Effetto del Covid-19 su Youtube? (titolo provvisorio) [fede]
Un'analisi dei video in tendenza

Gabriele Celeri, Federico Luzzi, Marco Peracchi, Christian Uccheddu

---

Introduzione e obiettivi (DA RISCRIVERE)
In questo lavoro l'obiettivo è quello di capire come un video di Youtube riesca ad avere successo. Per raggiungere questo scopo abbiamo deciso di analizzare i video di Youtube che si trovano in trending nelle varie ore delle giornata, studiandone caratteristiche, categorie di appartenenza, insieme a visualizzazioni, likes, dislikes, commenti e altro. L'obiettivo è anche quello di comprendere come cambiano le caratteristiche nei vari paesi, e cosa serve per avere successo in ognuno di questi.

IO SCRIVEREI TUTTO AL PRESENTE CON MAGARI COME SOGGETTO "NOI" (ALTRIMENTI SI PUÒ SCRIVERE IMPERSONALE) [fede]

Scelta degli strumenti (DA RISCRIVERE)

La prima domanda che ci siamo posti riguardo al progetto è stata: "Quali sono le "V" su cui focalizzare la nostra attenzione?". Dopo qualche indagine preliminare abbiamo deciso che che ci saremmo concentrati su "Volume" e su "Velocity", perché i nostri dati venivano raccolti in tempo reale dalle API di Youtube e la quantità di dati aumenta costantemente con l'aumentare del tempo di presa dati.

Per gestire il flusso di dati ci siamo affidati al software Kafka, permettendoci così di poter disaccoppiare la fase di lettura dei dati dalla fase di raccolta. Uno script python (*scraper_producer.py*) si occupava del producer, continuando ad effettuare richieste alle API di youtube, mentre lo script consumer (*scraper_consumer.py*) si occupava di leggere i file memorizzati nel topic di kafka, e successivamente di immagazzinarli in un database MongoDB in formato JSON. La scelta di MongoDB è stata dettata dalla sorgente dei dati, che venivano forniti in formato documentale.

Raccolta dati

La raccolta dati è basata su due fonti principali: YouTube e [fonte/i covid-19].

Youtube API

I video in tendenza su youtube variano ogni 15 minuti circa (fonte: `https://support.google.com/youtube/answer/7239739?hl=it`) anche se non per forza e solitamente vi è solo una manciata di video che entra/esce dalle tendenze. Per la raccolta di questi dati ci siamo basati sulle funzioni API messe a disposizione da youtube per poter raccogliere i video in tendenza in un dato momento. Le chiavi di richiesta gratuite fornite da Google developer però ci permettono di effettuare una quantità limitata di richieste.
Abbiamo effettuato sostanzialmente due differenti **sessioni** di scraping, con metodi leggermente differenti, dovuti allo scopo che ci eravamo prefissati:

1. dal 23 dicembre 2019 al 5 gennaio 2020 (raccolta ogni 30 minuti)
2. dal 18 marzo 2020 al 6 maggio 2020 (raccolta ogni 6 ore)

Abbiamo deciso di raccogliere dati delle tendenze dei seguenti paesi:

Italia, USA, Regno Unito, India, Germania, Canada, Francia, Corea del sud, Russia, Giappone, Brasile, Messico

Prima sessione

Inizialmente pensavamo di utilizzare i dati per analizzare come, in generale, un video riesce ad arrivare nelle tendenze di youtube e in base a questo fatto analizzare il suo comportamento per quanto riguarda visualizzazioni, likes, dislikes, ecc. La raccolta in questa sessione viene effettuata con lo script *scraper_timed.py* che sostanzialmente esegue questi passaggi:

Algorithm 1: Scraping youtube → csv

Data: country - insieme dei paesi prescelti; videos - insieme di video scaricati da un determinato paese

```
for every 30 minutes do
    foreach country do
        videos = APIrequest(max(50 video), country)
        while video in tendenza non finiti do
            videos += APIrequest(max(50 video), country)
        videos_fix = arrangeData(videos)
        saveToCsv(videos_fix)
        videos = ∅
```

Nota: per ogni API request al massimo possiamo scaricare 50 video e ogni paese a un numero diverso di video in tendenza (solitamente 150-200).

In questa sessione abbiamo pensato di salvare i dati in formato *csv* modo tale che siano facilmente manipolabili. Segue lo schema logico con cui li abbiamo immagazzinati:

Attributo	Descrizione
timestamp	data, ora e minuto della nostra rilevazione
video_id	identificativo unico del video
title	nome del video per esteso
publishedAt	data di pubblicazione
channelId	identificativo unico del canale che ha pubblicato il video
channelTitle	nome del canale per esteso
categoryId	identificativo unico della categoria
trending_date	data in cui il video è in tendenza
tags	stringa dei tag usati separati dal carattere "|"
view_count	numero di visualizzazioni
likes	numero di like (mi piace)
dislikes	numero di dislike (non mi piace)
comment_count	numero di commenti sotto il video
thumbnail_link	url all'immagine di copertina del video
comments_disabled	booleano che dichiara se i commenti sono disabilitati
ratings_disabled	booleano che dichiara se i like/dislike sono disabilitati
description	descrizione del video

Tabella 1: schema degli attributi dei dati csv

I dati così raccolti sono salvati nella cartella [INSERIRE NOME CARTELLA]

Seconda sessione

Per la seconda sessione di raccolta, in cui abbiamo deciso di concentrarci sugli effetti che l'epidemia del nuovo virus Covid-19, abbiamo adottato un approccio diverso. Innanzitutto abbiamo deciso di immagazzinare i dati in un database mongoDB nativamente. Pertanto oltre a costruire una pipeline di raccolta che salva i dati direttamente su mongoDB abbiamo deciso parallelamente di immagazzinarli in formato *json*.

Seconda considerazione è stata data la bassa variabilità dei video raccogliendo i dati ogni 30 minuti, vista nella precedente sessione, abbiamo deciso di effettuare lo scraping ogni 6 ore, in modo da avere 4 rilevazioni durante una giornata.

A scopo didattico abbiamo inoltre deciso di implementare una piccola pipeline Kafka in cui dividiamo la fase di raccolta dati (*scraper_producer.py*) e la fase di immagazzinamento (*scraper_consumer.py*).

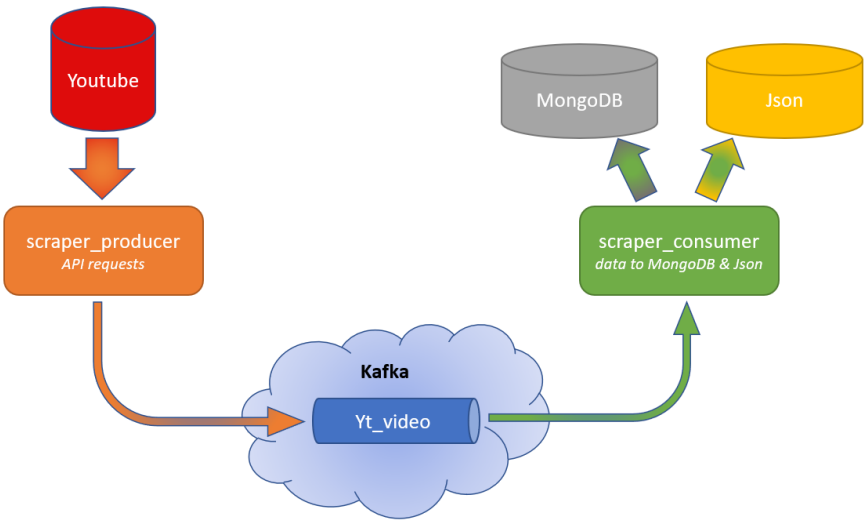

Figura 1: pipeline di raccolta dati - seconda sessione

Di seguito il dettaglio della computazione dei 2 script:

Algorithm 2: scraper producer

Data: country - insieme dei paesi prescelti; videos - insieme di video scaricati da un determinato paese; KafkaProducer(data, channel) - sends data to channel kafka

```
for every 6 hours do
    foreach country do
        videos = APIrequest(max(50 video), country)
        KafkaProducer(videos, yt_video)
        while video in tendenza non finiti do
            videos = APIrequest(max(50 video), country)
            KafkaProducer(videos, yt_video)
```

Algorithm 3: scraper consumer

Data: KafkaConsumer(channel) - receives data from channel kafka

```
while loop do
    if KafkaConsumer(yt_video) ≠ ∅ then
        videos = KafkaConsumer(yt_video)
        videos_fix = arrangeData(videos)
        saveToJson(videos_fix)
        saveToMongoDB(videos_fix)
```

I dati così raccolti sono stati salvati in formato json nella cartella [INSERIRE NOME CARTELLA].

Nota: abbiamo scelto di salvare i dati in formato Json in modo che fosse facile caricarli in un nuovo server mongoDB e renderli più trasportabili. Per il caricamento vedi lo script: *json_to_mongo.py*

Dati Covid-19

Qua dobbiamo scrivere come/dove abbiamo preso i dati covid

Qualità dati

Avendo una grande mole di dati ci siamo dovuti anche occupare di verificare la pulizia o meno del nostro dataset. Per prima cosa abbiamo notato che non c'era presenza di Missing Values, questo è stato di per sé una grande fortuna poiché di solito il Missing Replacement è una operazione molto lunga. Ci sono state però due problematiche principali che abbiamo dovuto affrontare:

- Ridondanza: Questo problema è nato dal fatto che i trending di Youtube sono stati presi ogni mezz'ora, con il risultato di avere molti dati simili riguardanti le stesse fasce della giornata.
- Buchi: Questo problema è nato dal fatto che Google non permette di avanzare troppe richieste per la presa dati nella stessa giornata, per arginare questo problema abbiamo utilizzato 3 chiavi diverse con cui mandare le richieste a Google in modo tale da coprire la maggior parte della giornata. Ci sono stati però dei punti in cui la presa dati non è riuscita. Il metodo che abbiamo utilizzato per riempire quei buchi è stato quello di duplicare i dati della presa dati precedente. Questo metodo è stato possibile poiché la presa dati è stat effettuata in intervalli di tempo in cui i dati non vengono modificati significativamente.

Scalabilità dell'algoritmo

Visto che una delle V che abbiamo deciso di usare concernenti i Big Data è stata quella relativa al volume dobbiamo occuparci di vedere come si comporta la nostra elaborazione dati con volumi di dati sempre crescenti. Abbiamo deciso di implementare lo **Sharding** su MongoDB. In particolare, abbiamo deciso di costruire 3 shard replica group formati da 3 repliche dello shard in modo da garantire la ridondanza nel caso di guasti e la frammentazione per rendere più efficienti le query. La chiave di sharding scelta è quella del campo "**country_name**" perché le query per rispondere alle nostre domande vengono fatte sul singolo paese. Questo approccio è stato pensato anche nell'ottica di dividere la grande mole di dati in server disposti in ciascun paese.

Segue lo schema logico applicato:

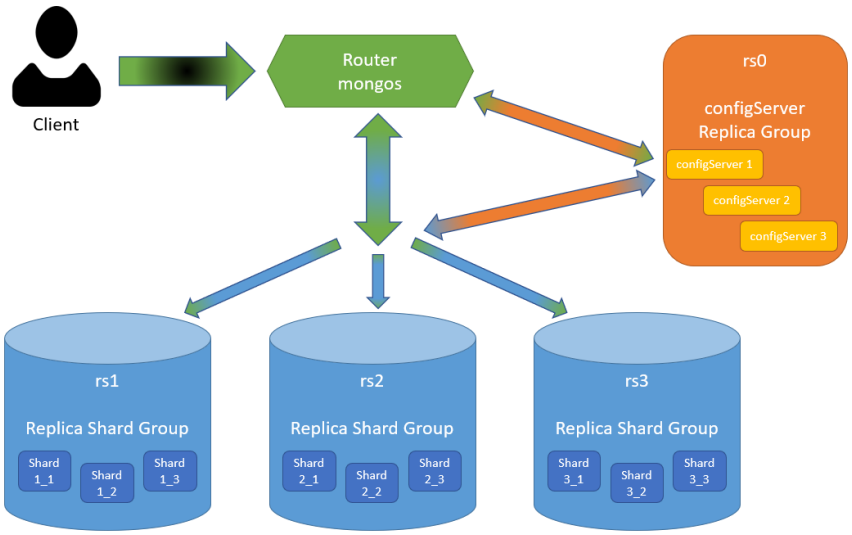

Figura 2: pipeline sharding

QUERY ANALISI:

Visualizzazione

Per la scelta delle visualizzazione abbiamo dovuto pensare per prima cosa a delle domande di ricerca a cui rispondere. Le domande che ci siamo posti sono le seguenti:

- Variazione della tipologia di video in tendenza da periodo pre covid-19 a periodo covid-19
- È vero che la fruizione di video su Youtube riguardanti i contagi di Covid-19 segue l'andamento dei dati sull'epidemia?

Scelta features

Per rispondere in modo coerente alle nostre domande di ricerca abbiamo deciso di concentrarsi sulle seguenti features del nostro dataset.

- Numero di visualizzazioni.
- Numero di like.
- Numero di dislike.
- Numero di commenti.
- Numero di iscritti ad un canale.

A partire da queste feature abbiamo inoltre definito un indice fittizio che abbiamo deciso di denominare *Indice di partecipazione.* Questo indice ci è servito per capire quale fosse il video con il maggior numero di partecipazioni rispetto al numero di visualizzazioni ottenute in modo da poter capire quanto le tendenze siano influenzate da questo indice. In particolare:

$$I = \frac{like + dislike + comment}{views}$$

Scelta della visualizzazione

La visualizzazione che meglio si presta a dati con natura fortemente temporale è una visualizzazione a righe. Per arricchire la visualizzazione di contenuti abbiamo inserito ulteriori caratteristiche a questa visualizzazione. In particolare la visualizzazione è composta nel seguente modo: In questo caso

Valutazione della qualità

La valutazione della qualità della nostra infografica si è articolata in tre macro passaggi:

User Test Durante questa fase ci siamo occupati di sottoporre la nostra infografica a 3 persone lasciando completa libertà di esplorazione. Le varie esplorazioni sono state registrate in modo da far sì che potessero emergere le diverse problematiche di cui non ci siamo accorti in fase di realizzazione delle infografiche. Esponiamo le problematiche emerse durante questa fase di valutazione e le correzioni applicate:

- Problema 1: soluzione 1
- Problema 2: soluzione 1
- Problema 3: soluzione 1

Risultati dei task Durante questa fase ci siamo occupati di sottoporre tre diverse richieste a 24 utenti che dovevano essere soddisfatte esplorando interattivamente la nostra infografica. In particolare i task da risolvere sono stati i seguenti:

- Qual'è il video con l'indice di partecipazione più alto in Italia?
- Qual'è il video con l'indice di partecipazione più alto tra tutti i paesi?
- Che categoria di video dovresti utilizzare se volessi creare contenuti che piacciono in USA?

Ci siamo occupati di registrare i tempi in cui gli utenti riuscivano a completare questu obiettivi e abbiamo visualizzato questi record nei seguenti violin plot: Questa visualizzazione è utile per capire se le nostre infografiche sono troppo dispersive o riescono a centrare gli obietivi facilmente.

Questionari Per quanto riguarda quest'ultima fase ci siamo occupati di rivolgere un questionario della valutazione della qualità a 24 persone. In particolare il questionario è stato articolato nella seguente maniera:

- Come valuti la chiarezza dell' infografica?
- Come valuti l'utilità dell'infografica?
- Quanto valuti la bellezza dell'infografica?
- Come valuti l'intuitività dell'infografica?

- Quanto è stata informativa l'infografica?
- Come valuti complessivamente l'infografica?

Le risposte sono state registrate grazie al tool: Questionari di Google. Una volta registrate le risposte ci siamo occupati di vedere se la valutazione complessiva dell'infografica fosse coerente con una ricostruzione complessiva data dalla regressione con i coefficienti di Cabitza-Locoro. E' stato comodo usare un box plot per la registrazione di queste risposte poiché la media è un indicatore di tendenza centrale e non fornisce alcuna informazione sulla distribuzione di questi dati. Per quanto riguarda invece della coerenza della valutazione complessiva rispetto alla ricostruzione data dai coefficienti abbiamo avuto la seguente distribuzione: