UNIVERSITÀ DELLA CALABRIA

DIPARTIMENTO DI INGEGNERIA INFORMATICA, MODELLISTICA, ELETTRONICA E SISTEMISTICA

Corso di Laurea Magistrale in

Ingegneria Informatica - Indirizzo Artificial Intelligence and Machine Learning

Relazione progetto Social Media and Network Analisys

Candidati:

Matteo Capalbo, Nicola Gabriele

ANNO ACCADEMICO 2023/2024

Indice

Link alle risorse: https://github.com/NicolaGabriele/MastodonContentCompliance

1 Introduzione al task

1.1 Mastodon

Mastodon è il più grande social network decentralizzato. Fa parte del fediverso, una comunità internazionale composta da oltre 13 milioni di iscritti distribuiti su circa 22000 server indipendenti ma federati fra di loro e quindi gli utenti di diverse istanze possono comunicare fra di loro. L'obiettivo è rimettere il social nelle mani degli utenti. A differenza dei social tradizionali non ha un proprietario, ogni server è gestito da privati, associazioni o collettivi, è open source, non raccoglie i dati degli iscritti, non ha pubblicità o algoritmi segreti che decidono quali contenuti proporre agli utenti. Dal punto di vista tecnico la caratteristica interessante di questo social è che le sue API rest sono aperte e disponibili e quindi si presta molto bene a studi come quelli di nostro interesse. Come sù detto, mastodon è composto da un insieme di istanze, gli amministratori di ciascuna di queste istanze hanno la possibilità di fissare le regole che gli iscritti ad una certa istanza devono rispettare. Dunque, in ogni istanza abbiamo degli utenti che fungono da moderatori che hanno il compito di garantire l'osservanza di tali regole da parte di tutti gli utenti.

1.2 Posts Compliance

Se consideriamo che Mastodon è un social in forte crescita sia per numero di utenti che per diretta conseguenza per numero di post, il compito di garantire la conformità dei post alle regole stabilite in ciscuna istanza a volte può diventare arduo. Con questo elaborato ci poniamo l'obiettivo di effettuare una analisi dei post delle istanze più rilevanti al fine di quantificare quanto questi siano allineati alle regole dell'istanza in cui sono stati pubblicati. Il seguente lavoro si propone come analisi preliminale per un successivo sviluppo di metodo di rilevazione automatica delle violazioni e/o di proposte di modifica

ai post. Scendendo più nel dettaglio vorremmo servirci di tecnologie come i *Large Language Models* per analizzare i post delle istanze e ricavare da questi uno **score di allineamento** per consentirci il calcolo delle statistiche sulla conformità dei post e ricavare da esse delle conclusioni. Nel prossimo capitolo segue una descrizione dettagliata della metodologia e delle scelte progettuali.

2 Metodologia

Per prima cosa abbiamo scaricato la lista delle istanze attive attraverso le API messe a disposizione dal servizio *instances.social* che permette di ottenere la lista delle istanze attive attraverso il seguente end point:

```
GET https://instances.social/api/1.0/instances/list
```

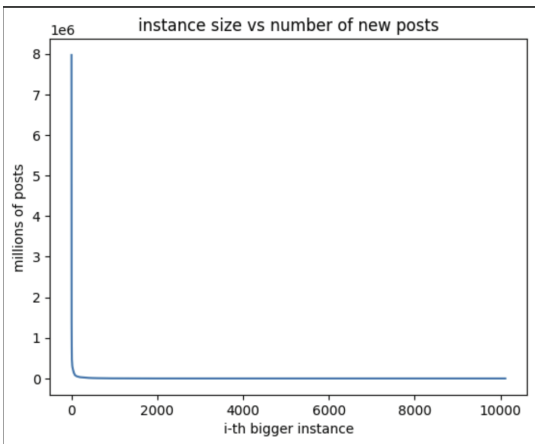
Come già detto nell'introduzione, Mastodon è un social in forte crescita. Mentre scriviamo, la sua più grande istanza è *"mastodon.social"* e conta circa **8 milioni di nuovi post solo negli ultimi 3 mesi**. La prima problematica con cui scontrarsi è quindi quella di trovare un modo per evitare di analizzare tutti i post garantendo però validità statistica del lavoro svolto.

2.1 Distribuzione dei post tra le istanze

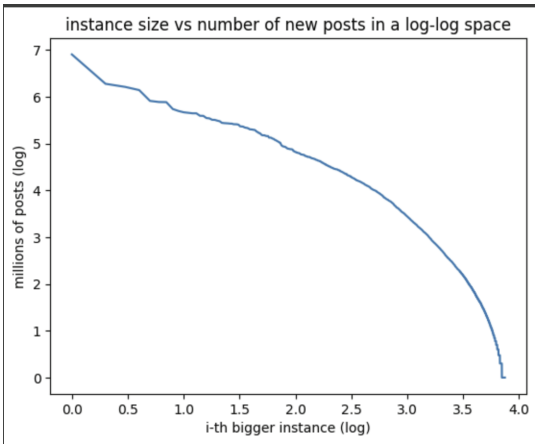
La prima cosa che abbiamo fatto è analizzare la distribuzione dei post tra le istanze per capire se questa seguisse un particolare tipo di legge da poter sfruttare per giustificare l'analisi di un numero ridotto di istanze. Per fare ciò abbiamo utilizzato l'informazione sul *numero di nuovi post negli ultimi 3 mesi* di ciascuna istanza recuperabile tramite API al seguente endpoint:

```
GET https://{instance_name}/api/v1/instance/activity
```

Una volta ottenuto il conteggio dei post per ciascuna delle istanze ci è stato possibile graficare questi dati. Diamo dunque uno sguardo ai grafici prodotti:



Per completezza plottiamo i dati anche in uno spazio *log-log*:



E' chiaro che questa distribuzione sia caratterizzata da un forte decadimento esponenziale. Questo ci permette di analizzare un numero ridotto di istanze senza perdere in generalità nelle conclusioni a cui giungiuamo. Decidiamo dunque di analizzare il **5%** delle istanze più rilevanti attive al momento in cui abbiamo recuperati i dati.

2.2 Crawling dei post

Una volta stabilito quali istanze analizzare possiamo procedere al crawling dei post delle varie istanze. Per avere maggiore agevolezza nel manipolare i dati abbiamo deciso di salvare i post di ciascuna istanza all'interno di un file dal formato json avente lo stesso nome dell'istanza a cui afferiscono i dati. Ciascun file è un oggetto json avente la seguente forma:

```
{
    "name": <nome dell'istanza>
    "rules": <lista delle regole dell'istanza>
    "records": <lista dei post sotto forma di oggetto json>
}
```

Ciascun oggetto della lista *records* è composto dei seguenti campi:

```
{
    "id": <id del post>
    "user_id": <id dell'utente>
    "user_posts_count": <numero di post pubblicati dall'utente>
    "text": <testo del post>
    "tags": <lista dei tag associati al post>
    "language": <lingua in cui è scritto il post>
    "favourites": <numero di like del post>
}
```

Per recuperare queste informazioni abbiamo utilizzato due diversi end point:

```
GET https://{instance_name}/api/v1/instance/rules
```

Per recuperare la lista delle regole di ciascuna istanza.

```
GET https://{instance_name}/api/v1/timelines/public
```

Per le informazioni relative ai post.
Inoltre, dato l'enorme numero di post disponibili, per limitare il tempo richiesto dal crawling abbiamo fissato il seguente vincolo sulla quantità dei post di ciascuna istanza: vengono recuperati i post afferenti ad un orizzonte temporale di tre mesi e ove ce ne fossero un gran numero questi vengono limitati ad un tetto massimo di 50 mila. L'unica eccezzione è l'istanza mastodon.social della quale abbiamo recuperato 400 mila post essendo la più significativa. Per il crawling dei post sono state impiegate 2 macchine, per un totale di 32 processori logici e 40 thread che hanno consentito una parallelizzazione del lavoro per un totale di 70 ore.

2.3 Analisi dei post

2.3.1 Utilizzo di llama3

Arriviamo dunque alla fase di analisi dei post che abbiamo recuperato attraverso il crawling. Come detto nel capitolo introduttivo, lo scopo di questo lavoro è quello di servirsi di un *Large Language Model* per l'analisi e lo scoring automatico dei post. In particolare il modello che abbiamo deciso di utilizzare è *"LLama 7b"*, un LLM sviluppato da meta avente circa 7 miliardi di parametri. Per utilizzare questo LLM è disponibile un framework chiamato **ollama** che mette a disposizione diversi language model come llama, mistral phi ecc... Questo framework permette di utilizzare i modelli sia da remoto che in locale, nel nostro caso dovendo processare una grande quantità di dati non possiamo utilizzare la versione remota per via dei limiti di utilizzo. Pertanto abbiamo optato per la soluzione locale. Il framework prevede di avviare il modello tramite il comando

```
ollama run llama3
```

una volta fatto questo avremo che il modello sarà in esecuzione e si potrà interagire con esso attraverso il protocollo http. Di default il llama sarà in ascolto sulla porta *11434* su localhost e si potrà interagire con il modello attraverso il seguente endpoint:

```
POST http://localhost:11434/api/generate
```

e nel corpo della richiesta andiamo a fornire un oggetto json della seguente forma:

```
{
    "model":"llama3"
    "prompt": <testo del prompt>
    "stream": False
}
```

dove il parametro stream viene impostato a **False** per far si che il modello risponda in un'unica soluzione anzichè *"token by token"*. Il modello risponderà a sua volta attraverso un oggetto json contenenti una serie di metadati più il testo della risposta etichettato secondo la chiave *'response'*.

2.3.2 Scoring

Abbiamo detto di voler assegnare a ciascun post uno score. Dunque dobbiamo definire come questo score debba essere fatto: per prima cosa abbiamo individuato il range in cui far variare tale score nell'intervallo $[0, 10]$ dove uno score pari a zero indica che il post viola completamente le regole dell'istanza in cui è stato pubblicato mentre uno score pari a 10 al contrario indica che il post rispetta diligentemente tutte le regole dell'istanza.

2.3.3 Quanti post per ogni istanza?

L'utilizzo del protocollo http per comunicare con il modello nonostante sia una scelta molto comoda per lo sviluppo di applicazioni che utilizzano gli LLM rende piuttosto lento il processamento dei post, che impiega in media dai 4 ai 6 secondi. Pertanto, risulta impossibile processare tutti 50 mila post raccolti per ciascuna istanza. Per venire in contro alle ai vincoli temporali e alla limitatezza in termini di risorse abbiamo deciso di adottare la seguente metodologia: per ciascuna istanza andiamo ad ordinare i post in ordine decrescente rispetto al numero di like, fatto ciò prendiamo i primi 100 post (quelli

con più like) e gli ultimi 100 post (quelli con meno like). Questa scelta è dovuta al fatto che i post con meno like, cioè quelli meno virali, hanno una maggiore probabilità di sfuggire al controllo dei moderatori dell'istanza. Quindi per ogni istanza processeremo un totale di **200** post rappresentativi sia della porzione virale dei post che di quella meno virale. Ciascuno di questi post verrà poi sottoposto alla valutazione di llama per 2 volte (vedi paragrafo che segue), per un totale di 400 interazioni con llama per ciascuna istanza.

2.3.4 Composizione del prompt e contesto

Diversi stufi hanno dimostrato che, allo stato attuale delle cose, quando si lavora con Large Language Models bisogna prestare particolare attenzione alle informazioni *"al contorno"* che si forniscono al modello. Infatti, la risposta restituita da un LLM in corrispondenza di un certo prompt può essere anche molto diversa in funzione del **ruolo** che si assegna ad esso. Per tale motivo, abbiamo deciso di processare ciascun post con 2 diversi prompt, i due prompt sono i seguenti:

1. `You are a decentralized social-network instance's moderator. Your role is to check if utent's post respects the instance rules. You must assign each post a score from 0 (if the post violate completely the instance's rules) to 10 (if the post respects all the instance's rules).`

2. `You are a decentralized social-network instance\'s owner. Your role is to check if utent's post respects the instance rules. In order to classify posts you have to assign each post a score from 0 (if the post violate completely the instance's rules) to 10 (if the post respects all the instance's rules). You must be very strict in your evaluations. Please pay a lot of attention and let's think step by step.`

Nel primo caso stiamo assegnando al modello il ruolo di **moderatore** invitandolo ad assegnare uno score tra 0 e 10 senza dare ulteriori direttive. Nel

secondo caso invece assegnamo al modello il ruolo di proprietario dell'istanza e oltre a questo rispetto al primo caso aggiungiamo 2 ulteriori direttive:

1. Stiamo invitando il modello ad essere molto severo nei suoi giudizi.
2. Stiamo invitando il modello a ragionare *step by step* quindi ci aspettiamo che analizzi separatamente le singole regole.

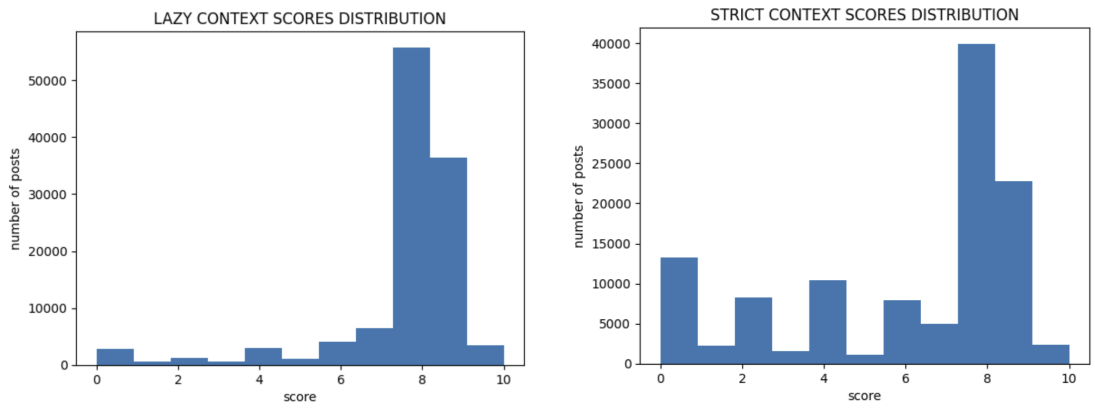
3 Presentazione dei risultati

Come già descritto nel precedente capitolo, ciascun post, è stato analizzato attraverso due prompt. Questi prompt differiscono sostanzialmente nella quantità di *"informazioni al contorno"* che si forniscono al modello, il primo, al quale ci riferiremo con l'appellativo *CONTEXT* assegna il ruolo di **moderatore** e chiede di assegnare uno score ai post; il secondo, al quale ci riferiremo con l'appellativo *STRICT_CONTEXT* assegna il ruolo di **proprietario** e chiede al modello di ragionare attentamente ed in modo severo quando assegna lo score. Dunque è interessante andare ad analizzare alcune statistiche riguardanti gli score sia considerando singolarmente i prompt che facendo dei paragoni tra essi. Iniziamo analizzando lo score medio assegnato dai due prompt:

- **CONTEXT mean score:** 7.7
- **STRICT_CONTEXT mean score:** 6.06

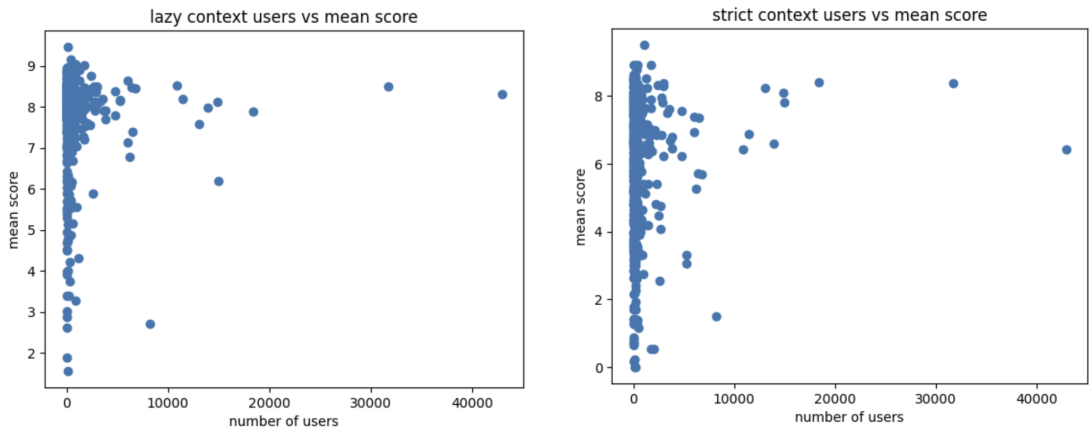
Cosa possiamo ipotizzare sulla base di questi dati? Intanto possiamo osservare che, come da aspettative, chiedendo al modello di essere più severo nelle sue valutazioni lo score mediamente è più basso. Inoltre, ricordandoci che lo score assegnato appartiene al range $[0, 10]$ sembrerebbe che, in media, siano almeno parzialmente disallineati rispetto alle regole dell'istanza in cui vengono pubblicati. Per avere una visione a scala più fine di questi dati possiamo andare ad analizzare le distribuzioni degli score.

Osserviamo i seguenti grafici:



Quello che si può osservare in entrambi i casi è che sicuramente la maggior parte delle istanze ha uno score medio alto (nel range $[8, 10]$). La differenza tra le 2 è che mentre nel caso del prompt *CONTEXT* abbiamo un picco molto più marcato in corrispondenza del range medio (con una coda di score bassi poco densa), nel caso del prompt *STRICT_CONTEXT* abbiamo un picco sul range medio meno marcato e una coda di valori bassi più densa. Dunque, utilizzando un contesto più stringente la distribuzione delle medie sugli score risulta essere meno sparsa. Dunque, mettendo insieme quanto visto con le medie globali e con le distribuzioni degli score medi cosa possiamo ipotizzare?. Secondo il nostro parere, nella maggior parte delle istanze mastodon l'attività di moderazione è portata avanti con discreto successo riuscendo a mantenere uno score sufficientemente vicino al completo allineamento alle regole. Tuttavia, la presenza di post poco virali che violano le regole e sfuggono al controllo dei moderatori tende a generare una coda di score bassi che abbassano di conseguenza la media degli score. Un'altro dato interessante risiede nelle **deviazioni standard**: nel caso del prompt *CONTEXT* abbiamo una deviazione standard pari a **1.28** mentre nel caso del prompt *STRICT_CONTEXT* la deviazione standard è pari a **2.30** quindi non solo mediamente abbiamo score più bassi, ma gli score assegnati hanno anche maggiore variabilità. Un ulteriore dato interessante da analizzare è il coefficiente di correlazione di pearson tra i due contesti che risulta essere pari a **0.55**, questo ci suggerisce che esiste una diretta correlazione (di intensità moderata). Quindi, presumibilmente,

una parte dei criteri con cui il modello assegna gli score nei due diversi casi è condivisa. Un'ultima che vorremmo analizzare, è il rapporto che c'è tra il numero di utenti attivi di una istanza e il suo score medio. Diamo uno sguardo ai seguenti grafici:



Possiamo osservare che il rapporto tra numero di utenti attivi e score medio non segue una particolare legge. Abbiamo che la maggior parte delle istanze hanno meno di 10 mila utenti attivi al mese e che tra queste gli score sono distribuiti secondo la distribuzione già vista sopra. Un dato interessante che possiamo inferire è che nelle istanze molto numerose (con più di 15 mila utenti attivi al mese) gli score sono tendenzialmente molto alti, il che ci porta a ipotizzare che **nelle istanze molto numerose l'attività di moderazione viene presa molto sul serio** il che è un dato positivo.

4 Conclusioni

Sulla base di quanto illustrato nelle precedenti sezioni possiamo concludere che l'utilizzo di Large Language Model per l'individuazione di post che violano le politiche di una community risulta essere una metodologia efficacie. La presente proposta rappresenta uno spunto di partenza per molti possibili sviluppi futuri, tra le altre cose, si potrebbero includere nelle informazioni al contorno fornite al modello, anche informazioni sul nodo della rete che ha scritto il post, nonchè informazioni topologiche come il grado di centralità, la closeness e/o una misura della capacità di un nodo di influenzare gli altri. Questo approccio sarebbe motivato dal fatto che se un nodo molto centrale ed influente viola spesso le regole della community, tenderà a propagare molto questo comportamento nella rete.Tuttavia questa è solo un esempio, il campo delle reti sociali può essere indagato da molti punti di vista differenti.

4.1 Nota sull'illegalità nei social network decentralizzati

Nell'introduzione abbiamo parlato di come mastodon sia un social network decentralizzato, questa caratteristica lo rende vantaggioso in quanto non raccoglie dati sugli utenti e non usa algoritmi di recommendation, risultando di fatto, più democratico rispetto ai social network tradizionali. Tuttavia, come ogni strumento creato dall'uomo, anche la tecnologia e più in particolare i social network possono essere usati per scopi benevoli e malevoli. Nel corso della nostra analisi, purtroppo, abbiamo constatato che la natura decentralizzata di questo social viene anche utilizzata per la diffusione di materiale **pedo-pornografico** e/o per la promozione di canali di diffusione di tali materiali. In particolare,ci tenevamo a denunciare una istanza in cui ci siamo imbattuti durante il crawling, tale istanza è pawoo.net. I post contenuti al suo interno contengono materiale pedo-pornografico e incitazioni alla pedofilia.Per ragioni etiche abbiamo escluso questa istanza dalle analisi documentate nelle precedenti sezioni. Di seguito viene mostrato degli screenshot di esempio per far comprendere l'entità del materiale contenuto all'interno di tale istanza che

sicuramente richiederebbe analisi più approfondite dalle autorità competenti. Ad aggravare ulteriormente i reati commessi in questa istanza, è il fatto che tutto questo viene fatto a scopo di lucro, cioè l'istanza risulta essere un vero e proprio **mercato** di scambio di materiale illegale.