

Lab 2: Watson NLP

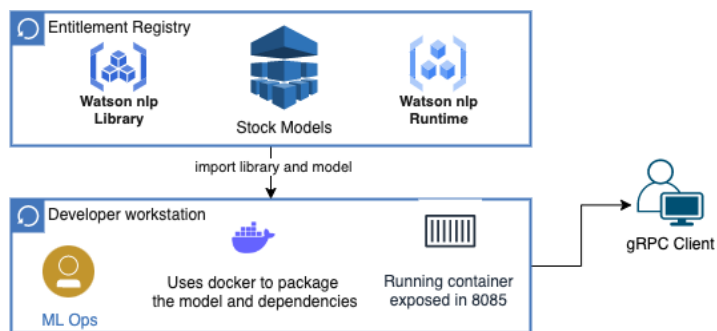
In questo laboratorio andremo a lavorare con le librerie di NLP.

Introduzione

Con IBM Watson NLP, IBM ha introdotto una libreria comune per l'elaborazione del linguaggio naturale e la comprensione dei documenti. IBM Watson NLP riunisce tutto sotto un unico ombrello, per garantire coerenza e facilità di sviluppo e distribuzione. Questa esercitazione illustra i passaggi per creare una container image per servire modelli Watson NLP pre-addestrati, ed eseguirla con Docker. L'immagine del container include sia il runtime di Watson NLP che i modelli stessi. Quando il container viene eseguito, espone endpoint REST e gRPC che i programmi client possono utilizzare per effettuare richieste di inferenza sui modelli. Queste immagini sono basate su architettura x86 e possono essere eseguiti ovunque.

Nel tutorial il container viene eseguito utilizzando Docker, ma la stessa immagine può essere eseguita anche su un cluster Kubernetes o Red Hat OpenShift, o su un servizio di container cloud come IBM Code Engine o AWS Fargate. Questo tutorial utilizza modelli preaddestrati per l'analisi del sentimento e la classificazione delle emozioni, ma possono essere utilizzati anche altri modelli preaddestrati.

Architettura



Watson NLP Runtime in a Container

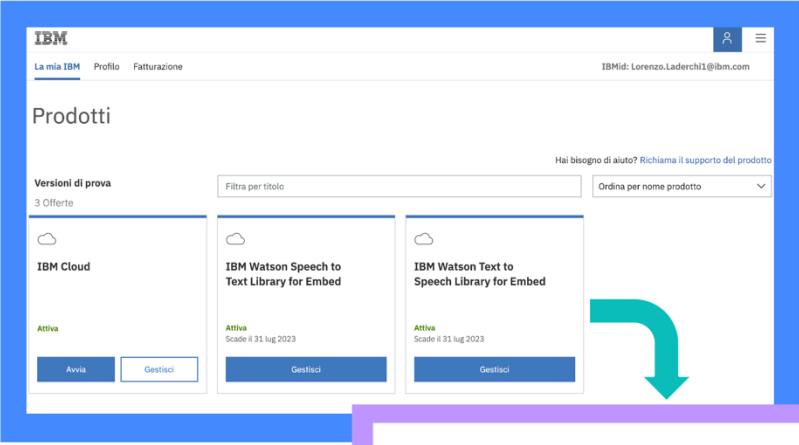
Step 0. Procurati una chiave trial

Per il laboratorio di oggi utilizzeremo una chiave trial già definita da noi come variabile d'ambiente, ma è possibile ottenerne una dalla durata di 180gg al seguente link:

<https://www.ibm.com/account/reg/us-en/subscribe?formid=urx-51754>

Per ottenerla è sufficiente eseguire il login con la propria IBMid (registrazione gratuita cliccando su “Crea IBMid”).

La chiave ha durata 6 mesi.



The screenshot shows the 'Prodotti' (Products) section of the IBM My IBM account. It lists three trial products: IBM Cloud, IBM Watson Speech to Text Library for Embed, and IBM Watson Text to Speech Library for Embed. Each product is marked as 'Attiva' (Active) and has a 'Gestisci' (Manage) button. A green arrow points from the 'Gestisci' button of the IBM Watson Text to Speech Library for Embed to a separate box below.

<https://myibm.ibm.com>

Note: 180gg free trial

Data di emissione: 1 febbraio 2023

Copy Elimina

La chiave temporanea che utilizzeremo oggi è già salvata nella variabile d'ambiente \$IBM_ENTITLEMENT_KEY; effettuiamo il login sul private registry che contiene le immagini utilizzando il seguente comando:

```
echo $IBM_ENTITLEMENT_KEY | sudo docker login -u cp --password-stdin cp.icr.io
```

Nota: se state eseguendo il laboratorio su un personal computer, potete comunque utilizzare la chiave temporanea per effettuare il login – la trovate nella root della seguente repo:

<https://github.com/LorenzoLade/Watson-Speech>

Step 1. Prepariamo il materiale del laboratorio

L'ambiente virtuale è unico ma ognuno avrà il suo progetto, pertanto digitiamo il seguente comando per creare una cartella personale:

```
mkdir tuo_cognome
```

Controlliamo la corretta creazione della cartella con il comando:

```
ls
```

Dovremmo essere in grado di vedere la cartella appena creata; posizioniamoci al suo interno con il seguente comando:

```
cd tuo_cognome
```

A questo punto è necessario clonare una repository github per avere tutto il materiale a disposizione:

```
git clone https://github.com/ibm-build-labs/Watson-NLP
```

Step 2. Creiamo l'immagine Docker

Il codice per il tutorial di oggi si trova nella sottocartella Watson-NLP/MLOps/Watson-NLP-Container, nello specifico posizioniamoci in:

```
cd Watson-NLP/MLOps/Watson-NLP-Container/Runtime
```

Controlliamo il Dockerfile presente nella cartella, che servirà da base per il build del container.

```
less Dockerfile
```

Dovremmo visualizzare qualcosa di simile:

```

$ ssh — itzuser@itz-2500002egd-k66x:/home/itzuser/laderchi/Watson-NLP-Contai...
# *****
# (C) Copyright IBM Corporation 2022.
#
# The source code for this program is not published or otherwise
# divested of its trade secrets, irrespective of what has been
# deposited with the U.S. Copyright Office.
# *****
ARG WATSON_RUNTIME_BASE="cp.icr.io/cp/ai/watson-nlp-runtime:1.0.18"
ARG SENTIMENT_MODEL="cp.icr.io/cp/ai/watson-nlp_sentiment_aggregated-cnn-workflow_lang_en_stock:1.0.6"
ARG EMOTION_MODEL="cp.icr.io/cp/ai/watson-nlp_classification_ensemble-workflow_lang_en_tone-stock:1.0.6"

FROM ${SENTIMENT_MODEL} as model1
RUN ./unpack_model.sh

FROM ${EMOTION_MODEL} as model2
RUN ./unpack_model.sh

FROM ${WATSON_RUNTIME_BASE} as release

RUN true && \
    mkdir -p /app/models

ENV LOCAL_MODELS_DIR=/app/models
COPY --from=model1 app/models /app/models
COPY --from=model2 app/models /app/models
~
~
~
(END)
```

I modelli pre-addestrati di Watson NLP sono memorizzati come container images. Quando questi container vengono eseguiti normalmente, invocano uno script `unpack_model.sh`, che estrae il file del modello e lo copia nella directory `/app/models` del file system del container. In questo Dockerfile, si esegue `unpack_model.sh` per attivare l'unpacking durante la compilazione; l'immagine finale generata utilizza l'immagine di Watson NLP Runtime come immagine di base. La variabile d'ambiente `LOCAL_MODELS_DIR` viene utilizzata per indicare al Watson NLP Runtime dove trovare i modelli che deve servire. I file dei modelli vengono copiati dalle immagini intermedie su `/app/models` nell'immagine finale.

Realizziamo la build dell'immagine appena descritta nel Dockerfile, utilizzando il seguente comando:

```
docker build . -t nlp-tuo_cognome
```

Verifichiamo il successo dell'operazione controllando l'esistenza della nostra immagine:

```
docker images
```

3. Lanciamo il servizio Watson NLP

Siamo finalmente in grado di lanciare il nostro container; utilizziamo il seguente comando:

```
docker run -e ACCEPT_LICENSE=true -d -p 8085:8085 nlp-tuo_cognome
```

I modelli sono ora accessibili tramite gRPC / API alla porta 8085

4. Testiamo il servizio

Verifichiamo che il client [Watson NLP Python](#) sia installato sul sistema.

Posizioniamoci ora sulla directory Client:

```
cd ../Client
```

Il comando `client` si aspetta come argomento una singola stringa di testo, e va ad evocare l'inferenza dai modelli pretrainati di Sentiment Analysis e Emotion Classification.

Nel caso siano stati aggiunti modelli diversi nel container (tramite Dockerfile), sarà necessario aggiornare il codice del client.

Esegui il comando client su un testo esempio:

```
python3 client.py "Watson NLP is awesome"
```

Dovremmo ottenere qualcosa di simile:

```
classes {
  class_name: "joy"
  confidence: 0.9687168002128601
}
classes {
  class_name: "anger"
  confidence: 0.03973544389009476
}
classes {
  class_name: "fear"
  confidence: 0.030667975544929504
}
classes {
  class_name: "sadness"
  confidence: 0.016257189214229584
}
classes {
  class_name: "disgust"
  confidence: 0.0033179237507283688
}
producer_id {
  name: "Voting based Ensemble"
  version: "0.0.1"
}

score: 0.9761080145835876
label: SENT_POSITIVE
sentiment_mentions {
  span {
    end: 21
    text: "Watson NLP is awesome"
  }
  score: 0.9761080145835876
  label: SENT_POSITIVE
}
producer_id {
  name: "Document CNN Sentiment"
  version: "0.0.1"
}
```

Versione video del lab: <https://youtu.be/UdUt5v4672k>

Documentazione API: <https://developer.ibm.com/apis/catalog/embeddableai--watson-natural-language-processing-apis/Introduction>