Università degli studi di Padova
Dipartimento di Scienze Statistiche

Corso di Laurea Triennale in
Statistica per le Tecnologie e le Scienze

Relazione Finale

Uno studio di metodi di Topic Modeling

Relatore Prof. Massimo Melucci
Dipartimento di Ingegneria dell'Informazione

Laureando Sebastiano Urban
Matricola N 1177589

Anno Accademico 2021/2022

Indice

Introduzione

Negli ultimi decenni, grazie all'avvento del Web, è presente una quantità sempre maggiore di dati, in particolare di tipo testuale. Una tra le tecniche più utilizzate per l'analisi di questi testi è la modellazione degli argomenti (*topic modeling*). Questo tipo di strumento è usato per trovare informazioni precedentemente sconosciute ma potenzialmente di alta qualità da grandi raccolte di documenti. In particolare, lo scopo di questi modelli è assegnare un *topic* (argomento) ad ogni documento della collezione: questa informazione ci permetterà di avere una sintesi facilmente interpretabile del documento di riferimento.
I primi passi di questo elaborato saranno mossi nel campo dell'elaborazione del linguaggio naturale, meglio conosciuto come *natural language processing* (NLP), dove verranno illustrati tutti i passaggi da compiere per trasformare il testo grezzo in una tipologia di dato più complesso che meglio si adatta al tipo di elaborazione che si desidera ottenere. Nel percorso da compiere durante questa trasformazione, dopo aver attuato una prima pulizia del testo, si entrerà in contatto con strumenti come la lemmatizzazione e lo stemming, per far fronte ai principali problemi di semantica. Per concludere il capitolo riguardante l'elaborazione del linguaggio naturale sarà spiegata l'importanza dei termini presenti nella collezione e come questi possano essere preziosi in base all'occorrenza nel corpus. Verrà introdotta inoltre una possibile tecnica di riduzione del vocabolario, con lo scopo di eliminare i termini meno informativi.
Inoltre, sarà illustrato un possibile metodo per strutturare i documenti in un formato chiamato *bag of words* (BoW), dall'inglese "borsa di parole". Inoltre, applicheremo al nostro corpus di riferimento due diverse tipologie di bag of

words, spiegando così le metriche tf e tf-idf.
Successivamente a queste fasi preparatorie, per entrare nel vivo del topic modeling, sfrutteremo il modello *Latent Drichlet Allocation* (LDA) da adattare ai nostri dati. Sarà spiegato il funzionamento generale di questo modello e dopo averlo applicato al corpus, sarà possibile confrontarsi con delle scelte che questo modello in particolare ci mette a disposizione. Principalmente spetterà a noi capire quale sia il miglior numero di topic da adottare e per aiutarci in questo faremo uso di una metrica chiamata *topic coherence*, che verrà spiegata nel paragrafo a lei dedicata.
Oltre a condurre un'analisi di topic modeling, mostrando tutte le strategie adottate per migliorare il risultato finale, si vogliono illustrare le principali tecniche di visualizzazione dei dati (*data visualization*). La componente grafica gioca un ruolo sempre più importante nel contesto delle analisi; pertanto, verranno prese in considerazione più alternative, introducendo algoritmi come t-SNE e UMAP per produrre scatter plot da poter confrontare, ma parleremo anche di output grafici di maggior impatto come le word clouds.

Capitolo 1

Elaborazione del linguaggio naturale

1.1 Introduzione ai Dati

Come dataset di riferimento è stato scelto una collezione di documenti in lingua italiana, in particolare una raccolta di 1.700.000 pagine prelevate da Wikipedia. Questo corpus così formato è particolarmente interessante poiché tratta principalmente documenti in lingua italiana, ma sono comunque presenti termini o citazioni in altre lingue straniere, specialmente l'inglese. Essendo una grande collezione proveniente da una delle più note enciclopedie online ci permette di confrontarci con un'ampia varietà di argomenti. Come è comprensibile dalla numerosità dei documenti, siamo dinnanzi ad un dataset davvero grande. Questo, in generale, non rappresenta un problema poiché le tecniche come quelle analizzate in questo progetto nascono proprio dall'esigenza di elaborare grandi moli di dati; nel caso specifico di questa relazione è stato tuttavia deciso di lavorare con una parte di questi documenti per ridurre i tempi di elaborazione e perché numeri così grandi richiedono memorie di capacità molto elevata per lo stoccaggio delle informazioni durante l'elaborazione. Quindi è stato effettuato un campionamento casuale dal dataset principale dal quale sono stati prelevati 10.000 documenti. Inoltre, per permettere la riproduzione di questo esperimento partendo dagli stessi dati è stato fissato il random seed prima del campionamento.

1.2 Elaborazione preliminare dei Dati

I dati di tipo testuale richiedono una fase di pre-processamento prima di poter essere elaborati. Questa prima revisione dei dati è essenziale per poter eliminare dai documenti una serie di segnali "rumorosi" che non sono utili al fine di elaborazioni successive.

1.2.1 Segni di punteggiatura

La prima pulizia della quale ci occupiamo riguarda i segni di punteggiatura. Simboli come i punti e le virgole sono utili nel linguaggio per capire quando finisce una frase o per dare la giusta intonazione di lettura; tuttavia, i nostri modelli si basano esclusivamente sulla frequenza delle parole, per cui il primo passo consiste nel rimuovere tutti questi simboli dai documenti. Si vuole comunque sottolineare che questa decisione viene presa basandosi sulle tecniche che verranno approfondite durante l'elaborato, ma non è un passo obbligatorio nell'ambito dell'elaborazione del linguaggio naturale. Ad esempio, esistono modelli molto recenti nell'ambito del NLP che cercano di catturare schemi all'interno del documento, basandosi proprio sui segni di punteggiatura per carpire l'ironia delle frasi. Un diverso tipo di rimozione può interessare una serie di simboli che negli ultimi anni sono diventati d'uso comune nei testi, ovvero gli emoji e altri simboli grafici.

1.2.2 Stopwords

Entrando nel vivo del Natural Language Processing, il secondo passo riguarda le cosiddette "stopwords", ovvero un insieme di parole presenti in molte lingue, sebbene in proporzioni diverse, che sono ritenute non informative a causa del loro uso troppo comune. Queste parole non danno nessuna informazione riguardo l'argomento del quale si parla in un documento; quindi anch'esse come i segni di punteggiatura sono rimosse dal corpus. Nel caso della lingua italiana alcune stopwords comuni sono ad esempio: "quando", "allora", "e" "anche", etc. Nel nostro caso specifico, sono state prese in considerazione sia le stopwords della lingua italiana, sia quelle della lingua inglese

poiché, come detto in precedenza, sono presenti molti termini inglesi anche nella sezione in italiano di Wikipedia.
Un ulteriore snellimento del corpus può essere fatto eliminando anche i caratteri di tipo numerico. Questa operazione solitamente è a discrezione di chi conduce l'analisi. In alcuni casi, i numeri possono essere esplicativi per condurre l'analisi desiderata, ma in questo caso sono stati eliminati poiché ritenuti poco informativi. Data la natura dei testi che prendiamo in considerazione, questi riporteranno molti numeri al loro interno, come: date, anni, abitanti, etc. Si vuole evitare collegamenti errati bastati su cifre del tutto incorrelate. Ad esempio, non vogliamo si crei un'associazione tra un paese con 1900 abitanti e un manufatto storico del 1900.

1.2.3 Codice

Passando all'atto pratico della questione, tutte le operazioni di cui abbiamo parlato finora sono racchiuse un'unica grande funzione creata ex novo per lo studio specifico a cui ci siamo approcciati. Questa funzione in realtà si occupa anche di compiere altre operazioni particolari nel corpus che affronteremo nei prossimi paragrafi, come lo stemming e la creazione di lemmi.

```
1 def Create_Corpus(dict_input, stopwords, stemming = True,
2                   lemmatization = True, remove_numeric = True,
3                   cut_vocabolary = True, min=0.025, max=0.975,
4                   hist = False, report = True):
```

Alcuni di questi argomenti verranno affrontati in futuro, per ora possiamo analizzare gli input che già conosciamo:

- dict_input -> corrisponde ad un dizionario Python formato dall'identificatore e dal testo del documento, in questo modo id: testo, id: testo, ...
- stopwords -> è un array di stringhe, come ['e', 'allora',...]. Nel nostro caso le stopword sono state prelevate dalla libreria NLTK (from nltk.corpus import stopwords) (Loper e Bird 2002).

- remove_numeric -> se impostata su True, tutte le cifre verranno rimosse durante la creazione del corpus.

Gli output della funzione sono due:

- unpreprocessed_corpus -> un array di stringhe del tipo [doc1, doc2, ...], dove l'unica modifica apportata a quest'ultimo è aver reso tutti i caratteri minuscoli. Come vedremo meglio più avanti questo ci servirà per alcuni modelli basati sul contesto delle frasi che pre-elaborando i documenti viene perso.

- preprocessed_corpus -> un array di stringhe del tipo [doc1, doc2, ...], dove i documenti avranno subito tutti i cambiamenti da noi scelti tramite le proprietà della funzione.

1.3 Problemi di semantica

Trattando documenti di tipo testuale, non possiamo non tenere in considerazione le problematiche legate al significato delle parole, in particolare si parla di polisemia e sinonimia. Una parola si dice polisemica quando ha più di un significato; ad esempio, con la parola "piano", si potrebbe intendere un pianoforte, o l'azione del "fare piano", senza fare rumore, o ancora si potrebbe intendere il piano di un palazzo. Come si può intuire, non è facile interpretare il significato di una singola parola; alcune tecniche, infatti, si basano sul contesto per cercare di distinguerne i vari usi. In ogni caso questo non è l'unico problema legato al significato, in quanto dobbiamo confrontarci anche con le parole sinonime, come ad esempio "viso", "volto" e "faccia". Anche se, nel linguaggio comune vengono usati diversi termini per esprimere lo stesso concetto, al fine di identificare un topic comune, vorremmo che queste parole rappresentassero lo stesso topic, spesso però fare questo diventa molto complesso utilizzando strumenti che si basano esclusivamente sulla frequenza delle parole, perché a tutti gli effetti stiamo trattando parole differenti anche se con lo stesso significato.
Per affrontare questo tipo di problema, si possono adottare tecniche conosciute come stemming e lemmatizzazione.

1.3.1 Stemming

In particolare, lo stemming (dall'inglese stem -> radice o stelo) è una tecnica che sostituisce ad ogni parola all'interno del documento (anche detta "token") con la sua radice grammaticale. In questo modo si cerca di associare più derivazioni di una stessa parola ad un'unica radice comune detta appunto stem. Ad esempio, parole come correre, corro, corriamo e correremo vengono ridotte tutte alla stessa radice "corr". Questa tecnica ha due principali vantaggi: il primo è che vengono associati tra loro i documenti che originariamente non condividono parole, pur accomunate dallo stem. Il secondo vantaggio invece è quello di poter creare un "vocabolario" di dimensioni ridotte. Con vocabolario in questo caso intendiamo l'insieme di termini che appaiono almeno una volta all'interno dell'intera collezione. È chiaro che questi vocabolari possono avere dimensioni davvero molto grandi, tuttavia, se alle parole sostituiamo gli stem allora anche la dimensione totale diminuirà in base a quanti più termini hanno una radice comune. Da notare però anche uno svantaggio intrinseco allo stemming, ovvero la parziale perdita di semantica. Ad esempio leggiamo "corr" in una frase, non sappiamo dare un significato preciso.

1.3.2 Lemmatizzazione

La lemmatizzazione, invece, ha uno scopo simile a quello dello stemming, ovvero trovare un token comune che rappresenti un insieme di parole che hanno approssimativamente lo stesso significato. Questo procedimento non si basa però semplicemente sul troncare la parola, ma su un meccanismo molto più complesso. Tornando all'esempio di prima, se applicassimo una lemmatizzazione alla serie di parole correre, corro, corriamo e correremo verrebbero tutte sostituite con il lemma "correre". Questo tipo di pratica è non troppo complessa se si parla della lingua inglese con la sua grammatica non eccessivamente articolata e le poche forme irregolari; diverso è se si parla della lingua italiana. A causa delle molte coniugazioni verbali e delle eccezioni grammaticali, la lemmatizzazione richiede qualche attenzione in più. In Python, grazie alla libreria SpaCy (Honnibal e Montani 2017) è possibile creare lemmi per la maggior parte delle lingue; nel sito è disponibile una sezione

dalla quale scaricare un "dizionario" per ognuna di esse. Questo file non è altro che una lunghissima lista di parole in italiano alla quale viene associato il relativo lemma, come una tabella di conversione. Le funzionalità offerte da questa libreria non finiscono qui, in quanto è presente una terza colonna dove viene riportata la tipologia di parola per definire, ad esempio, che si tratti di un verbo, un pronome o un aggettivo. Questo ci permette di poter decidere quali lemmi mantenere e quali invece eliminare dal corpus, perché ritenuti poco informativi al fine dell'analisi. Ad esempio, potremmo decidere di non considerare i nomi propri.

1.3.3 Codice

```
def Create_Corpus(dict_input, stopwords, stemming = True,
                  lemmatization = True, remove_numeric = True,
                  cut_vocabolary = True, min=0.025, max=0.975,
                  hist = False, report = True):
```

Alcuni di questi argomenti verranno affrontati in futuro, per ora possiamo analizzare gli input che già conosciamo:

- stemming -> Se impostato su True, viene eseguito lo stemming come descritto in precedenza. In particolare, utilizzando lo "SnowBallStermmer" della libreria NLTK (Loper e Bird 2002).

- lemmatization -> Se impostato su True, viene eseguita la lemmatizzazione come descritta in precedenza.

È doveroso aggiungere una piccola nota: per ovvie ragioni non è possibile eseguire sia lo stemming che la lemmatizzazione sull'insieme di documenti in ingresso alla funzione. Pertanto, è presente una verifica all'avvio del procedimento che, nel caso in cui entrambe le proprietà fossero impostate su True, restituisce un errore.

1.4 Riduzione del vocabolario

Da quanto detto finora, è chiaro che lo scopo principale di questa prima fase di pre-elaborazione è eliminare tutta l'informazione superflua all'interno del corpus, in modo tale che ogni token che costituisce un documento sia prezioso per l'associazione del topic. Dato che il legame tra due documenti viene stabilito tramite la proporzione di token in comune tra essi, è intuitivo pensare che una parola, che all'interno della collezione si presenta una sola volta, non possa identificare nessun legame tra i documenti. Ampliando lo stesso concetto possiamo pensare invece che un token che appare in soli 2 o 3 documenti su un totale di 10k non può rappresentare uno schema utile al fine di trovare un insieme di documenti dall'argomento comune. Allo stesso modo però, possiamo pensare ad una parola troppo frequente, che ad esempio compare nel 90% dei documenti. Anch'essa non rappresenta un'informazione utile, poiché il gruppo di documenti che identifica risulta troppo vasto.
In questa fase, infatti, vogliamo trovare il modo di eliminare questi token rari o molto frequenti all'interno del nostro vocabolario; tuttavia, l'eliminazione non è l'unica strategia applicabile per risolvere questo tipo di problema. Esistono infatti dei metodi alternativi che, anziché ridurre il vocabolario, pesano le parole applicando pesi piccoli alle parole rare o molto frequenti, rispetto alle parole con una frequenza compresa in un intervallo predefinito; parliamo più approfonditamente di questa tecnica nel corso dell'elaborato con l'introduzione del sistema noto con il termine "Term Frequency-Inverse Document Frequency" (TF-IDF).
Nel nostro caso, ricordando che lo strumento che vogliamo creare non dev'essere basato sul dataset selezionato, ma mantenere un approccio più generale possibile, la soluzione adottata è di tipo probabilistica. L'operazione che viene eseguita infatti, è in primo luogo quella di stimare la distribuzione basata sulle frequenze delle parole; in questo modo possiamo selezionare i quantili dove "tagliare" la distribuzione, in modo tale da tralasciare le code destra e sinistra che superano i quantili selezionati.

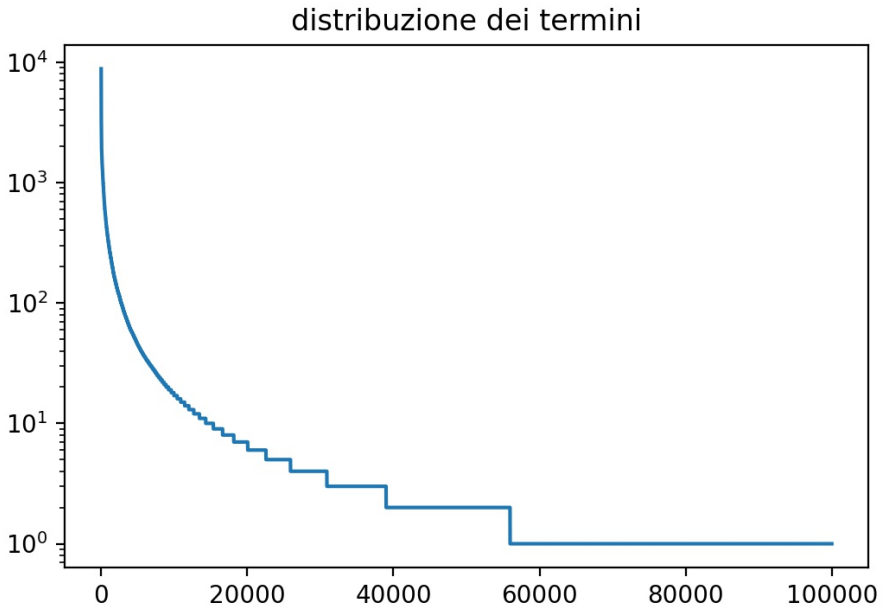

Figura 1.1

Dal grafico soprariportato possiamo notare come su un totale di 120.000 parole, solo 56.000 circa hanno una frequenza superiore ad 1. Questo ci permette di dimezzare la dimensione del vocabolario con una minima perdita di informazione, poiché da quanto detto finora le parole rare sono poco informative per la creazione legami tra i documenti.

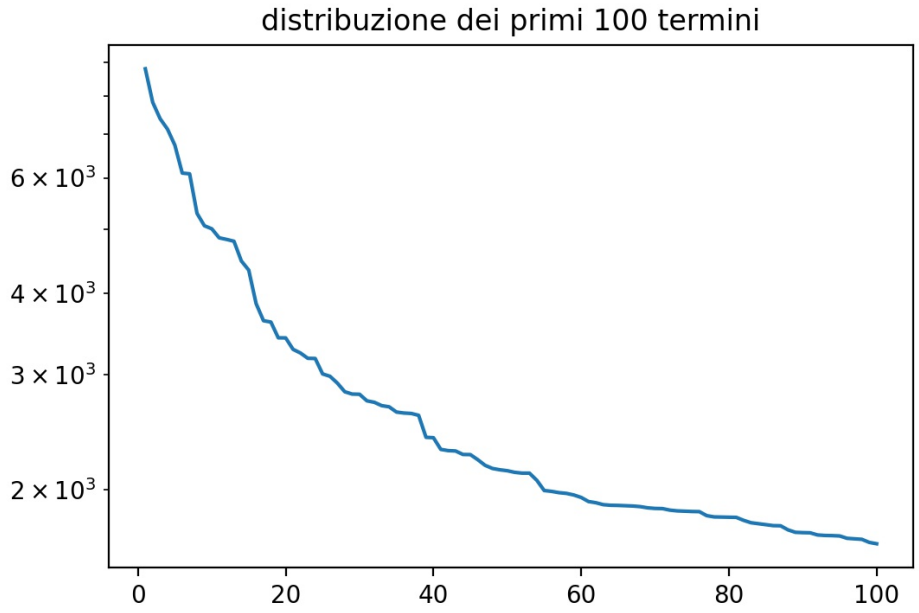

Figura 1.2

Da questo secondo grafico riguardante la distribuzione dei termini possiamo vedere più nel dettaglio l'andamento della prima centinaia di parole. Come possiamo notare, le prime unità hanno frequenze comprese tra 6.000 e 9.000. Basta spostarci di poco verso destra per vedere questa curva scendere molto rapidamente, infatti, già il sessantesimo termine più frequente ha un'occorrenza che scende sotto i 2.000.
Questo conferma quanto detto in linea teorica, ovvero, la distribuzione è caratterizzata da molti termini poco frequenti nella coda a sinistra e da pochi termini molto frequenti nella coda destra.
Passando ora all'atto pratico della questione, riprendiamo la funzione che ha caratterizzato l'intero processo descritto in questo primo capitolo riguardante l'elaborazione del linguaggio naturale, per analizzare gli ultimi elementi.

1.4.1 Codice

```
def Create_Corpus(dict_input, stopwords, stemming = True,
                  lemmatization = True, remove_numeric = True,
                  cut_vocabolary = True, min=0.025, max=0.975,
                  hist = False, report = True):
```

Abbiamo già visto in precedenza la maggior parte delle proprietà, così come gli output della funzione, Concentriamoci ora su:

- cut_vocabolary -> Se impostato su True, viene eseguita la riduzione del vocabolario come descritta finora, ovvero tramite la distribuzione basata sulle frequenze dei token.
- min -> rappresenta il quantile inferiore della distribuzione dove verrà eseguito il taglio. Di default è impostato a 2,5
- max -> rappresenta il quantile superiore della distribuzione dove verrà eseguito il taglio. Di default è impostato a 97,5
- hist -> Se impostato su True, restituisce l'istogramma della distribuzione del vocabolario
- report -> Se impostata su True, fornisce alla fine del processo una serie di informazioni sui quantili reali della distribuzione e sulla numerosità del vocabolario prima e dopo la riduzione

Capitolo 2

Bag of Words

2.1 Che cos'è una Bag of Words?

Dopo la fase di pulizia del testo, ci ritroviamo con un vettore di parole, o meglio token, a descrivere ogni documento della collezione. Lavorare con questi vettori non è ottimale poiché le parole all'interno spesso sono ripetute molte volte data l'alta frequenza di alcuni termini come visto in precedenza. Per risolvere questo problema si usa una particolare forma di archiviazione dei documenti chiamata "bag of words" (BoW) che letteralmente in italiano significa "borsa di parole". Il promo passo che viene fatto a livello informatico, anche se non strettamente necessario, è quello di sostituire i termini con un identificativo numerico. Questi stessi identificativi li ritroviamo anche nel vocabolario creato partendo dalla collezione. Ora quindi, ci si ritrova con il documento descritto da una lista di identificatori numerici anzichè token di tipo testuale, anche se il problema della ripetizione di elementi all'interno del vettore non è ancora stato risolto. Un primo modo di procedere potrebbe essere quello di prendere tutti gli elementi della lista che descrive un documento in modo univoco, ovvero prendendo gli elementi una sola volta, non considerando le eventuali ripetizioni. Questo metodo può essere utilizzato ma va tenuto in considerazione lo svantaggio che porta; Procedendo in questo modo viene assegnato ad ogni token che originariamente costituiva il documento uno stesso peso, mettendo così allo stesso livello termini rari e termini

molto frequenti. Per ovviare a questo tipo di svantaggio sono state create diverse metriche da poter associare ad ogni identificatore in modo da soppesare differentemente i token in base ad alcune caratteristiche, prima tra tutte, la frequenza assoluta.

2.2 Bag of Words basata sulle frequenze

Come spiegato poco sopra, se si desidera aggiungere informazione alla nostra bag of words è necessario stabilire una misura da usare come peso per ogni token del documento. Una delle metriche più classiche per fare questo è sicuramente la frequenza assoluta. Infatti, come riporteremo in seguito con un esempio pratico, il legame tra un termine e il relativo documento può essere stabilito proprio dal numero di volte che il token si presenta all'interno del testo.
L'aumento d'informazione introdotta da questo metodo è sicuramente un vantaggio, ma con esso ci sono anche degli svantaggi da non trascurare. Principalmente, quando si usa la frequenza assoluta come metrica, viene dato maggior peso ai token più usati in un documento, ma come è stato fatto notare nel capitolo riguardante il NLP, i termini troppo frequenti sono poco informativi, questo potrebbe portare ad una non corretta classificazione del documento in seguito, poiché rappresentato da token troppo comuni.

2.2.1 Esempio

Nota: in questo esempio nella BoW vengono usati i token e non i corrispettivi identificatori per una maggiore comprensione, ma come detto in precedenza, a livello informatico si usa un numero come identificatore univoco per ogni token.
Doc:

1. Il mio cane abbaia ai vicini
2. Il mio gatto gioca con il gatto dei vicini
3. Il cane dei vicini si chiama Oscar

BoW:

1. {“il”: 1, “mio”: 1, “cane”: 1, “abbaia”: 1, “ai”: 1, “vicini”: 1}
2. {“Il”: 2”, “mio”: 1, “gatto”: 2, “gioca”: 1, “con”: 1, “dei”: 1, “vicini”: 1}
3. {“Il”: 1, “cane”: 1, “dei”: 1, “vicini”: 1, “si”: 1, “chiama”: 1, “Oscar”: 1}

2.2.2 Codice

```
1 from gensim.corpora.dictionary import Dictionary
2 corpus_split = [d.split() for d in preprocessed_corpus]
3 id2word = Dictionary(corpus_split)
4 corpus_bow = [id2word.doc2bow(text) for text in corpus_split]
```

Non è difficile creare una propria funzione che generi la Bag of Words basata sulle frequenze, poiché si tratta di una concatenazione di cicli for per scorrere il corpus e i token all’interno di ogni documento, utilizzando poi un contatore per tener traccia delle frequenze. Tuttavia, per una questione di efficienza computazionale, è preferibile utilizzare la libreria “gensim” (Rehurek e Sojka 2011) che integra un modulo “Dictionary” appositamente creato per questo tipo di conversioni. Come possiamo notare dal codice riportato, oltre ad importare il modulo e a creare quello che viene chiamato “corpus_split” (che rappresenta l’insieme di documenti suddivisi in token), vengono eseguite altre due operazioni. “id2word” è un oggetto python che memorizza gli ID univoci che vengono assegnati ad ogni parola presente nel corpus. Questo oggetto mette a disposizione delle funzioni, come possiamo vedere nell’ultima riga riportata, dove si chiede con il comando “id2word.doc2bow(text)” di convertire ogni documento in una lista di (IDtoken, Freq) creando così la relativa BoW.

2.3 Bag of Words basata su Tf-Idf

Il sistema di pesatura tf-idf (term frequency–inverse document frequency) è spesso usato nell'ambito dell'information retrieval data la sua completezza nel fornire un peso ad un termine, basandosi su più aspetti come: frequenza del termine, lunghezza del documento e grandezza del corpus. Verrà mostrato attraverso un esempio pratico come viene calcolata questa metrica, ma prima definiamola in linea teorica. Come si può notare il sistema tf-idf è composto dall'unione di due sotto misure: la prima detta "tf", che può essere usata anch'essa come metrica di pesatura, e la seconda "idf" che rappresenta in qualche modo un fattore di aggiustamento, analizzeremo di seguito la formula nel dettaglio.
Analizziamo le due componenti partendo da quella "tf":

$$tf_{i,j} = \frac{n_{i,j}}{|d_j|}$$

Dove $n_{i,j}$ è la frequenza del termine i nel documento j, mentre $|d_j|$ rappresenta la dimensione del documento j espressa come numero di token. Passiamo ora alla componente "idf":

$$idf_i = log_{10} \frac{|D|}{|\{d : i \in d\}|}$$

Dove |D| è il numero di documenti nel corpus, mentre il denominatore è il numero di documenti che contengono il termine i. Infine, non ci resta che unire le due componenti:

$$(tf - idf)_{i,j} = tf_{i,j} * idf_i$$

2.3.1 Esempio

Doc:

1. Il mio cane abbaia ai vicini
2. Il mio gatto gioca con il gatto dei vicini
3. Il cane dei vicini si chiama Oscar

BoW:

1. {“il”: 0, “mio”: 0.029, “cane”: 0.029, “abbaia”: 0.079, “ai”: 0.079, “vicini”: 0}
2. {“Il”: 0, “mio”: 0.019, “gatto”: 0.1, “gioca”: 0.053, “con”: 0.053, “dei”: 0.019, “vicini”:0}
3. {“Il”:0, “cane”: 0.025, “dei”: 0.025, “vicini”: 0, “si”: 0.068, “chiama”: 0.068, “Oscar”: 0.068}

2.3.2 Codice

```
from gensim.models import TfidfModel
model_tfidf = TfidfModel(corpus_bow)
corpus_tfidf = model_tfidf[corpus_bow]
```

Per calcolare la BoW con lo schema di pesatura tf-idf ancora una volta useremo la libreria “gensim” che ci fornisce un modulo chiamato “TfidfModel”. Come possiamo vedere, il modello viene applicato direttamente alla BoW calcolata in precedenza basandoci sulle frequenze assolute; infatti, questa corrisponde proprio alla parte Tf del modello. Il risultato che ci viene restituito da questo modello, salvato sulla variabile “corpus_tfidf”, per ogni documento fornisce una lista di (IDtoken, peso Tf-Idf) per ogni token univoco all’interno del documento.

Capitolo 3

Latent Dirichlet Allocation

3.1 Che cos'è Latent Dirichlet Allocation?

Latent Dirichlet Allocation (LDA) identifica un modello generativo ancora oggi molto utilizzato nel campo del Topic Modeling, anche se presentato per la prima volta nel 2003 (Blei, Ng, & Jordan, 2001). La visione sulla quale si basa questa tecnica è che ogni documento viene considerato come un insieme di parole che, combinate tra loro, formano uno o più sottoinsiemi di argomenti latenti e, ciascun argomento (topic) è caratterizzato da una particolare distribuzione di termini. In sostanza quello che avviene all'interno di questo modello è la determinazione di una distribuzione da assegnare ad ogni documento; per ottenere questo risultato, ci si basa sulle parole presenti in esso. Come suggerisce il nome del modello, la distribuzione probabilistica basata sui termini considerati come variabili casuali, è una distribuzione di Dirichlet. Allo stesso modo si calcolano delle distribuzioni anche per i topic latenti basandoci sui termini che lo compongono, dove naturalmente i token con frequenze più alte caratterizzano maggiormente il topic e la relativa distribuzione (ancora una volta si sta parlando di una distribuzione di Dirichlet). Quindi, semplificando, quello che fa questo algoritmo è confrontare le distribuzioni di documenti e topic per poi abbinare quelle più simili tra loro. Questa operazione non è così semplice: come detto in precedenza ogni documento potrebbe trattare più di un topic, cosa che ci impone di dover

assegnare ad ogni topic un peso di partecipazione al singolo documento. All'atto pratico questo viene fatto tramite una distribuzione multinomiale che ha lo scopo di trovare le similarità tra topic e documenti e fornirci anche delle percentuali relative agli argomenti.

3.2 Applicazione ai Dati

Vediamo ora in questo paragrafo come mettere in pratica quanto detto fin ora, utilizzando come in precedenza la libreria "gnsim".

```
lda_model = gensim.models.LdaMulticore(corpus=corpus_tfidf,
                                       id2word=id2word,
                                       num_topics=Numero_topic_selezionato,
                                       random_state=100,
                                       chunksize=1000,
                                       passes=25,
                                       per_word_topics=True)
```

- corpus -> richiede la nostra collezione in formato BoW, sta a noi in questo caso scegliere se passare la Bow basata sulle frequenze oppure con il modello Tf-Idf.
- id2word -> questo input corrisponde al vocabolario creato in precedenza con l'apposito formato richiesto da gensim
- num_topic -> è richiesto il numero di topic da individuare, ma approfondiremo l'argomento in seguito
- random_state -> corrisponde ad un random seed, anche se essendo il modello probabilistico basato su più fattori aleatori si ha una parziale riproducibilità dei risultati che saranno comunque leggermente differenti tra loro
- chunksize -> indica il numero di documenti da elaborare ad ogni iterazione

- passes -> è un parametro che regola quanti cicli del corpus deve compiere il modello durante l'apprendimento, questo parametro è particolarmente importante per la precisione dei risultati finali

Una volta terminata la fase di training del modello, possiamo visualizzare la stima delle probabilità di ogni topic per ogni documento con il seguente comando:

```
topic_for_doc = lda_model.get_document_topics(bow = corpus_bow,
                                              minimum_probability = 0)
```

L'output è per ogni documento è una lista grande tanto quanto il numero di topic selezionato in fase di stima del modello, contenente delle tuple così formate, (IDtopic, Probabilità).

3.3 Topic Coherence

Come descritto nell'articolo "Exploring the Space of Topic Coherence Measures" (Röder, Both e Hinneburg 2015), la topic coherence (coerenza degli argomenti) è una metrica appositamente sviluppata per la valutazione della "bontà" dei topic prodotti da un modello. Il punto chiave per capire come funziona questa metrica è concentrarsi sulla parola "coerenza" (coherence). Di solito, quando si parla di coerenza, si parla di una caratteristica di cooperazione. Ad esempio, un insieme di argomenti è coerente se si confermano a vicenda; infatti, ciò che la metrica topic coherence valuta è quanto bene un argomento è "supportato" da un set di testi di riferimento (corpus). Utilizzando statistiche e probabilità tratte dal corpus, in particolare focalizzate sul contesto delle parole, assegna un punteggio di coerenza ad un argomento. Questo aspetto è particolarmente importante perché fa capire che la coerenza di un topic non dipende solo dalle parole che lo formano, ma anche dal corpus alla quale facciamo riferimento.
Quello che si desidera dunque è che ogni topic generato dal nostro modello abbia un valore alto di "coerenza", specificando che l'intervallo nel quale è costruita questa metrica è compreso tra 0 e 1.

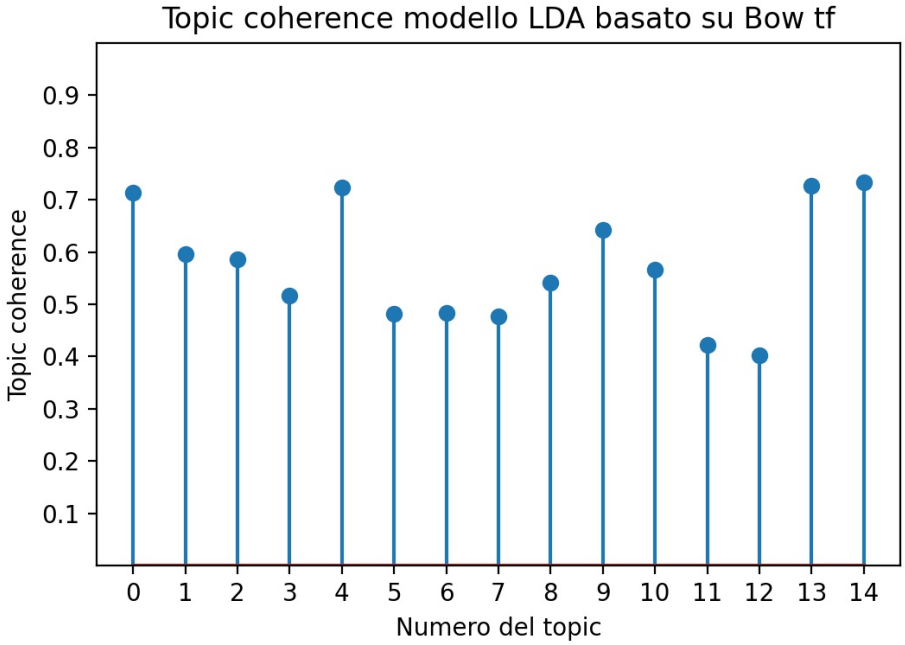

Figura 3.1

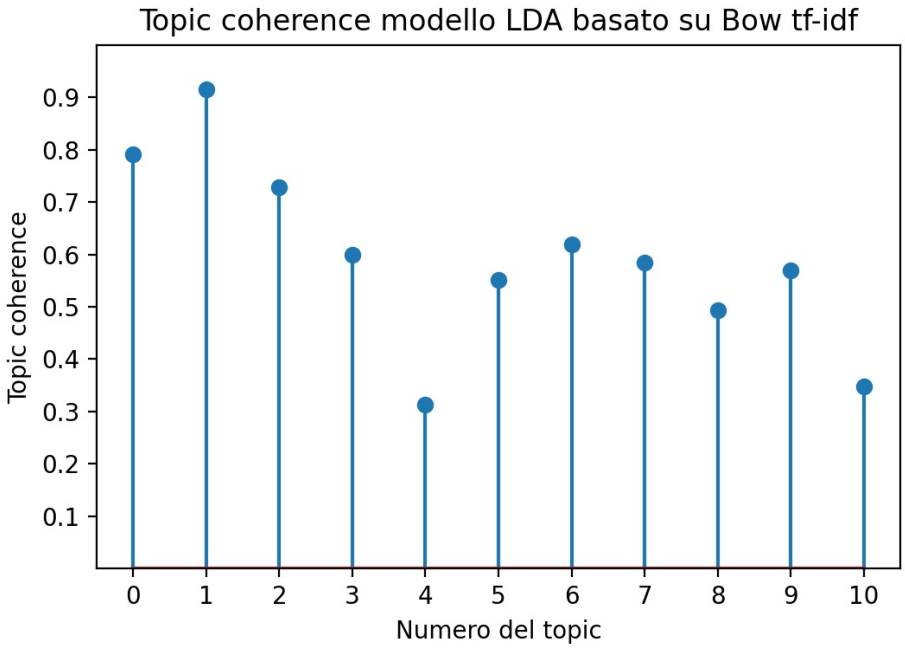

Figura 3.2

Da questi due grafici possiamo notare i differenti valori di coerenza dei topic prodotti dal modello LDA utilizzando due diverse configurazioni. Il primo grafico è prodotto utilizzando la BoW basata sulla frequenza dei termini, in cuivediamo come ci sia una minor variabilità tra le altezze delle colonne; infatti, i topic con una minor coerenza si stabilizzano attorno allo 0.4, mentre quelli con coerenza maggiore superano di poco lo 0.7, con una media di circa 0.57. Osservando il secondo grafico basato sulla BoW tf-idf, si può notare una maggiore variabilità generale con colonne di altezza attorno allo 0.3 nel caso peggiore, che arrivano anche a punte di 0.9 nel caso migliore. La media in questo caso supera la precedente (0.57) con un valore di circa 0.59.
Si vuole far notare che i due grafici riportano un diverso numero di topic come si può vedere dall'asse delle ascisse. Questo perché si è alla ricerca della miglior configurazione possibile per i nostri modelli; pertanto, anche la scelta del numero di topic sarà motivo di discussione e verrà affrontato nel prossimo paragrafo.

3.4 Numero di topic

Un aspetto fondamentale per il nostro modello LDA è proprio la scelta del numero di topic, per questo motivo è stata introdotta la metrica "topic coherence" analizzata in precedenza. Se pensiamo ad un modello con un numero elevato di topic, questo ci porta ad avere pochi documenti per ognuno di essi, aumentando così la coerenza dei singoli topic, ma abbassando di molto la diversità tra i topic stessi, aspetto della quale la nostra metrica di valutazione tiene conto. Inoltre, ricordiamo che l'idea alla base delle tecniche di topic modeling è che ci siano "pochi" topic latenti sottostanti al corpus. D'altro canto, avere una configurazione con troppi pochi gruppi, porta ad unire temi diversi sotto lo stesso topic, abbassando così la coerenza del topic stesso. Come si può intuire, si tratta di trovare il giusto compromesso, per fare questo ci basiamo proprio sula metrica introdotta in precedenza, ovvero la topic coherence.

Passando all'atto pratico, è stata creata una funzione che esegue varie configurazioni del modello LDA, calcolando ad ogni iterazione la topic coherence. Un aspetto da non trascurare è che il modello in questione è probabilistico, questo vuol dire che ad ogni esecuzione produrrà risultati leggermente diversi tra loro. Per evitare di basarci su una singola esecuzione, che potrebbe produrre valori lontani dalla media, si calcola il modello più volte con la stessa configurazione, per poi riportare il valore medio di topic coherence.

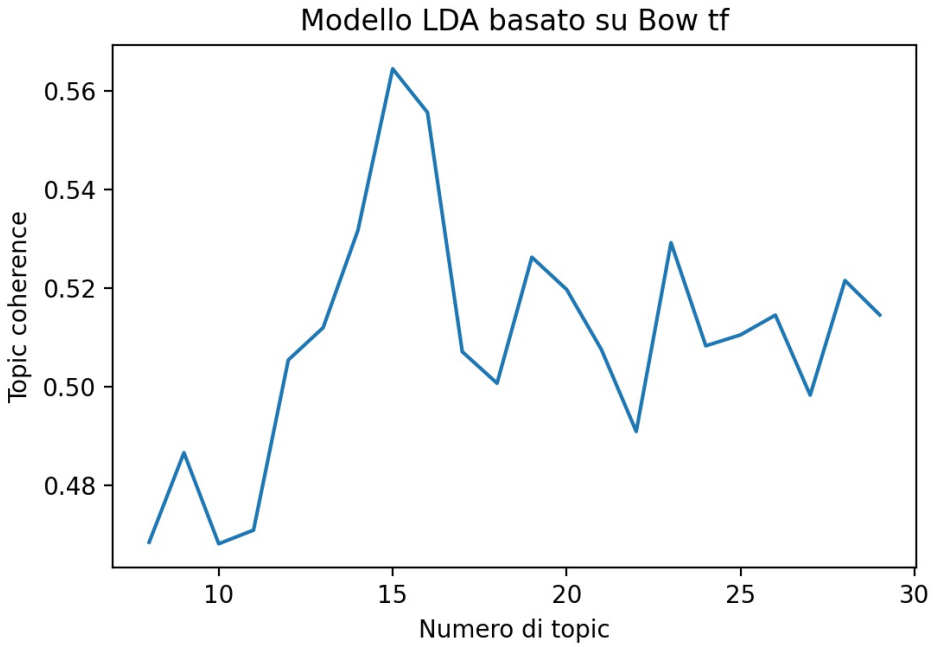

Figura 3.3

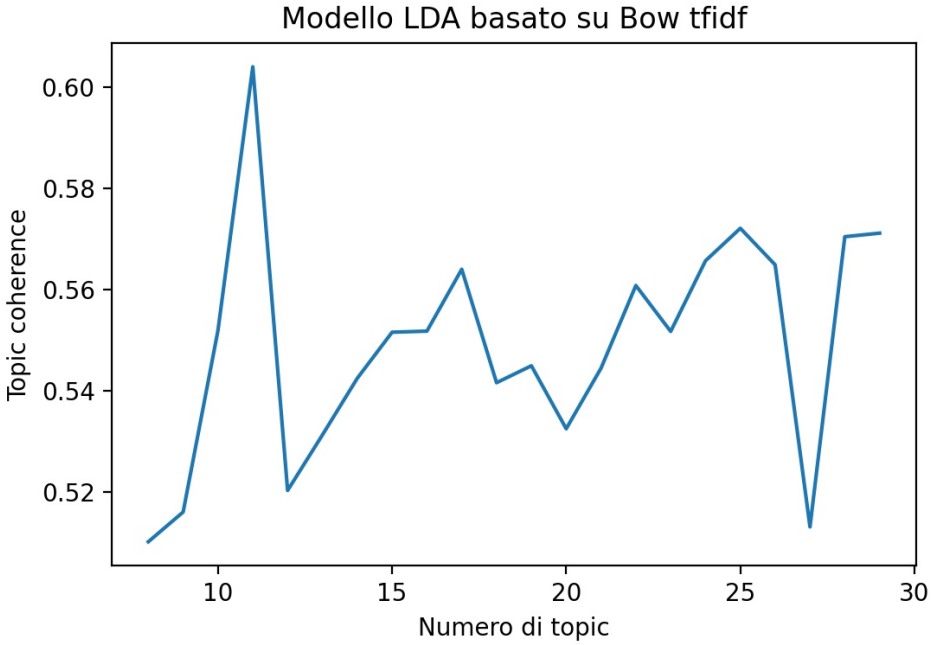

Figura 3.4

Come fatto notare in precedenza, la stessa tipologia di modello, eseguita a partire da due diverse bag of words, produce due risultati diversi, nonostante il corpus di partenza sia lo stesso, con l'unica differenza di essere "memorizzato" in maniera differente. In conclusione, d'ora in poi, si useranno due modelli LDA: il primo basato su una bag of words di tipo tf (calcolata a partire dalle frequenze) con 15 topic, un secondo modello basato su una BoW di tipo tf-idf con 11 topic.

Capitolo 4

Visualizzazione dei dati

4.1 Introduzione alla visualizzazione

La componente di visualizzazione dei dati (data visualization) sta assumendo un ruolo sempre più importante all'interno delle analisi odierne. A differenza dei complessi modelli matematici o delle implementazioni informatiche, la componente grafica, espressa tramite schemi o grafici, è d'impatto immediato e comprensibile alla maggior parte degli utenti. Con l'avvento dei big data, ovvero dati ad elevata dimensionalità, la rappresentazione grafica ha raggiunto un livello superiore di difficoltà. Solitamente l'uomo è abituato a considerare lo spazio attraverso tre dimensioni al massimo, come lo spazio che ci circonda. Questa tipologia di dati invece si esprime in molte più dimensioni, nell'ordine delle centinaia o migliaia, per questo ancor prima di visualizzazione si parla di tecniche di riduzione della dimesionalità. Esistono inoltre delle particolari tecniche che non proiettano in uno spazio le nostre unità statistiche, ma che sono comunque molto utili al fine di trasmettere un concetto. Nello specifico vogliamo introdurre le "word clouds" (nuvole di parole), che ci permettono di capire l'argomento trattato da un topic tramite la visualizzazione dei principali token che lo compongono. La caratteristica principale delle word clouds è la grandezza dei caratteri che compongono i token: tale misura è direttamente proporzionale ad una metrica da evidenziare. Nel nostro caso più grandi sono i caratteri che compongo la singola

parola, tanto più il token è importante all'interno del topic considerato. Vene riportato ora qualche esempio derivato dal modello LDA basato su BoW tf con un totale di 15 topic.

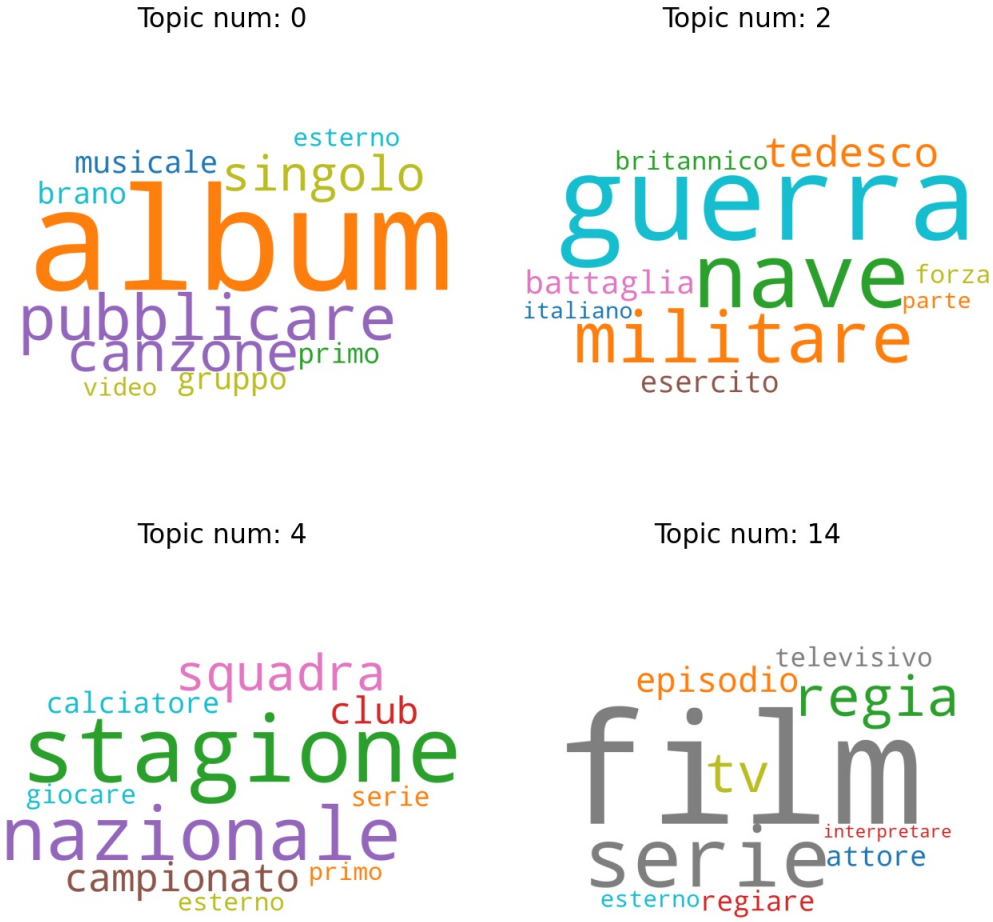

Figura 4.1

Analizzando la word cloud del topic numero 14, sembra chiaro che il tema trattato sia "cinematografia" e tutto ciò che è legato al mondo dei film e delle serie tv. Questa tipologia di rappresentazione è davvero di forte impatto, infatti, non necessita di nessun dato numerico o asse cartesiano per trasmettere ciò che ha da esprimere, è estremamente intuitivo e anche chi non ha mai avuto a che fare con il topic modeling può stabilire l'argomento trattato in quel preciso topic. Possiamo notare delle parole in comune tra

i topic riportati, come la parola “esterno”; questo accade spesso con alcune parole comuni, ma come si può intuire dalla ridotta dimensione dei caratteri che compongono la parola, questa non assume un ruolo principale all’interno dei topic.

4.2 Analisi delle componenti principali

Un documento della nostra collezione, per quanto visto fino ad ora, è descritto dalla sua bag of words. Quest’ultima sappiamo essere un insieme di coppie di valori, dove il primo è un identificatore del token presente nel documento, il secondo valore è la frequenza assoluta oppure il valore tf-idf del termine rispetto al documento. Passando ad un’interpretazione più cartesiana, il token rappresenta un asse dello spazio multidimensionale, e la metrica associata indica il valore che assume il documento nel corrispettivo asse. Se pensiamo invece all’intera collezione questa identifica uno spazio con tante dimensioni quante sono le parole presenti nel proprio vocabolario, nel nostro caso lo spazio generato dal corpus conta poco più di 56.000 dimensioni. Questo numero così elevato rappresenta un ostacolo dal punto di vista grafico, poiché la concezione dell’uomo si ferma alla terza dimensione; quindi, è necessaria una strategia per ridurre queste 56.000 dimensioni ad un massimo di 3. L’analisi delle componenti principali (PCA) è stata pensata proprio per questo tipo di problematiche; Ha l’esatto obiettivo di massimizzare la varianza, calcolando il peso da attribuire ad ogni variabile di partenza per poterle concentrare in una o più nuove variabili (dette componenti principali) che saranno combinazione lineare delle variabili originarie. Dopo aver calcolato le componenti principali, quello che vorremmo ottenere sono un numero limitato di componenti che riescano a spiegare la maggior parte della varianza prodotta dai dati. Le componenti sono disposte in ordine di “importanza”, pertanto la prima componente è quella che cattura la massima variabilità e così via a scalare fino all’ultima che sarà la meno informativa. Ecco che se le prime 3 componenti raccolgono un buon livello di varianza spiegata, sembra sensato poter produrre un grafico a partire da esse.

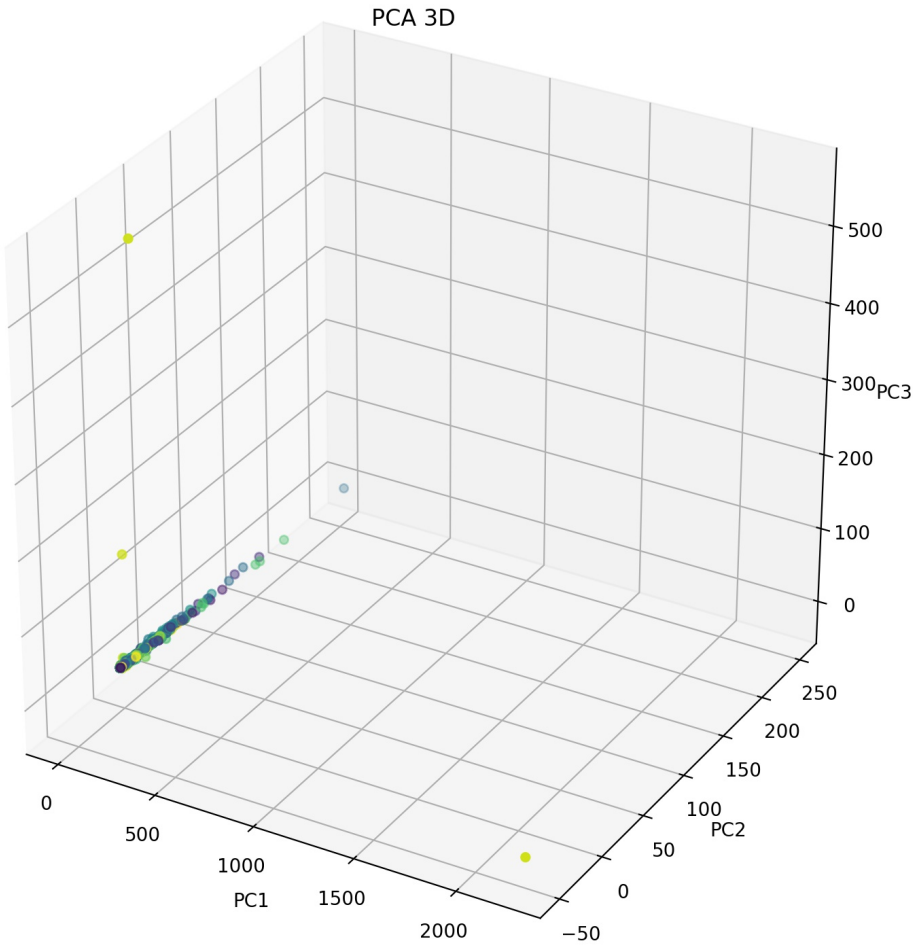

Figura 4.2

Com'è possibile vedere dal grafico riportato, l'output finale non fa capire molto di come si distribuiscono i documenti all'interno dello spazio. L'analisi delle componenti principali è una tecnica di riduzione della dimensionalità lineare e questo rappresenta un limite in questo caso. Nonostante il grafico poco informativo, il lavoro fatto fino ad ora per calcolare le PCA non è vano, potremmo sfruttare questa riduzione della dimensionalità come base per tecniche più sofisticate.

4.3 T-sne

t-SNE (t-Distributed Stochastic Neighbor Embedding) (van der Maaten e Hinton 2008) è una tecnica che visualizza dati ad alta dimensione assegnando a ciascun punto una posizione in una mappa bi o tridimensionale. La tecnica è la variazione di Stochastic Neighbor Embedding (SNE) con l'aggiunta della distribuzione t di student, usata per soppesare le distanze tra i punti. Tecniche come la PCA sono dette lineari perché trattano linearmente la distanza tra i diversi punti; con t-SNE invece, le distanze vengono convertite in probabilità facendo riferimento alla distribuzione t. Questo significa che, prendendo un'osservazione come riferimento, le altre unità vicine al punto considerato avranno un maggior peso, mentre quelle distanti avranno un peso pressoché nullo.

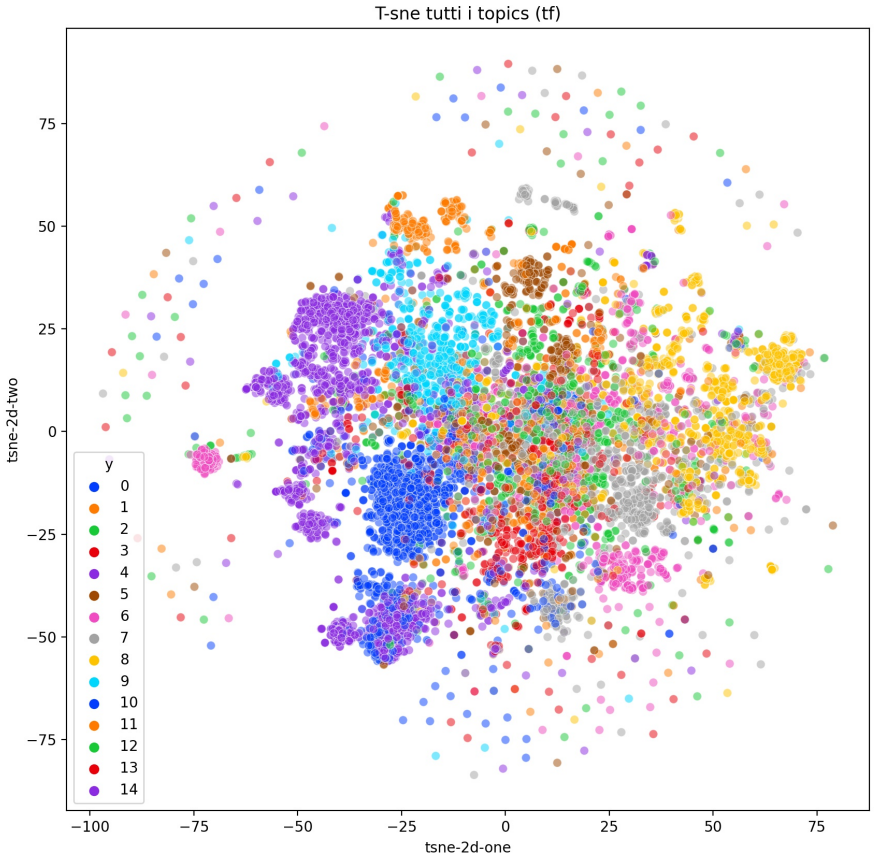

Figura 4.3

Come possiamo vedere dal grafico che rappresenta il modello LDA basato su BoW tf, alcuni gruppi sono ben riconoscibili nella zona centrale, con un risultato migliore rispetto a quello prodotto dalla PCA. Si vuole fare un ragionamento a partire da questo grafico riconducendoci anche a quanto visto nella figura 3.1, dove si evidenziava un'alta coerenza nel topic 0 e 4, mentre valori bassi di coerenza nei topic 11 e 12. Verranno ora riportati i grafici t-SNE relativi ai topic appena citati in modo da evidenziare le differenze.

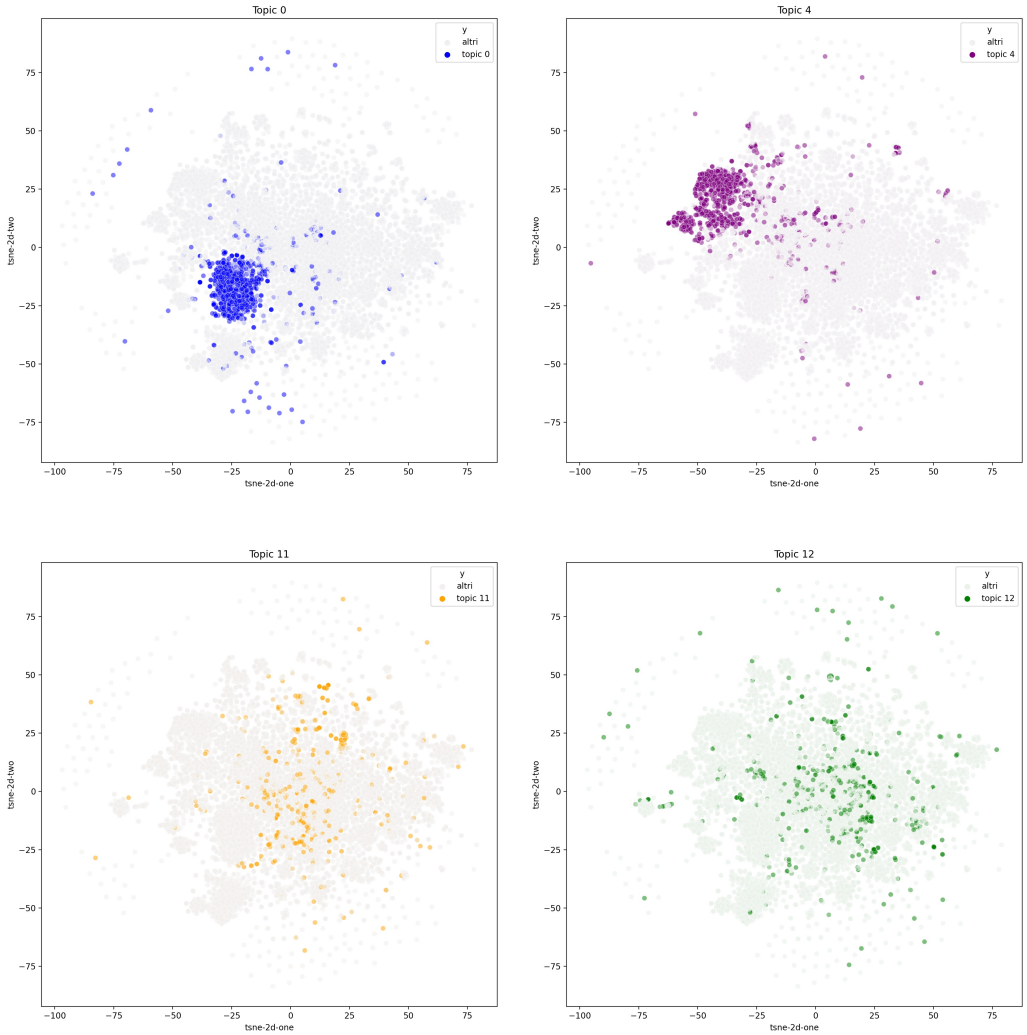

Figura 4.4

Come si può notare dai grafici dei topic 0 e 4, grazie alla loro coherence di circa 0.7, i punti nello spazio risultano molto più concentrati e raggruppati in un'unica zona. Questo fa capire che i documenti appartenenti a questi gruppi trattano un argomento molto specifico e limitato anche come area tematica. Differentemente i topic 11 e 12, come possiamo vedere dalla figura 4.4, coprono un'area più vasta: questo fenomeno non indica necessariamente un errore nella creazione del gruppo, ma potrebbe trattarsi semplicemente di un topic con tema più generale rispetto ai precedenti, meno di nicchia.
Prendiamo ora in considerazione il secondo modello LDA che è stato stimato, ovvero quello basato sulla bag of words tf-idf. Essendo i valori della metrica tf-idf molto più ravvicinati tra loro, poiché subiscono una trasformazione logaritmica, è difficile evidenziare delle differenze significative tra i documenti, anche facendo uso di tecniche come t-SNE.

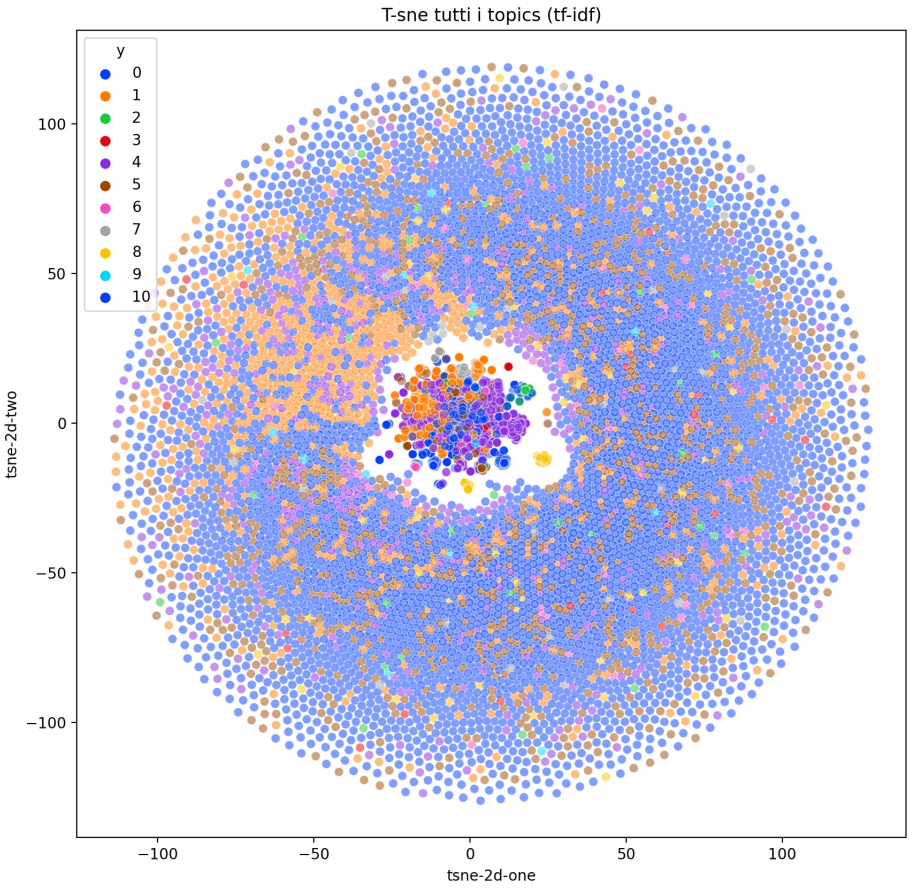

Figura 4.5

Come è possibile vedere dal grafico, non viene localizzato nessun gruppo. Per questo si ricorre ad una combinazione di due tecniche, in modo da unire i vantaggi di entrambe in un unico risultato grafico. Il primo passo consiste nel calcolare le PCA per ottenere delle componenti che, a differenza delle variabili originali, riescano a cogliere maggior varianza essendo così più disomogenee tra loro.
Successivamente viene calcolato t-SNE a partire proprio dalle componenti appena calcolate, producendo il seguente output.

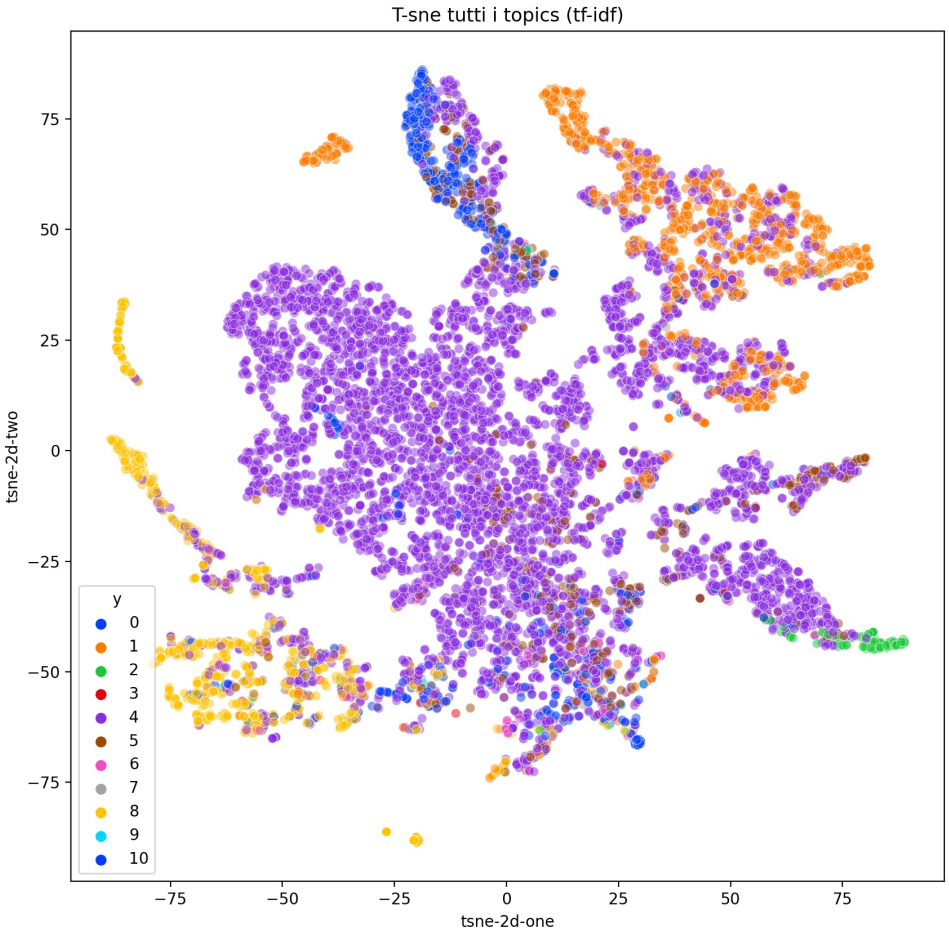

Figura 4.6

Anche in questo caso si vuole far notare, in riferimento a quanto visto in figura 3.2, come in base ai diversi livelli di topic coherence cambi anche la dispersione dei documenti appartenenti ai topic.

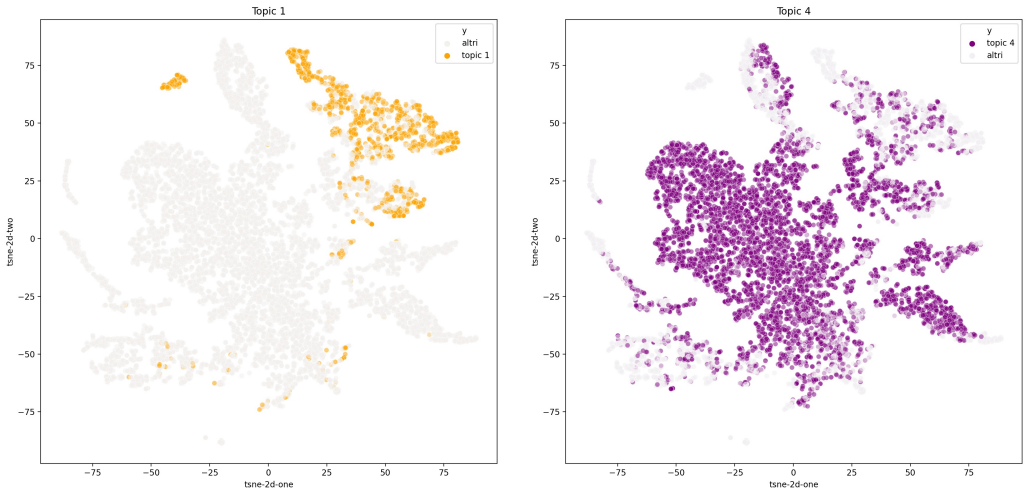

Figura 4.7

Pur essendo consapevoli della differenza di numerosità tra il topic 1 e 4, possiamo comunque confermare quanto detto in precedenza. Il topic 4, con il suo basso livello di topic coherence, è presente in un'area molto vasta, differentemente da quanto accade per il topic 1.

4.4 UMAP

L'algoritmo UMAP (Uniform Manifold Approssimation and Projection) è una tecnica di riduzione della dimensionalità simile a t-SNE, ma a detta degli autori, gode di una maggiore efficienza e velocità di elaborazione . I meccanismi sottostanti a questo algoritmo sono gli stessi di t-SNE ma con una ottimizzazione matematica molto complessa spiegata dettagliatamente nell'articolo dedicato (McInnes et al. 2018). L'utilizzo di questa complessa tecnica in questo elaborato è limitato, ma è utile sapere che UMAP supporta una serie di funzioni, inclusa la possibilità di utilizzare etichette (o etichette parziali) per la riduzione della dimensionalità supervisionata (o semi-supervisionata).
Questa tecnica è stata usata per creare un grafico tridimensionale dove proiettare i documenti del nostro corpus nelle diverse configurazioni (tf e tf-idf).

Considerando il modello LDA con bag of words tf composto da 15 topic, questo è l'output associato.

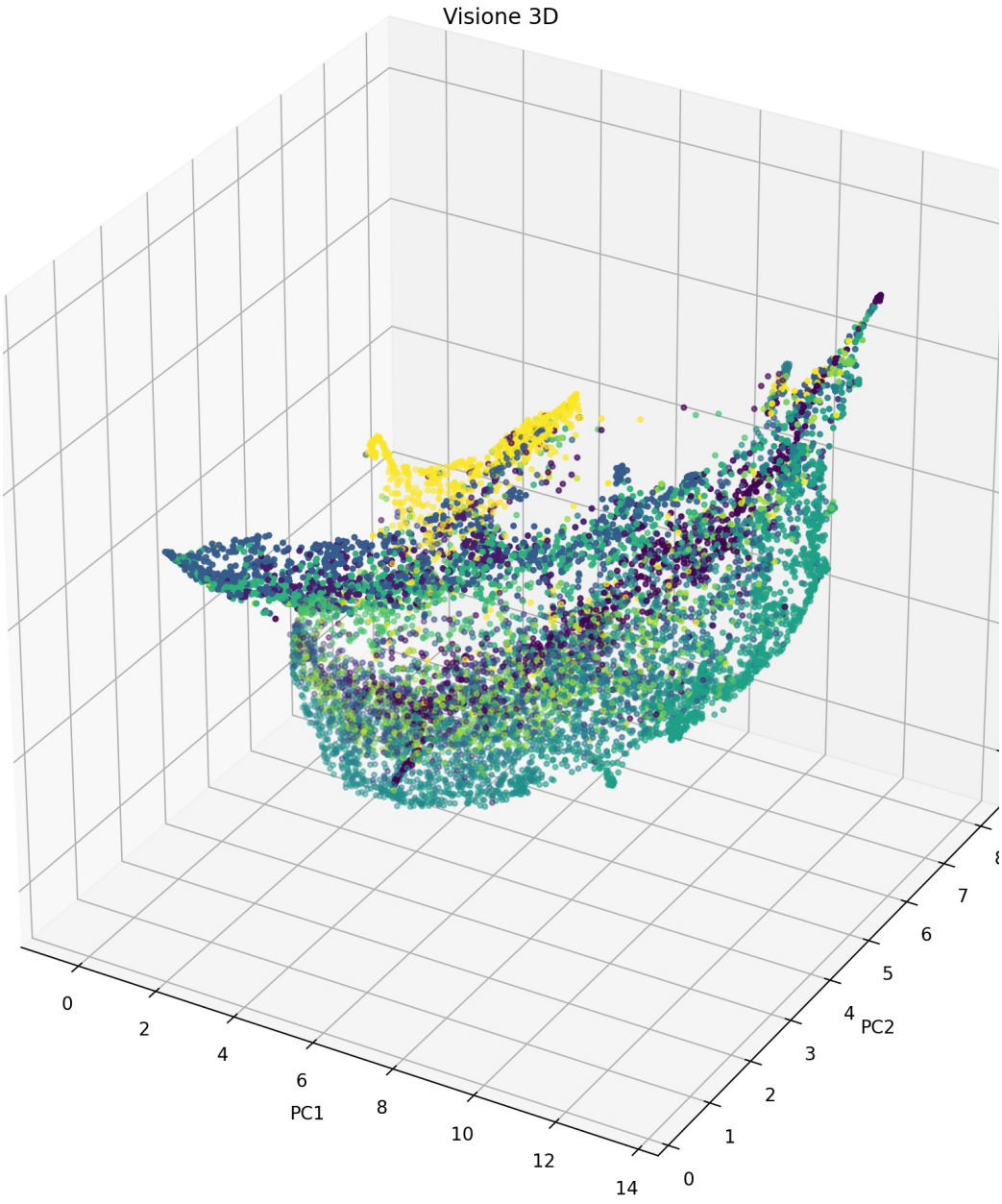

Figura 4.8

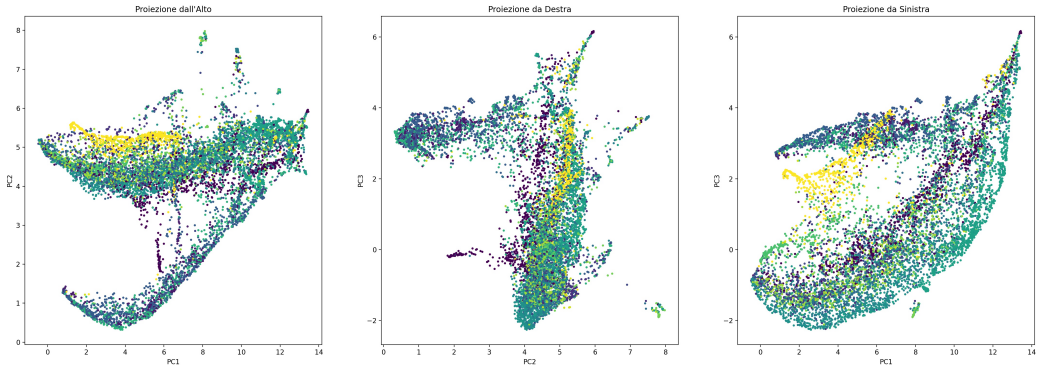

Figura 4.9

Come si può notare, alcuni gruppi sono ben distinguibili dalla massa dei punti, altri invece sembrano più sparsi all'interno dello spazio. Per quanto detto fino ad ora, questa differenza di forma tra i gruppi dovrebbe essere colta dalla metrica topic coherence. Per verificare quanto appena detto, si vuole provare ad isolare dei topic e valutarne l'area in cui si distribuiscono.

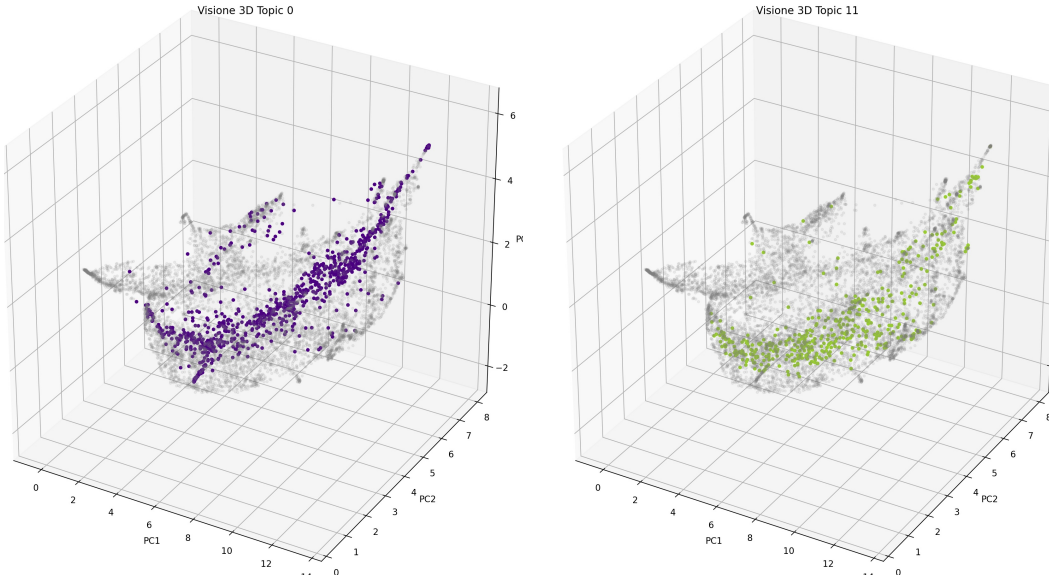

Figura 4.10

Da quanto visto in figura 3.1, i topic 0 e 11 hanno rispettivamente un valore di 0,7 e 0,4 di coherence. Questa differenza la si può cogliere guardano i grafici in 3d dove la massa di punti viola del topic 0 sembra essere concentrati attorno ad una retta, mentre la massa di punti verde sembra essere più libera nello spazio, occupando comunque un'area limitata. Osserviamo ora le proiezioni di questi topic.

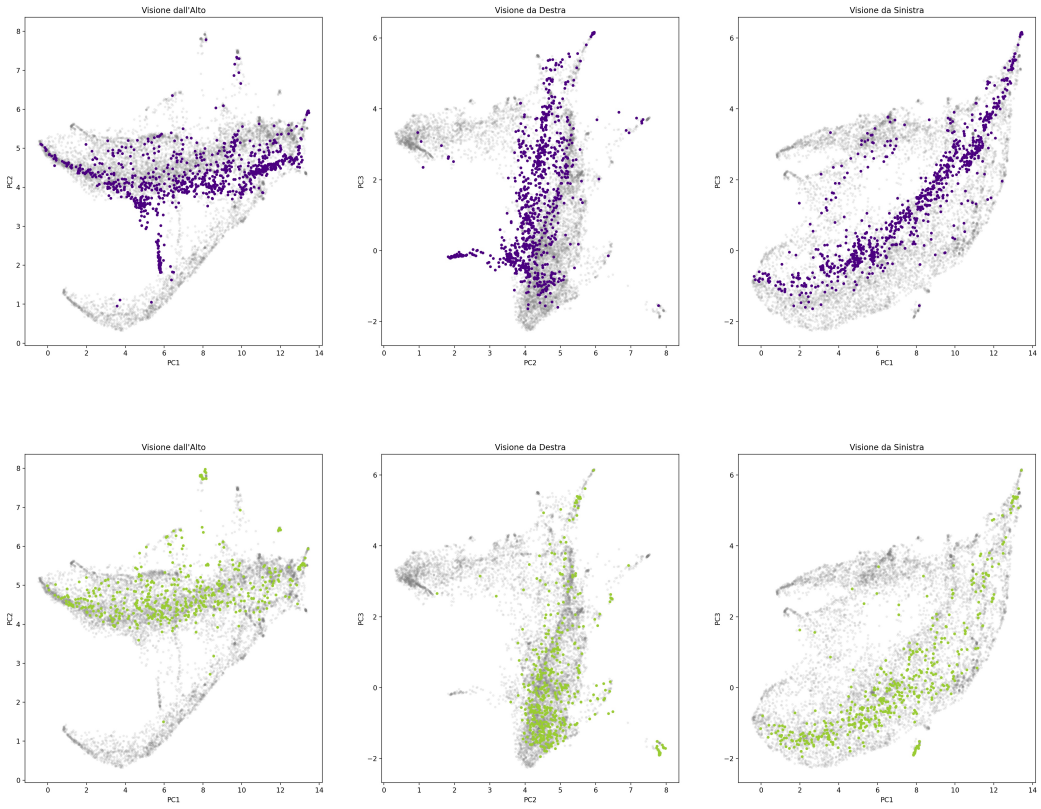

Figura 4.11

Guardando le proiezioni dei topic nei grafici bidimensionali, la differenza non è poi così evidente, se non per la proiezione da sinistra dove il topic 0 mostra uno schema un po' più preciso e ristretto attorno ad una retta.
Passiamo ora all'analisi della seconda configurazione adottata per il modello LDA, ovvero quella basata su bag of words tf-idf. È interessante mostrare

anche questa seconda serie di grafici poiché la forma assunta dalla massa di punti è completamente differente, pur partendo dallo stesso corpus.

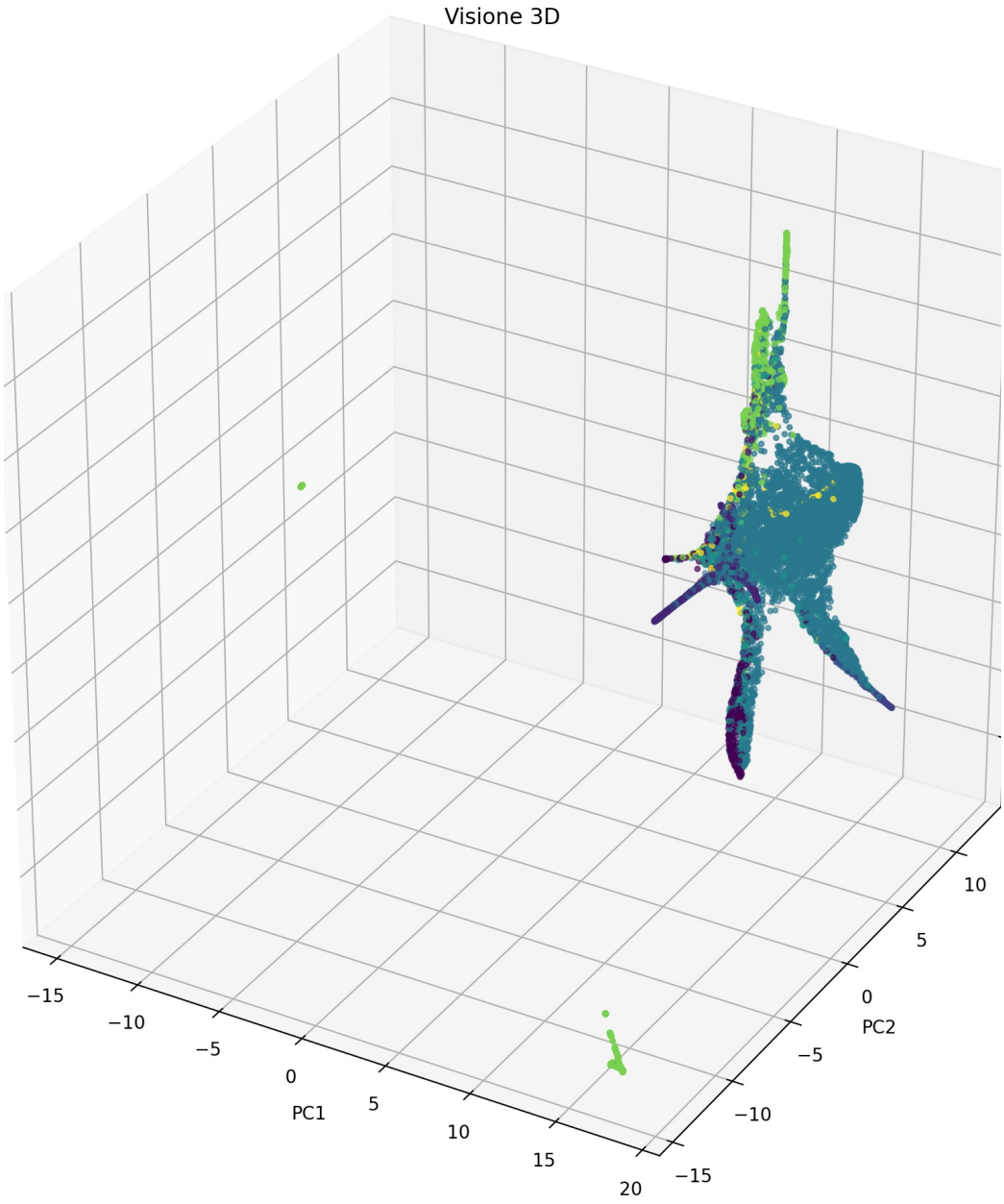

Figura 4.12

Come si può notare, la massa di punti risulta essere molto più concentrata in una piccola zona, ma questo è dovuto solamente ad una diversa scala del grafico nei tre assi. Nonostante questo, il grafico risulta poco informativo al fine di comprendere i gruppi che si formano grazie a questa tecnica. Può essere d'aiuto consultare anche le proiezioni laterali.

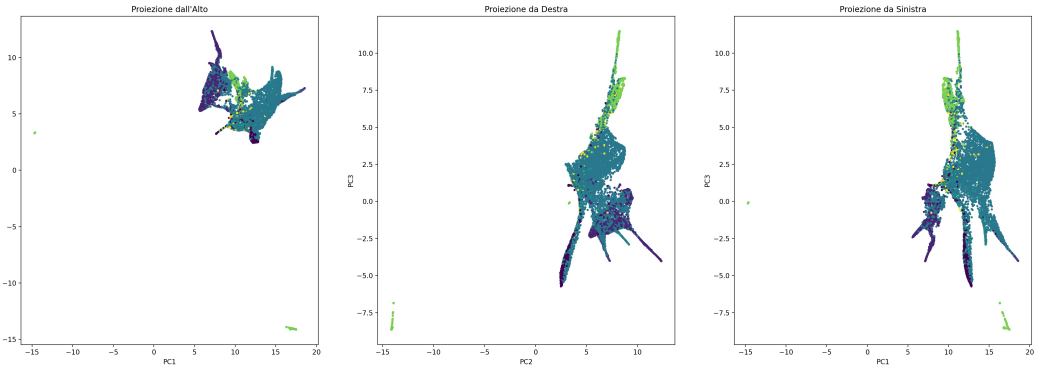

Figura 4.13

Anche le proiezioni laterali non sono particolarmente informative, questo principalmente a causa della grande massa di punti azzurri che copre la maggior parte dell'area. Questo fenomeno era già stato riportato durante l'analisi del t-SNE dove era stato fatto notare la diversa numerosità dei topic (in particolare la massa di punti azzurri corrisponde al topic 4), anche se graficamente t-SNE produceva un output migliore.
In base a quanto detto fino ad ora, l'elevata dispersione del topic 4 è giustificata anche da quanto visto in figura 3.2, ovvero un basso valore di topic coherence (circa 0,3). Proviamo ad isolarlo per visualizzarlo meglio e confrontarlo con un topic ad alta coerenza come il numero 0.

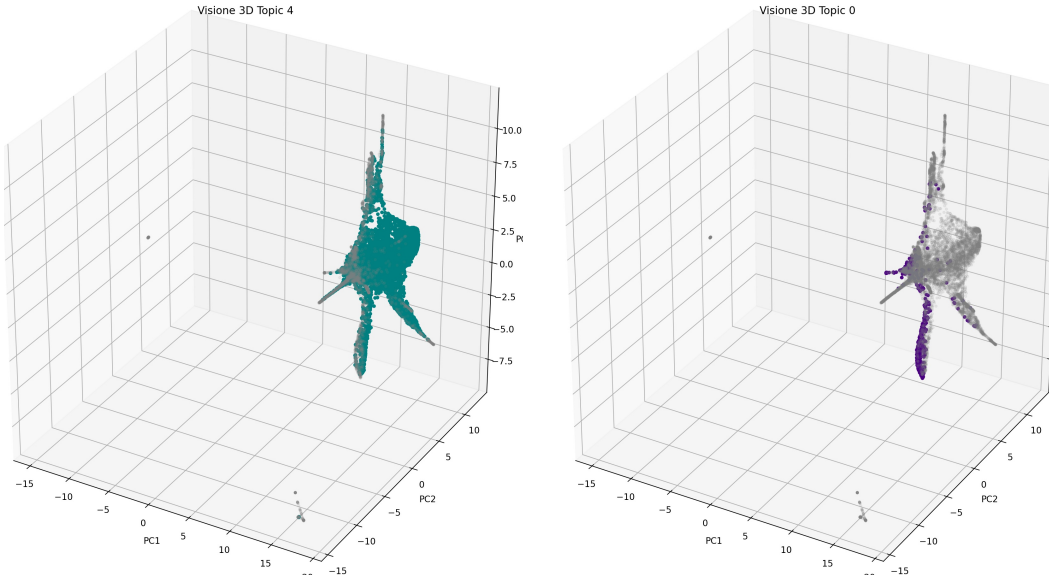

Figura 4.14

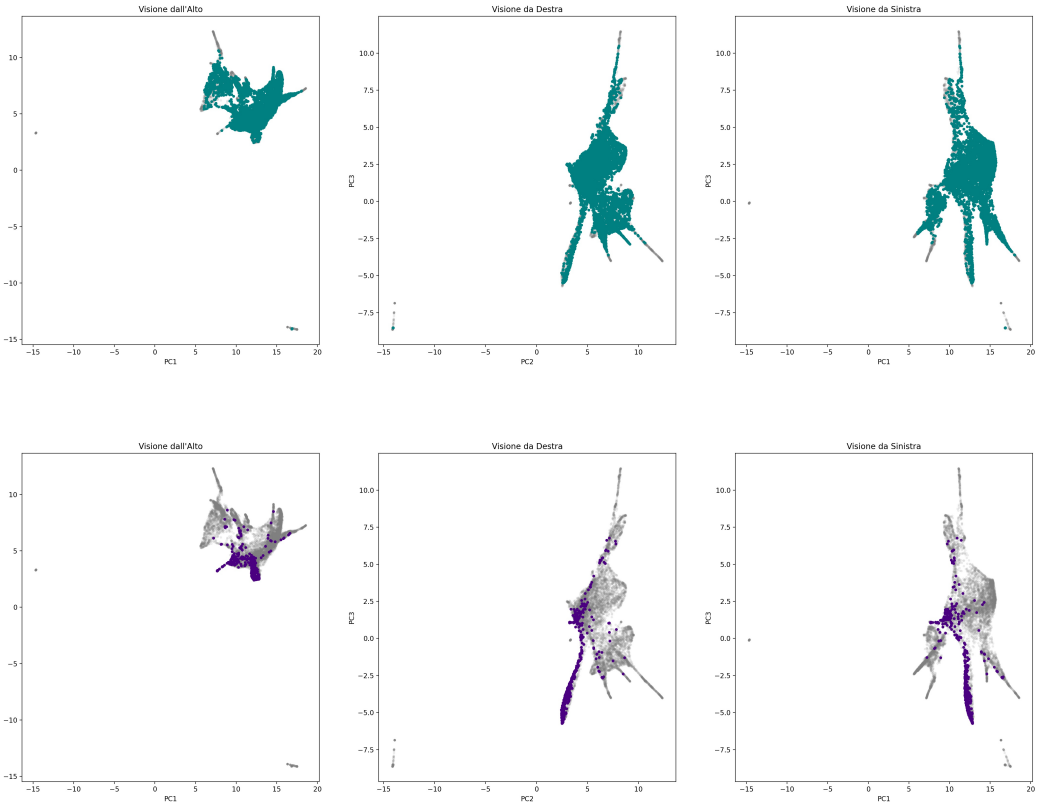

Figura 4.15

In questo caso la differenza risulta essere più marcata, vediamo infatti come il topic 0 (colore viola) si concentri un un'area molto più limitata e con una struttura ben definita. Mentre il topic 4 sembra andare a toccare pressoché tutte le aree tematiche del corpus.

Conclusioni

Come è stato possibile vedere durante il corso dell'elaborato, al giorno d'oggi le problematiche principali, quando si parla di topic modeling, sono due. La prima è legata alla lingua, in quanto questi strumenti vengono sviluppati principalmente per la lingua inglese, ma grazie a molte librerie a disposizione, è possibile sfruttare queste tecniche di NLP con ottimi risultati nella maggior parte delle lingue, nel nostro caso l'italiano. Il secondo problema da affrontare è il limite computazionale imposto dalle macchine sulle quali lavoriamo. Infatti, come spiegato nei primi capitoli, anche per progetti di questo tipo, spesso si usano collezioni limitate di documenti per non dover attendere ore, o addirittura giorni, di elaborazione.
Passando alla creazione dei modelli, è stato mostrato il Latent dirichlet allocation che rappresenta un compromesso tra complessità ed efficienza computazionale. Inoltre, come è stato mostrato le possibili variazioni per personalizzare questo tipo di modello sono tante, dalla scelta di come configurare i documenti in ingresso (nel nostro caso due differenti bag of words), al numero di topic desiderati.
Concludendo con la sezione assegnata alla visualizzazione, è possibile notare come queste tecniche dedicate ai risultati grafici siano ancor'oggi particolarmente complesse. Si tratta infatti di dover confrontare più tecniche per trovare quella che meglio riesce a descrivere il nostro corpus nello spazio. Non esistono infatti tecniche consolidate che forniscano un riferimento in questo contesto, tecniche di riduzione della dimensionalità classiche come la PCA falliscono in caso di dataset non lineari, tecniche più complesse come t-SNE e UMAP invece, portano a buoni risultati grafici, ma questo non è sempre garantito come mostrato dagli esempi.

Bibliografia

Honnibal, Matthew e Ines Montani (2017). «spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing». To appear.

Loper, Edward e Steven Bird (2002). «NLTK: The Natural Language Toolkit». In: *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*.

McInnes, Leland et al. (2018). «UMAP: Uniform Manifold Approximation and Projection». In: *Journal of Open Source Software* 3.29, p. 861. DOI: 10.21105/joss.00861. URL: https://doi.org/10.21105/joss.00861.

Rehurek, Radim e Petr Sojka (2011). «Gensim–python framework for vector space modelling». In: *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic* 3.2.

Röder, Michael, Andreas Both e Alexander Hinneburg (2015). «Exploring the Space of Topic Coherence Measures». In: *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. WSDM '15. Shanghai, China: Association for Computing Machinery, 399–408. ISBN: 9781450333177. DOI: 10.1145/2684822.2685324. URL: https://doi.org/10.1145/2684822.2685324.

van der Maaten, L.J.P. e G.E. Hinton (2008). «Visualizing High-Dimensional Data Using t-SNE». English. In: *Journal of Machine Learning Research* 9.nov. Pagination: 27, pp. 2579–2605. ISSN: 1532-4435.