Documentazione Progetto: Technologies for Advanced Programming

Alberto Bocchieri

18 febbraio 2025

Indice

1 Introduzione

Questo progetto integra diverse tecnologie per realizzare un sistema completo per la ricerca, il download e il monitoraggio di torrent, con un arricchimento dei dati e un'interfaccia interattiva tramite un Bot Telegram. Le principali funzionalità includono: *scraping*, *streaming*, *indicizzazione*, *interazione*, *arricchimento* e *containerizzazione*.

2 Architettura del Sistema

Il sistema è composto da diversi componenti, ognuno dei quali è containerizzato per garantire portabilità e scalabilità. Di seguito una panoramica:

1. **Scraping:** Un'applicazione Python che esegue lo scraping dal sito KickassTorrent per estrarre informazioni sui torrent.
2. **Streaming con Kafka:** I dati raccolti vengono inviati a un topic Kafka per lo streaming in tempo reale.
3. **Indicizzazione con Elasticsearch:** I torrent vengono indicizzati in Elasticsearch per rendere possibile una rapida ricerca e analisi.
4. **Bot Telegram:** Un Bot sviluppato in Python permette all'utente di interagire con il sistema per cercare torrent, avviare download e monitorare il progresso.
5. **qBittorrent:** Il sistema interagisce con qBittorrent tramite la sua API per avviare e monitorare i download.
6. **Apache Spark MLlib:** Implementazione dell'algoritmo ALS per il collaborative filtering per costruire il sistema di raccomandazioni.

3 Componenti del Progetto

3.1 Scraping e gestione dei Torrent

Lo script di scraping utilizza le librerie `requests` e `BeautifulSoup` per:

- Accedere al sito di KickassTorrent.
- Estrarre i titoli e i magnet link dei torrent.
- Filtrare i torrent in base a criteri di alta qualità (ad es. presenza di parole chiave come `4K`, `2160p`, `HDR`, `BDRemux`, `x265`).

I dati raccolti vengono inviati come messaggi a un topic Kafka. Un consumer elabora questi dati e li indicizza in Elasticsearch.

3.2 Elasticsearch e Kafka

Elasticsearch è un server di ricerca con supporto ad architetture distribuite. Tutte le funzionalità sono nativamente esposte tramite interfaccia RESTful, mentre le informazioni sono gestite come documenti JSON. In questo progetto vengono creati due indici: uno per l'indicizzazione delle informazioni sui torrent e l'altro per archiviare i feedback utente sui film.

Apache Kafka è una piattaforma open source di stream processing, ovvero per tutte le applicazioni di elaborazioni di stream di dati in tempo reale. In questo caso viene impiegato per la comunicazione tra gli script python e Elasticsearch.

3.3 Bot Telegram e qBittorrent API

Il bot Telegram, sviluppato con la libreria `python-telegram-bot`, offre le seguenti funzionalità:

- **Ricerca:** `/search` per cercare torrent e ricevere risultati arricchiti.
- **Download:** Possibilità di avviare il download tramite pulsante inline che comunica con qBittorrent.
- **Monitoraggio:** `/monitor` per visualizzare in tempo reale lo stato dei download. Inoltre saranno disponibili dei pulsanti inline per fermare/riprendere singolarmente i torrent disponibili.
- **Consigli:** `/consigliami` per ricevere dei consigli sulla base dei feedback dell'utente.

Il bot interagisce con qBittorrent utilizzando la libreria `qbittorrentapi` e avvia i download tramite la sua API.

3.4 Docker e Orchestrazione

È stato utilizzato Docker per containerizzare diverse componenti del progetto, quali: Elasticsearch e Kafka. Entrambi girano in un container separato. Quindi viene utilizzato *docker-compose* per orchestrare e gestire facilmente tutti i servizi, garantendo isolamento, scalabilità e portabilità del progetto.

4 Flusso di Lavoro del Sistema

Il flusso di lavoro tipico del sistema è il seguente:

1. **Interazione dell'Utente:** L'utente invia comandi al bot Telegram (ad es. `/search` o `/consigliami`).
2. **Ricerca e Scraping:** Il modulo di scraping estrae i torrent da Kickasstorrent e li invia a Kafka.
3. **Indicizzazione:** I dati vengono processati e indicizzati in Elasticsearch.
4. **Download:** Tramite pulsanti inline, l'utente può avviare il download di un torrent su qBittorrent.
5. **Feedback:** Sempre attraverso due pulsanti inline posti al di sotto del precedente, l'utente può mandare un feedback sul torrent.
6. **Monitoraggio:** Il bot aggiorna periodicamente lo stato dei download e lo mostra all'utente.

5 Interazione con l'Utente (Bot Telegram)

Il bot Telegram offre un'interfaccia interattiva con i seguenti comandi:

- `/start`: Avvia il bot e mostra il menu dei comandi.
- `/search`: Permette di cercare torrent specificando il nome del film.
- `/monitor`: Avvia il monitoraggio in background dei download, aggiornando periodicamente lo stato.
- `/stop_monitor`: Ferma il monitoraggio in background.

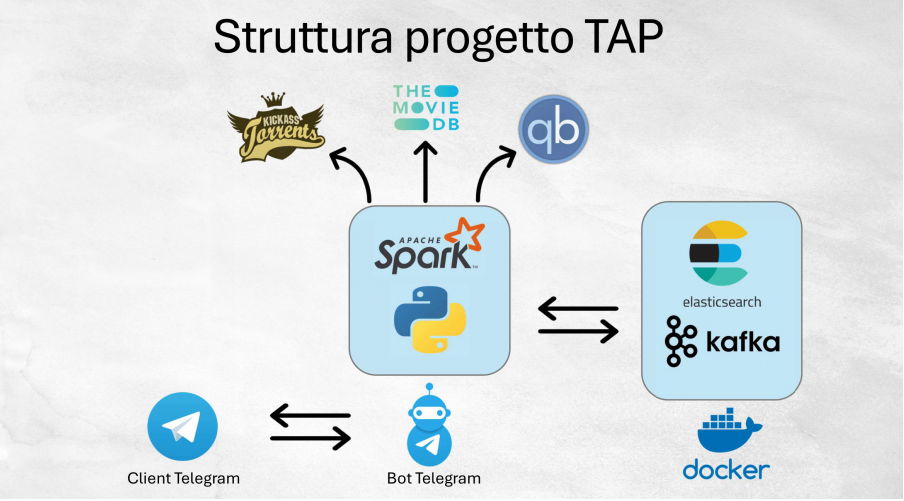

Figura 1: Struttura completa

- `/consigliami`: Suggerisce un film in base ai gusti dell'utente.

Il bot utilizza anche pulsanti inline per:

- Avviare il download su qBittorrent.
- Fermare singolarmente i torrent attivi.

6 Arricchimento del Dato

Per rendere il dato più utile e completo, il sistema integra:

- **API TMDb:** Per ottenere locandine, sinossi, cast, regista, anno di uscita e valutazioni dei film.
- **Classificazione automatica:** Algoritmi che analizzano il titolo per determinare la qualità (es. presenza di termini come `4K`, `HDR`, `BDRemux`).
- **Normalizzazione:** Funzioni che rimuovono o standardizzano parti del titolo (ad esempio, rimuovendo il gruppo di rilascio).
- **Feedback dell'Utente:** Possibilità di fornire feedback tramite il bot per affinare le raccomandazioni future.
- **Raccomandazioni:** Attraverso l'uso di Apache Spark e, in particolare dell'algoritmo ALS, è implementato il collaborative filtering, attraverso cui è possibile consigliare dei film all'utente sulla base dei feedback che quest'ultimo fornisce.

6.1 Implementazione del collaborative filtering

Spark MLlib fornisce l'algoritmo ALS (Alternating Least Squares), utilizzato per costruire sistemi di raccomandazione. ALS scompone la matrice che rappresenta le valutazioni degli utenti nei confronti di vari oggetti (come film in questo caso) in due matrici: una che cattura le preferenze degli utenti e l'altra che rappresenta le caratteristiche degli oggetti in questione. Il processo si svolge iterativamente: si fissa una delle due matrici e si risolvono i problemi di regressione lineare per calcolare l'altra, e poi si alternano queste

ottimizzazioni fino a quando il modello converge o si raggiunge un numero massimo di iterazioni.

Nel caso in cui non si conoscano direttamente le preferenze degli utenti per alcuni oggetti, queste si potranno dedurre osservando il comportamento collettivo. Quindi, se molti utenti che hanno gradito determinati film tendono a preferire anche altri film, il modello sarà in grado di identificare queste correlazioni e fare previsioni sui gusti di utenti che non hanno ancora valutato tutti gli oggetti. Le previsioni, però, nel caso in cui l'utente avesse poche interazioni, potrebbero essere meno accurate (problema "cold start")

7 Possibili Estensioni Future

- **Monitoraggio Avanzato:** Integrazione di Grafana per visualizzazioni più sofisticate e alerting in tempo reale.
- **Analisi dei Contenuti:** Utilizzare strumenti NLP per estrarre ulteriori metadati dai torrent e arricchire ulteriormente il dato.
- **Subscribe:** Integrazione di un sistema di subscribe per ricevere notifiche in tempo reale dei nuovi torrent caricati

8 Conclusioni

Con l'integrazione di tecnologie come Kafka, Elasticsearch, Spark, Docker e un Bot Telegram, il sistema non solo raccoglie e gestisce i torrent, ma li arricchisce con informazioni dettagliate, permette il monitoraggio in tempo reale e offre un'interfaccia utente interattiva e personalizzata. Le possibilità di estensione sono numerose e il sistema è progettato per essere scalabile e modulare, facilitando l'integrazione di nuove funzionalità e miglioramenti futuri.