelp": restituisce le informazioni del negozio più vicino in una determinata posizione.
- "SDK di eBay Python": restituisce il prezzo di un dato articolo.

# Dimostrazione di NetApp Retail Assistant

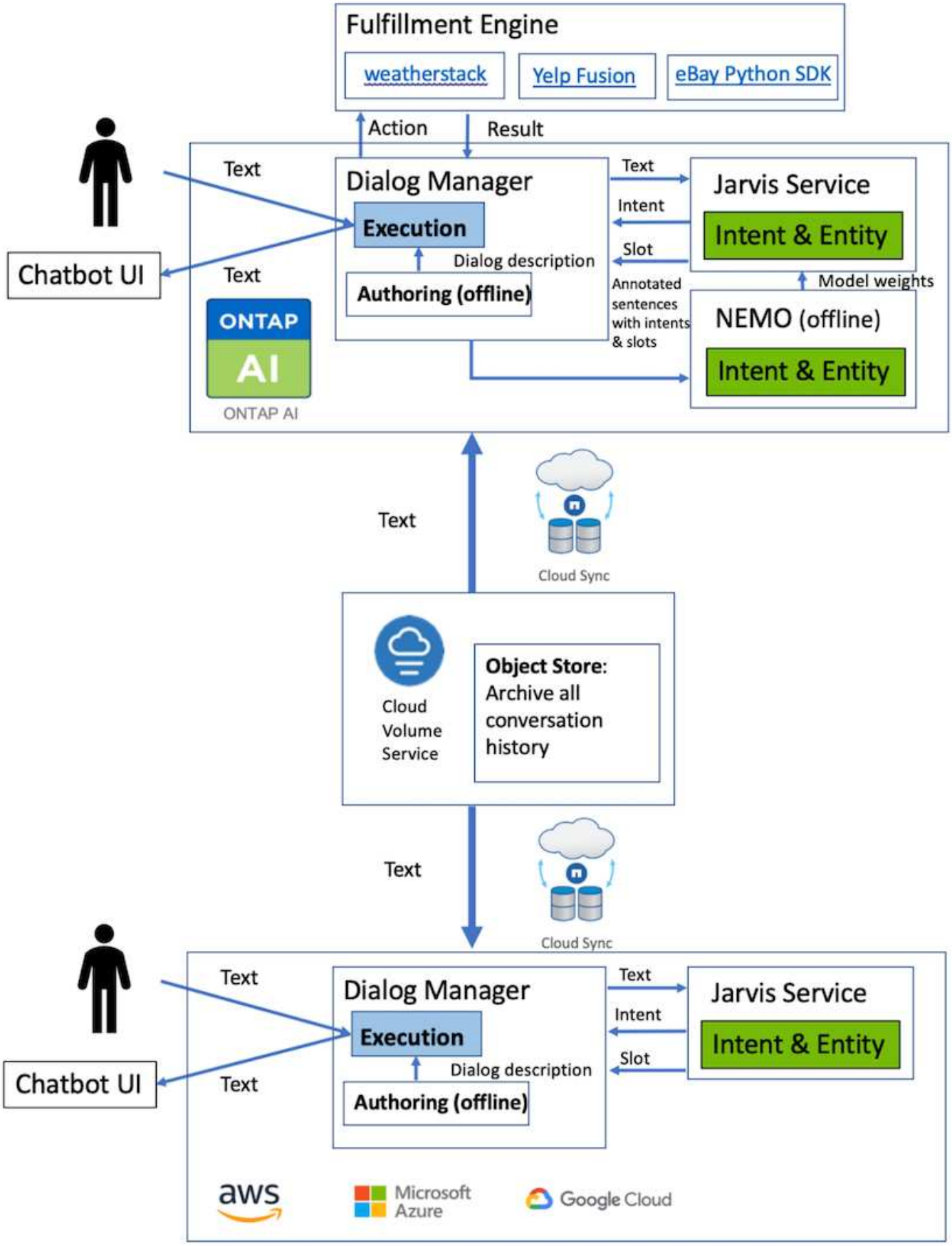
Abbiamo registrato un video dimostrativo di NetApp Retail Assistant (NARA).

### Video dimostrativo DI NARA

Video dimostrativo DI NARA

## Utilizza la copia e sincronizzazione di NetApp BlueXP per archiviare la cronologia delle conversazioni

Scaricando la cronologia delle conversazioni in un file CSV una volta al giorno, possiamo quindi sfruttare BlueXP Copy e Sync per scaricare i file di log nello storage locale. La figura seguente mostra l'architettura di avere implementato Jarvis on-premise e nei cloud pubblici, mentre utilizza BlueXP Copy e Sync per inviare la cronologia delle conversazioni per il training NEMO. I dettagli del training NEMO sono disponibili nella sezione "Espandi i modelli di intento utilizzando NEMO Training".


Fulfillment Engine
weatherstack
Yelp Fusion
eBay Python SDK
Action
Result
Text
Chatbot UI
Text
ONTAP
AI
ONTAP AI
Dialog Manager
Execution
Dialog description
Authoring (offline)
Text
Intent
Slot
Annotated sentences with intents & slots
Jarvis Service
Intent & Entity
Model weights
NEMO (offline)
Intent & Entity
Text
Cloud Sync
Cloud Volume Service
Object Store: Archive all conversation history
Text
Cloud Sync
Text
Chatbot UI
Text
Dialog Manager
Execution
Dialog description
Authoring (offline)
Text
Intent
Slot
Jarvis Service
Intent & Entity
aws
Microsoft Azure
Google Cloud

# Espandi i modelli di intento utilizzando NEMO Training

NVIDIA NEMO è un toolkit creato da NVIDIA per la creazione di applicazioni ai conversazionali. Questo toolkit include raccolte di moduli pre-formati per ASR, NLP e TTS, che consentono a ricercatori e data scientist di comporre facilmente architetture di rete neurali complesse e di concentrarsi maggiormente sulla progettazione delle proprie applicazioni.

Come illustrato nell'esempio precedente, NARA può gestire solo un tipo limitato di domanda. Questo perché il modello di NLP pre-addestrato si allena solo su questi tipi di domande. Se vogliamo consentire A NARA di gestire una gamma più ampia di domande, dobbiamo rielaborare il sistema con i nostri set di dati. In questo caso, dimostreremo come possiamo utilizzare NEMO per estendere il modello NLP in modo da soddisfare i requisiti. Iniziamo convertendo il log raccolto da NARA nel formato NEMO, quindi ci alleniamo con il set di dati per migliorare il modello NLP.

## Modello

Il nostro obiettivo è consentire A NARA di ordinare gli elementi in base alle preferenze dell'utente. Ad esempio, potremmo chiedere A NARA di suggerire il ristorante di sushi più classificato o di cercare I jeans CON il prezzo più basso. A tal fine, utilizziamo il modello di rilevamento degli intenti e di riempimento degli slot fornito in NEMO come modello di training. Questo modello consente A NARA di comprendere l'intento della ricerca delle preferenze.

## Preparazione dei dati

Per formare il modello, raccogliamo il dataset per questo tipo di domanda e lo convertiamo nel formato NEMO. Qui sono elencati i file utilizzati per la formazione del modello.

### dict.intents.csv

Questo file elenca tutti gli intenti che vogliamo che NEMO comprenda. In questo caso, abbiamo due intenti primari e un solo intento utilizzato per classificare le domande che non si inseriscono in nessuno degli intenti primari.

```
price_check
find_the_store
unknown
```

### dict.slots.csv

Questo file elenca tutti gli slot che possiamo etichettare sulle nostre domande di training.

```
B-store.type
B-store.name
B-store.status
B-store.hour.start
B-store.hour.end
B-store.hour.day
```

```
B-item.type
B-item.name
B-item.color
B-item.size
B-item.quantity
B-location
B-cost.high
B-cost.average
B-cost.low
B-time.period_of_time
B-rating.high
B-rating.average
B-rating.low
B-interrogative.location
B-interrogative.manner
B-interrogative.time
B-interrogative.personal
B-interrogative
B-verb
B-article
I-store.type
I-store.name
I-store.status
I-store.hour.start
I-store.hour.end
I-store.hour.day
I-item.type
I-item.name
I-item.color
I-item.size
I-item.quantity
I-location
I-cost.high
I-cost.average
I-cost.low
I-time.period_of_time
I-rating.high
I-rating.average
I-rating.low
I-interrogative.location
I-interrogative.manner
I-interrogative.time
I-interrogative.personal
I-interrogative
I-verb
I-article
```

```
O
```

**train.sv**

Questo è il set di dati di training principale. Ogni riga inizia con la domanda che segue l'elenco delle categorie di intento nel file dict.intent.csv. L'etichetta viene enumerata a partire da zero.

**train_slot.sv**

```
20 46 24 25 6 32 6
52 52 24 6
23 52 14 40 52 25 6 32 6
…
```

## Formare il modello

```
docker pull nvcr.io/nvidia/nemo:v0.10
```

Quindi, viene utilizzato il seguente comando per avviare il container. In questo comando, limitiamo il container a utilizzare una singola GPU (ID GPU = 1), poiché si tratta di un esercizio di formazione leggero. Inoltre, mappiamo la nostra area di lavoro locale /Workspace/nemo/ nella cartella all'interno di container /nemo.

```
NV_GPU='1' docker run --runtime=nvidia -it --shm-size=16g \
                        --network=host --ulimit memlock=-1 --ulimit
stack=67108864 \
                        -v /workspace/nemo:/nemo\
                        --rm nvcr.io/nvidia/nemo:v0.10
```

All'interno del container, se si desidera partire dal modello BERT originale pre-addestrato, è possibile utilizzare il seguente comando per avviare la procedura di training. data_dir è l'argomento per impostare il percorso dei dati di training. work_dir consente di configurare la posizione in cui si desidera memorizzare i file del punto di verifica.

```
cd examples/nlp/intent_detection_slot_tagging/
python joint_intent_slot_with_bert.py \
    --data_dir /nemo/training_data\
    --work_dir /nemo/log
```

Se abbiamo nuovi set di dati di training e vogliamo migliorare il modello precedente, possiamo utilizzare il seguente comando per continuare dal punto in cui ci siamo fermati. checkpoint_dir porta il percorso alla cartella checkpoint precedente.

```
cd examples/nlp/intent_detection_slot_tagging/
python joint_intent_slot_infer.py \
    --data_dir /nemo/training_data \
    --checkpoint_dir /nemo/log/2020-05-04_18-34-20/checkpoints/ \
    --eval_file_prefix test
```

## Deduzione del modello

Dobbiamo convalidare le performance del modello formatosi dopo un certo numero di epoche. Il seguente comando consente di eseguire il test della query uno per uno. Ad esempio, in questo comando, si desidera verificare se il modello è in grado di identificare correttamente l'intenzione della query `where can I get the best pasta`.

```
cd examples/nlp/intent_detection_slot_tagging/
python joint_intent_slot_infer_b1.py \
--checkpoint_dir /nemo/log/2020-05-29_23-50-58/checkpoints/ \
--query "where can i get the best pasta" \
--data_dir /nemo/training_data/ \
--num_epochs=50
```

Di seguito viene riportato l'output dell'inferenza. Nell'output, possiamo vedere che il nostro modello addestrato può prevedere correttamente l'intenzione find_the_store e restituire le parole chiave a cui siamo interessati. Con queste parole chiave, consentiamo A NARA di cercare ciò che gli utenti desiderano e di effettuare una ricerca più precisa.

```
[NeMo I 2020-05-30 00:06:54 actions:728] Evaluating batch 0 out of 1
[NeMo I 2020-05-30 00:06:55 inference_utils:34] Query: where can i get the
best pasta
[NeMo I 2020-05-30 00:06:55 inference_utils:36] Predicted intent:       1
find_the_store
[NeMo I 2020-05-30 00:06:55 inference_utils:50] where   B-
interrogative.location
[NeMo I 2020-05-30 00:06:55 inference_utils:50] can     O
[NeMo I 2020-05-30 00:06:55 inference_utils:50] i       O
[NeMo I 2020-05-30 00:06:55 inference_utils:50] get     B-verb
[NeMo I 2020-05-30 00:06:55 inference_utils:50] the     B-article
[NeMo I 2020-05-30 00:06:55 inference_utils:50] best    B-rating.high
[NeMo I 2020-05-30 00:06:55 inference_utils:50] pasta   B-item.type
```