

Information Retrieval

Giuseppe Magazzù

2021 - 2022

Contents

1	Definitions	1
1.1	Document	1
1.2	Terms	1
1.3	Stop Words	1
2	Text Processing	2
2.1	Tokenization	2
2.2	Normalization	3
2.3	Stop Words Removal	3
3	Text Representation	4
3.1	Bag Of Words	4
3.2	Zipf's Law	5
3.3	Luhn's Analysis	5
4	Text Enrichment	7
4.1	Part-of-Speech (POS) tagging	7
4.2	Named Entity Recognition (NER)	8
5	Statistical Language Models	10
5.1	Language Model	10
6	Word Embedding	12
6.1	How to Represent Word Vectors?	12
6.2	Count-based models	14
6.2.1	SVD	15
6.2.2	GloVe	15
6.3	Predictive Models	16
7	Information Retrieval	18
7.1	Data Structures	20
7.1.1	Inverted File Structure	21
7.1.2	Posting File Optimization	21
7.1.3	Dictionary Organization	22
7.1.4	Altre Strutture di Indicizzazione	22
7.2	Models	22
7.2.1	Boolean Model	23

Chapter 1

Definitions

1.1 Document

Un **documento** è solitamente formato da un testo, una struttura, altri media (immagini, suoni, ...) e da dei metadata.

Per **testo** si intende una sequenza di stringhe di caratteri di un alfabeto.
E.g. le sequenze del genoma, formule chimiche e parole del linguaggio naturale.

Un documento può essere composto da

- structured data (tabelle, database, ...)
- semi-structured data (html, xml, ...)

I **metadata** sono dati esterni riguardo al documento. Possono essere classificati in due categorie:

- **metadata descrittivi**: riguardano la creazione del documento (e.g. titolo, autore, data, ...)
- **metadata semantici**: descrivono informazioni contestualmente rilevanti o specifiche del dominio (e.g. ontologie)

1.2 Terms

I termini sono dei descrittori che vengono associati al testo.

1.3 Stop Words

I termini che non sono significativi per la rappresentazione del testo (particelle, articoli, ...).

Chapter 2

Text Processing

Il text processing è una fase necessaria per preparare e pulire il testo.

2.1 Tokenization

La tokenization consiste nell'identificare e separare all'interno di un testo delle unità chiamate token. I token possono essere parole, frasi, simboli o n-grammi. Ogni token è un candidato a essere un termine significativo (index).

e.g. "Text mining is to identify useful information"

Tokens: "Text", "mining", "is", "to", "identify", "useful", "information"

Problemi:

- parole composte ("Hewlett-Packard" → "Hewlett", "Packard")
- numeri, date ("Mar. 12, 1991", "12/3/1991", "(800) 234-2333")
- problemi linguistici (parole composte, assenza di spazi, ...)

I token possono essere raggruppati in sequenze contigue di N elementi chiamate N-grammi.

e.g. "Corpus is the collection of text documents."

Bigrammi: "Corpus is", "is the", "the collection", "collection of", "of text", "text documents", "documents ."

La tokenization si può effettuare tramite espressioni regolari o metodi statistici.

2.2 Normalization

Ad una parola possono essere associati diversi token. La normalizzazione consiste nell'ottenere le classi di equivalenza dei token rimuovendo punti, trattini, accenti.

U.S.A. \Leftrightarrow USA
anti-aliasing \Leftrightarrow antialiasing
résumé \Leftrightarrow resume
15/10/2021 \Leftrightarrow 15 Ott 2021

Lemmatization

Le parole vengono ridotte alla loro forma base (lemma) tenendo in considerazione l'intero vocabolario della lingua e analizzando la parte del discorso.

e.g. "ladies" \Rightarrow "lady", "forgotten" \Rightarrow "forgot"

Stemming

Le parole vengono ridotte a una radice (stem) rimuovendo le flessioni tramite l'eliminazione dei caratteri non necessari.

e.g. "automate(s)", "automation", "automatic" \Rightarrow "automat"

Case folding

Tutte le parole vengono convertite in lowercase a parte alcune eccezioni.

Thesaurus and Soundex

Un thesaurus (tesauro) è una risorsa linguistica generata manualmente da essere umani in cui è possibile esprimere relazioni tra parole (e.g. gerarchie, sinonimi, ...).

Soundex è un algoritmo fonetico che permette di rappresentare correttamente diverse parole omofone nonostante differenze di ortografia usando delle euristiche fonetiche.

2.3 Stop Words Removal

Le **stop words** sono le parole più frequenti all'interno di un testo che possono essere rimosse senza perdere il significato. Queste parole essendo presenti in più documenti non portano informazioni utili per distinguerli.

Esistono delle liste di **stop words** definite in base alla lingua che possono essere usate per la rimozione.

I web search engine non effettuano la rimozione delle **stop words** perché sono necessarie per alcune ricerche.

Chapter 3

Text Representation

In un sistema di information retrieval i documenti devono essere rappresentati in un formato interno e ordinati per essere indicizzati.

3.1 Bag Of Words

Un modo semplice per rappresentare un testo è una matrice in cui sulle righe ci sono termini estratti dal corpus (vocabolario) e sulle colonne i documenti.

La **Bag Of Words (BOW)** è una rappresentazione del testo che descrive le occorrenze di parole in un documento.

Incidence Matrix: specifica la presenza di un termine in un ogni documento.

Ogni documento può essere rappresentato da un insieme di termini o da un vettore binario.

	Doc1	Doc2	Doc3	Doc4	
Term1	1	1	1	0	Rappresentazione di Doc1
Term2	0	1	1	1	$R1 = \{\text{Term1}, \text{Term2}, \text{Term3}\}$
Term3	0	0	1	0	$R1 = \langle 1, 0, 0 \rangle$

Count Matrix: specifica la numero di occorrenze di un termine in ogni documento.

Un documento viene rappresentato da un vettore di occorrenze.

	Doc1	Doc2	Doc3	Doc4	
Term1	57	57	71	133	Rappresentazione di Doc1
Term2	4	34	17	92	$R1 = \langle 157, 4, 232 \rangle$
Term3	232	2	10	293	

Le rappresentazioni vettoriali non considerano l'ordine delle parole nel testo.

Bag Of Words con N-grammi

Pro: cattura le dipendenze locali e l'ordine

Contro: incrementa la frequenza delle parole

3.2 Zipf's Law

Descrive la frequenza di un evento (parola) in un insieme in base al suo rank.

rank: posizione di un termine nell'ordine decrescente di frequenza dei termini in tutta la collezione.

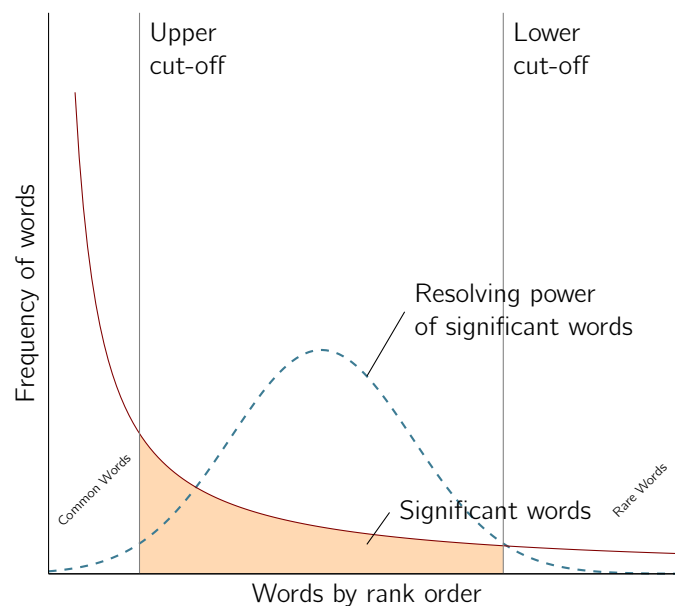
La frequenza di una parola w , $f(w)$ è proporzionale a $1/r(w)$.

$$f \propto \frac{1}{r} \Rightarrow f \cdot r = k \text{ (costante)}$$

$$P_r = \frac{f}{N} = \frac{A}{r} \quad \text{probabilità del termine di rank } r, \quad A = \frac{k}{N} \approx 0.1$$

3.3 Luhn's Analysis

Generalmente termini con frequenza molto alta e molto bassa sono inutili per discriminare i documenti.



L'abilità delle parole di discriminare il contenuto di un documento è massimo nella posizione tra i due livelli di cut-off.

Vogliamo assegnare dei pesi ai termini tenendo conto di questi due fattori:

- corpus-wide: alcuni termini portano più informazione riguardo al documento
- document-wide: non tutti i termini sono ugualmente importanti

Andiamo a definire due frequenze:

- Term Frequency (TF): frequenza di un termine all'interno di un documento
- Inverse Document Frequency (IDF): frequenza di un termine in tutta la collezione

Il peso di un termine deve essere proporzionale a TF e inversamente proporzionale a IDF.

La **term frequency** $tf_{t,d}$ è il numero di occorrenze del termine t nel documento d diviso il numero totale di termini nel documento.

$$tf_{t,d} = \frac{f_{t,d}}{\sum_{t_i} f_{t_i,d}}$$

Questa misura può essere normalizzata dividendo per la frequenza massima nel documento d per essere confrontabile tra documenti diversi.

$$ntf_{t,d} = \frac{f_{t,d}}{\max_{t_i} f_{t_i,d}}$$

La **inverse document frequency** idf_t è la frazione inversa della frequenza di un termine in un documento in scala logaritmica.

$$idf_t = \log(N/df_t), \quad df_t \leq N$$

dove df_t è la **document frequency**, ovvero il numero di documenti che contengono il termine t , e N il numero totale di documenti.

Infine possiamo calcolare la funzione TF-IDF come prodotto di TF e IDF.

$$tf-idf_{t,d} = (ntf_{t,d} / \max_{t_i} ntf_{t_i,d}) * \log(N/df_t)$$

Questa funzione rappresenta il peso del termine t all'interno di un documento d .

- termine comune in un documento \rightarrow high tf \rightarrow high weight
- termine raro nella collezione \rightarrow high idf \rightarrow high weight

Chapter 4

Text Enrichment

Il text enrichment permette di avere informazioni aggiuntive sul testo che possono aiutare a fare delle analisi più approfondite e predizioni più precise.

Due tecniche che ci permettono di arricchire il testo sono il POS tagging e la Named Entity Recognition. Entrambe le tecniche possono essere applicate usando degli insiemi di regole o degli approcci di apprendimento supervisionato.

4.1 Part-of-Speech (POS) tagging

Il POS tagging è il processo che marca ogni termine nel documento con un tag che corrisponde a una part-of-speech.

$\frac{A}{DET} + \frac{dog}{NOUN} + \frac{is}{AUX} + \frac{chasing}{VERB} + \frac{a}{DET} + \frac{boy}{NOUN} + \frac{in}{PREP} + \frac{the}{DET} + \frac{park}{NOUN}$

Word Classes

Le classi di parole hanno le seguenti caratteristiche:

- hanno trasformazioni simili
- hanno le stesse funzioni nella frase
- appaiono in un contesti simili

Le word class possono essere chiamate anche part-of-speech, lexical categories, morphological classes, lexical tags o POS.

Per effettuare il POS tagging è necessario scegliere un insieme di tag da associare alle parole. Comunemente vengono identificate 9 classi di POS: noun, verb, adjective, adverb, preposition, article, interjection e conjunction. Comunque è possibile usare più classi e sottoclassi.

Tag Ambiguity

Spesso una parola può essere associata a più di un POS, quindi è necessario considerare il contesto.

- | | |
|------------------------------------|-----------------|
| • The back door | (adjective) |
| • Promised to back the bill | (verb) |
| • Win the voters back | (adverb) |
| • On my back | (singular noun) |

E' possibile disambiguare le parole considerando il contesto. Spesso dei tag co-occorrono con altri tag (e.g. gli articoli e i sostantivi).

Rule Based Tagging

Si assegna ad ogni parola tutti i possibili tag usando un dizionario. Tramite delle regole scritte a mano, vengono rimossi alcuni tag finché non ne rimane uno solo.

4.2 Named Entity Recognition (NER)

Il Named Entity Recognition (NER) consiste nel trovare e classificare entità in un testo (persone, date, luoghi, organizzazioni).

Il NER necessita di definire delle categorie di interesse. Tre categorie universalmente accettate sono: persone, luoghi e organizzazioni. Altri task possono richiedere il riconoscimento di data/ora, espressioni, misure e in alcuni casi è utile usare entità specifiche del dominio (farmaci, navi, elementi chimici, ...).

IL NER risulta difficile nei casi di **metonimia**, ovvero la sostituzione di un termine con un'altro che ha una certa relazione con la prima e mantiene in un certo modo il significato.

- “mi piace leggere Dante” / le opere di Dante (scambia l'autore per l'opera)
- “confidare nell'amicizia” / negli amici (scambia l'astratto per il concreto)
- “ha una buona penna” / sa scrivere bene (scambia la causa per l'effetto)

Rule Based

Si usa una lista di parole e frasi che categorizzano le entità e delle regole per verificare o trovare nuove entità. Le regole e le categorie dipendono dal linguaggio.

- “<number> <word> street” per gli indirizzi
- “<street address>, <city>” o “in <city>” per i nome di città
- “<title> <name>” per trovare nuovi nomi

Statistical Machine Learning

Le entità vengono trovate stimando la probabilità che appaia insieme ad altre parole. (e.g. “marathon” è un luogo o evento sportivo, mentre “boston marathon” è uno specifico evento sportivo.) Le probabilità vengono ottenute usando un training set etichettato.

Un possibile approccio sono le Hidden Markov Model. Questo modello si basa sulla proprietà Markoviana, per cui la prossima parola in una sequenza dipende solo dalla precedente. L'entità viene riconosciuta dalla sequenza di parole con la probabilità più alta.

Chapter 5

Statistical Language Models

Un statistical LM ci permette di rappresentare un testo mantenendo degli aspetti semantici, tramite una distribuzione di probabilità su sequenze di parole.

Applicazioni:

- Machine Translation, $P(\text{high winds tonite}) > P(\text{large winds tonite})$
- Spell Correction, $P(\text{about 15 minutes}) > P(\text{about 15 minuets})$
- Speech Recognition, $P(\text{I saw a van}) > P(\text{eyes awe of an})$

5.1 Language Model

L'obiettivo di un Language Model è quello di calcolare la probabilità di una sequenza di parole $P(W) = P(w_1, w_2, \dots, w_n)$.

Con una Language Model è possibile calcolare anche la probabilità di una parola data una sequenza $P(w_5 | w_1, w_2, w_3, w_4)$.

Possiamo calcolare la probabilità della sequenza W con la chain rule:

$$P(w_1, w_2, \dots, w_n) \approx \prod_{i=1}^n P(w_i | w_1, \dots, w_{i-1})$$

Con la Markov Assumption possiamo ridurre il numero di parole da condizionare

$$P(w_1, w_2, \dots, w_n) \approx \prod_{i=1}^n P(w_i | w_{i-k}, \dots, w_{i-1})$$

Un Language Model è ben formato su un alfabeto Ω se $\sum_{s \in \Omega} P(s) = 1$.

N-grams Language Model

La probabilità di una parola di una sequenza dipende dalle N parole precedenti. Nel caso di uni-grammi la probabilità non dipende da nessuna altra parola.

Sparsity Problems

1. La parola di cui vogliamo calcolare la probabilità non è presente nel corpus.
Soluzione: aggiungere una δ alla frequenza di ogni parola (smoothing).
2. La sequenza di cui vogliamo calcolare la probabilità non è presente nel corpus.
Soluzione: ridurre la sequenza da condizionare.

I modelli con uni-grammi sono i più usati

- Spesso sono sufficienti per valutare l'argomento
- Con N più grandi ci sono più problemi di sparsity
- Implementazione semplice ed efficiente

Perplexity

Chapter 6

Word Embedding

Il termine **word embedding** indica una famiglia di tecniche che rappresentano le parole come vettori di numeri reali.

La rappresentazione può essere ottenuta in due modi:

1. Modelli count-based
2. Modelli predittivi

6.1 How to Represent Word Vectors?

Local Representation

Ogni termine del vocabolario T è rappresentato da un vettore binario (one-hot) di dimensione $|T|$, in cui una sola componente ha valore 1 e tutte le altre 0. Ogni elemento del vettore corrisponde a una parola diversa.

	King	Woman	Princess	
King	1	0	0	King = [1, 0, 0]
Woman	0	1	0	Woman = [0, 1, 0]
Princess	0	0	1	Princess = [0, 0, 1]

Problemi:

- Il numero di dimensioni incrementa linearmente con il numero di parole del vocabolario.
- La matrice è molto sparsa.
- Non c'è informazione condivisa tra le parole

Distributed Representation

Ogni termine del vocabolario T è rappresentato da un vettore di k valori reali. Le dimensioni del vettore possono essere osservate (e.g. hand-crafted features) o latenti (e.g. embedding dimensions).

Il numero di dimensioni k vogliamo sia molto piccolo rispetto alla dimensione del vocabolario $|C| = k \ll |T|$.

Questa rappresentazione distribuita ci permette di raggruppare parole simili in base al contesto considerato.

	Femininity	Youth	Royalty	
Man	0	0	0	King = [0.0, 0.0, 1.0]
Woman	1	0	0	Woman = [1.0, 0.0, 0.0]
King	0	0	1	Princess = [1.0, 1.0, 1.0]
Princess	1	1	1	Child = [0.5, 1.0, 0.0]
Child	0.5	1	0	

Parole simili hanno vettori simili quindi possiamo eseguire delle combinazioni lineari dei vettori per scoprire le relazioni tra le parole.

$$\begin{aligned} \text{King} - \text{Man} + \text{Woman} &= \text{Queen} \\ [0, 0, 1] - [0, 0, 0] + [1, 0, 0] &= [1, 0, 1] \end{aligned}$$

Vantaggi:

- La matrice è molto meno sparsa.
- Vengono catturate e mantenute le relazioni tra parole.

Co-occurrence Matrix

Un'altra possibilità è quella di considerare un certo contesto limitando le occorrenze a un certo numero di termini vicini. Il contesto può essere espresso a diverse granularità: documenti, frasi, n-grammi.

Definiamo una matrice chiamata **window-based co-occurrence matrix** (oppure term-context matrix) che contiene il numero di volte che ogni *context word* co-occorre in una finestra di una specifica dimensione con una *target word*.

La finestra di dimensione k contiene le k parole a sinistra e le k a destra della parola in considerazione.

Data la seguente collezione di documenti. Prendiamo in considerazione come target words *magazine* e *newspaper*. Individuiamo le parole nella finestra di dimensione due per ogni target word.

- | | |
|--|--|
| <ul style="list-style-type: none"> • He is reading a magazine • This magazine published my story • She buys a magazine every month | <ul style="list-style-type: none"> • I was reading a newspaper • The newspaper published an article • He buys this newspaper every day |
|--|--|

Costruiamo la matrice di co-occorrenza window-based

	reading	a	this	published	my	buys	the	an	every	month	day
magazine	1	2	1	1	1	1	0	0	1	1	0
newspaper	1	1	1	1	0	1	1	1	1	0	1

Per poter rappresentare ogni termine del vocabolario V la matrice di co-occorrenza deve essere calcolata su tutti i termini. Dal momento che ogni termine comparirà almeno una volta in qualche finestra anche le context word saranno tutti i termini. Quindi la matrice sarà composta da $|V| \times |V|$ elementi.

Questa matrice contiene vettori che sono semanticamente legati, ma non sono densi. E' necessario ridurre il numero di parole utili a rappresentare le varie parole.

PointWise Mutual Information (PMI)

Come già osservato con l'analisi di Luhn le frequenze assolute non sono una buona rappresentazione. Utilizziamo quindi una tecnica che assegna dei pesi a ogni coppia di parole term e context $PMI(w_i, c_j)$.

$$PMI(w_i, c_j) = \log_2 \left(\frac{P(w_i, c_j)}{P(w_i)P(c_j)} \right)$$

$$PPMI(w_i, c_j) = \max \left(\log_2 \left(\frac{P(w_i, c_j)}{P(w_i)P(c_j)} \right), 0 \right)$$

Sia f_{ij} il numero di volte che le parole w_i e c_j co-occorrono.

$$P(w_i, c_j) = \frac{f_{ij}}{\sum_i^W \sum_j^C f_{ij}} \quad P(w_i) = \frac{\sum_j^C f_{ij}}{\sum_i^W \sum_j^C f_{ij}} \quad P(c_j) = \frac{\sum_i^W f_{ij}}{\sum_i^W \sum_j^C f_{ij}}$$

6.2 Count-based models

Si calcolano delle statistiche su quanto spesso una parola co-occorre con le parole vicine in una grande collezione. Queste statistiche vengono mappate in un vettore denso di piccole dimensioni.

I vettori vengono imparati eseguendo una riduzione della dimensionalità della matrice term-context. La matrice viene fattorizzata a una matrice term-feature di dimensione ridotta in cui ogni parola è rappresentata da un vettore denso.

Alcuni modelli:

- Latent Dirichlet Allocation (LDA)
- Singular Value Decomposition (SVD)
- Global Vectors (GloVe)

6.2.1 SVD

Una qualsiasi matrice rettangolare W di dimensione $w \times c$ può essere espressa come prodotto di tre matrici $W = U \times S \times V^T$.

- U matrice $w \times m$ dove le righe w corrispondono alle righe di W e le colonne m rappresentano una dimensione (feature) in un nuovo **spazio latente**.
- S matrice $m \times m$ di singular values che esprimono l'**importanza** di ogni dimensione (feature).
- V^T matrice $m \times c$ dove le colonne c corrispondono alle colonne della matrice W e le righe m corrispondono ai singular values.

Possiamo ottenere un'approssimazione di dimensione minore tenendo soltanto k dei m singular values. La matrice U troncata contiene le k feature più importanti associate a tutte le parole presenti in W .

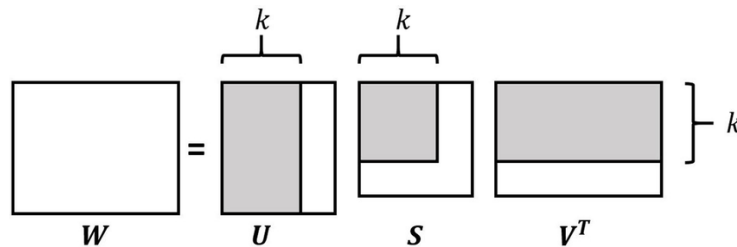


Figure 6.1: Troncamento della SVD [1].

Problemi:

- Se vogliamo aggiungere nuove parole dobbiamo ricalcolare la fattorizzazione perché cambia la dimensione della matrice.
- La matrice è comunque molto sparsa perché la maggior parte delle parole non co-occorre.
- Costo quadratico per il calcolo della fattorizzata.

Una soluzione allo sbilanciamento delle frequenze può essere quella di ignorare le stop words, oppure pesare la co-occorrenza in base alla distanza delle parole nel documento.

6.2.2 GloVe

Sia X la matrice di term-context. Indichiamo con X_{ij} la frequenza di una parola j che occorre nel contesto della parola i , e X_i la frequenza di una qualsiasi parola di apparire nel contesto della parola i .

Definiamo la probabilità di co-occorrenza P_{ij} , ovvero la probabilità che la parola j occorra nel contesto della parola i .

$$P_{ij} = P(j|i) = \frac{X_{ij}}{X_i}, \quad X_i = \sum_k X_{ik}$$

...

valutiamo il rapporto $\frac{P_{ik}}{P_{jk}}$

GloVe costruisce una funzione F che permette di imparare la rappresentazione.

$$F(w_i, w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

Minimizziamo una funzione obiettivo J .

$$J = \sum_{i=1, k=1}^V f(X_{ik})(w_i^T \tilde{w}_k + b_i + \tilde{b}_k - \log(X_{ik}))^2$$

6.3 Predictive Models

I modelli predittivi provano direttamente a predire una parola dai termini vicini.

Il modello più usato è il Word2Vec, una rete neurale poco profonda con un hidden layer che permette di ricostruire il contesto di una parola. Il raw text viene usato come training set di un approccio di apprendimento supervisionato.

Gli esempi di training vengono generati da coppie (target, context) scorrendo una finestra di dimensione fissa su tutta la collezione.

Sia gli input che gli output sono codificati usando la rappresentazione one-hot.

Dopo avere eseguito l'addestramento la matrice dei pesi W conterrà i word embedding del vocabolario V .

Il word embedding della parola V_i sarà rappresentato dal vettore di N elementi $W_i = (w_{(i,1)}, w_{(i,2)}, \dots, w_{(i,N)})$.

Word2Vec può produrre il word embedding utilizzando due architetture:

- **CBOW**

Predice la parola corrente (target word) dalle vicine (context word).

Molto più veloce del modello Skip-gram nella fase di addestramento e ha un'accuratezza leggermente migliore nelle parole frequenti.

- **Skip-gram**

Predice le parole vicine (context word) dalla parola corrente (target word).

Funziona bene con training set piccoli e rappresenta bene le parole rare.

Metodi di training:

- Hierarchical Softmax
- Negative sampling

Grazie alla rete neurale è possibile catturare dei pattern complessi oltre la similarità tra le parole.

Più il training set è grande ($> 10M$ parole) più le performance saranno migliori.

I testi devono includere più parole diverse possibili.

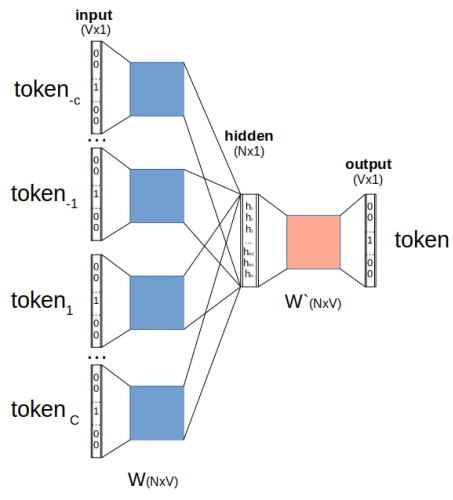


Figure 6.2: Schema della rete con architettura CBOW [2].

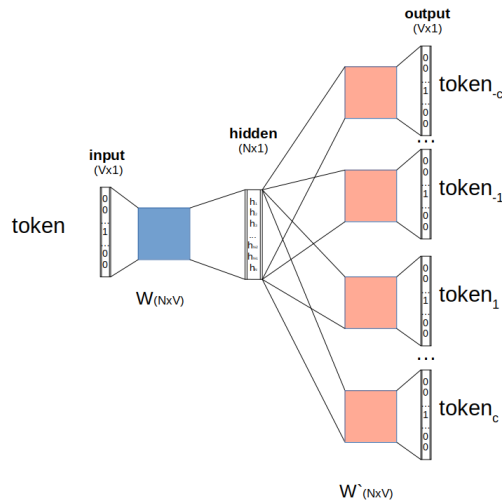


Figure 6.3: Schema della rete con architettura Skip-gram [3].

Chapter 7

Information Retrieval

L'information retrieval è un campo dell'informatica che si occupa di memorizzare e identificare documenti.

L'obiettivo è quello di definire un sistema software che permetta di memorizzare una grande quantità di documenti in un archivio, e un reperimento efficiente dei documenti rilevanti per l'informazioni che necessita un utente.

La **rilevanza** è una proprietà soggettiva difficile da definire e misurare.

Diversi tipi di sistemi per accedere a informazioni:

- Information Retrieval System (Search Engine)
- Database Management System (DBMS)
- Information Filtering System (Recommender System)

Diversi tipi di comunicazione dell'informazione:

- **Pull Technology**: l'utente richiede esplicitamente un'informazione (information retrieval, browsing hypertext, ...)
- **Push Technology**: l'utente è aggiornato automaticamente con informazioni di possibile interesse (recommendation system)

I dati sono dei fatti elementari che vanno interpretati per generare l'informazione.

$$\text{Informazione} = \text{Dati} + \text{Interpretazione}$$

Un sistema di information retrieval interpreta il contenuto di un documento e definisce una rappresentazione formale per ogni documento. Data una query fornita da un utente genera un ranking di documenti basato sulla rilevanza che hanno rispetto alla query.

Componenti del sistema:

- **Document collection:** un documento è l'unità di informazione reperibile.
- **Rappresentazione formale:** generazione di un documento surrogato basato sull'output dell'indexing.
- **Query Language:** vengono specificate le condizioni per la selezione dei documenti.
- **Matching Mechanism:** comparazione delle rappresentazioni formali di documenti e della query

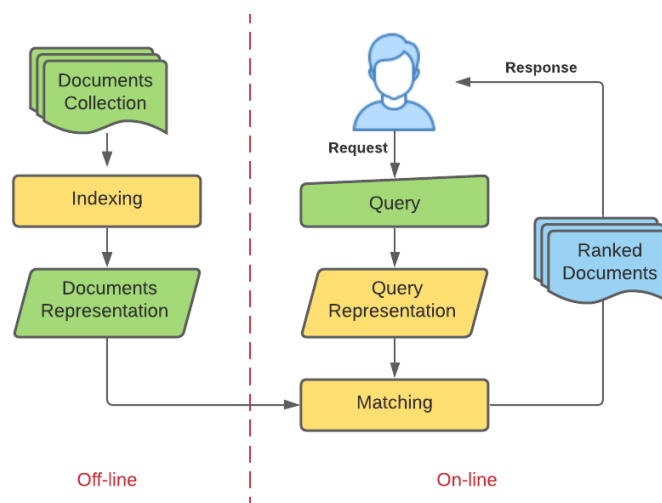


Figure 7.1: Schema di un sistema di information retrieval.

Un sistema di information retrieval è basato su un modello matematico che fornisce una descrizione formale del documento, della query e di come comparare le rappresentazioni della query e del documento per stimare la rilevanza.

L'indicizzazione è un processo basato sull'estrazione di elementi (feature) che descrivono (rappresentano) il documento. Nel caso di testi gli elementi sono generalmente parole e sono prendono il nome di indici.

Il matching può essere:

- Esatto (binario): rilevante o non rilevante
- Parziale (graduale): la comparazione tra le rappresentazioni ha una certa tolleranza ai mismatch.

Per gestire l'informazione in modo automatico bisogna considerare due fattori:

- **Efficienza:** come l'informazione viene rappresentata e processata.
- **Efficacia:** il modo in cui l'informazione viene sintetizzata e come la rappresentazione mantiene il significato originale.

Per misurare l'efficienza di un sistema di information retrieval è possibile calcolare precision e recall per ogni query.

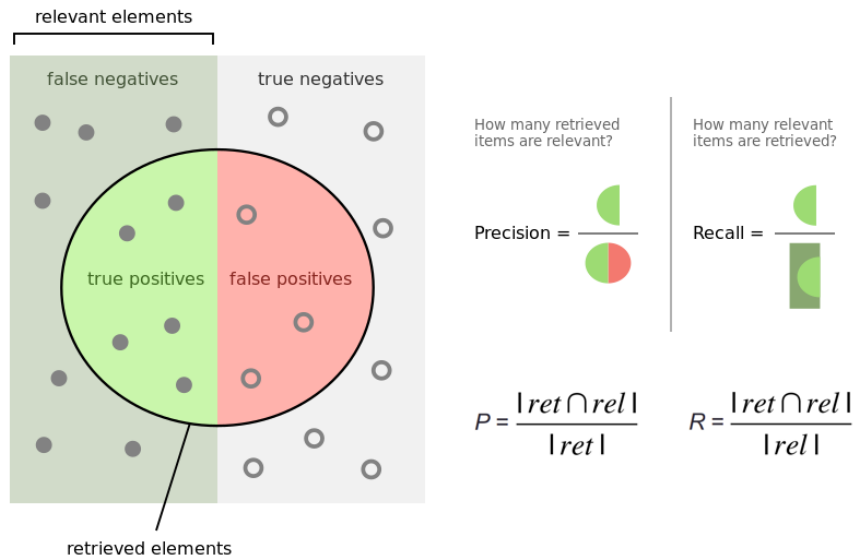


Figure 7.2: Rappresentazione di precision e recall [4].

“IR must try to satisfy information needs expressed in a vague, inaccurate way through the ambiguities of the natural language, and must compare them, in an approximate way with the information contained in a document, and expressed through the same natural language”

— Smeaton, 1997

L'information retrieval è caratterizzata da:

- **Incompletezza** nella rappresentazione dei documenti
- **Soggettività** nel concetto di rilevanza
- **Ambiguità** nel significato dei termini
- **Vaghezza** nelle richieste dell'utente
- **Incertezza** della correttezza dei risultati
- **Aprossimazione** del meccanismo di matching

7.1 Data Structures

L'indexing automatico di un documento testuale è un processo che mira ad associare gli indici a un testo.

L'uso di indici rende la ricerca di informazioni più efficiente basandosi sulle keyword specificate da una query.

L'insieme di tutti gli indici estratti da tutti i documenti della collezione costituiscono il **dizionario** della collezione.

Una volta estratti, gli indici vanno organizzati con una struttura dati con l'obiettivo di avere un accesso più efficiente possibile.

7.1.1 Inverted File Structure

Gli indici del dizionario vengono memorizzati e organizzati in un file chiamato **dictionary**. Ogni termine punta a una lista (**posting list**) che contiene le reference ai documenti di cui il termine è un indice. Le posting list sono memorizzate in un file a parte chiamato **posting file**.

Il dictionary contiene per ogni indice:

- termine
- frequenza globale (nella collezione)
- puntatore alla posting list nel posting file

La posting list può contenere per ogni elemento:

- identificatore univoco per il documento (DocID) che è associato ad un URI
- frequenza del termine nel documento (tf)
- posizione nel documento di ogni occorrenza del termine

Lo spazio richiesto dal dizionario è $O(n^\beta)$, dove n è lo spazio occupato dal testo dei documenti e β è una costante tra 0.4 e 0.6.

Lo spazio richiesto per memorizzare il numero di occorrenze è maggiore $O(n)$ perché ogni termine nel testo è referenziato una volta nella struttura dato.

7.1.2 Posting File Optimization

DocID Compression

La lista di DocID viene ordinata in ordine crescente e ogni DocID_i viene rimpiazzato dalla differenza $\text{DocID}_i - \text{DocID}_{i-1}$. I numeri ottenuti richiedono meno bit per essere codificati e permettono di ottenere una riduzione del 10-15% di spazio.

Block Division

Il testo di ogni documento viene diviso in blocchi e la posizione delle occorrenze viene puntata al blocco. Il numero di bit per codificare il puntatore sarà più piccolo del numero richiesto per le occorrenze nel punto esatto del testo.

Le ricerche contestuali però richiedono più operazioni.

7.1.3 Dictionary Organization

Linear Structure

I termini sono memorizzati in ordine alfabetico.

- accesso veloce con binary search $O(\log n)$
- memorizzazione efficiente
- adatto per la valutazione sequenziale
- per aggiungere nuovi termini bisogna ricostruire la struttura

Binary Tree Structure

I termini sono memorizzati in un albero binario. Ogni nodo ha due figli che separano gli indici a metà nell'ordine lessicografico.

La ricerca parte dalla radice e a ogni nodo viene scelto un figlio finché non si arriva al termine esatto.

- accesso veloce con binary search $O(\log n)$
- la struttura va bilanciata se vengono aggiunti o rimossi dei termini

B-Tree Structure

I termini sono memorizzati in un albero binario bilanciato di ordine d . Ogni nodo ha un numero variabile di figli e contiene un massimo di d termini e puntatori ai sotto-alberi.

La radice contiene un numero di termini tra 1 e d , mentre tutti gli altri nodi possono contenere un numero tra $d/2$ e d .

- la ricerca ha tempo $O(\log_d n)$ dove n è l'altezza dell'albero
- i termini possono essere aggiunti velocemente
- memorizzazione efficiente
- inefficiente per le ricerche sequenziali
- può diventare sbilanciato quando n cresce.

7.1.4 Altre Strutture di Indicizzazione

7.2 Models

- set-based: boolean model, fuzzy model
- algebraic: VSM, generalized VSM, latent semantic indexing
- probabilistic: belief networks, language models

7.2.1 Boolean Model

Un documento è formalmente rappresentato da un insieme di indici.

Per ogni documento d_j associamo ad ogni indice t_i un peso binario $w_{ij} \in \{0, 1\}$.

$$R(d_j) = \{t_i | w_{ij} = 1\}$$

La rappresentazione dei documenti viene invertita e associata ai termini.

$$\begin{aligned} R(d_1) = \{t_1, t_2, t_3\} &\rightarrow R(t_1) = \{d_1, d_2\} \\ R(d_2) = \{t_1, t_4, t_5\} &\rightarrow R(t_2) = \{d_1, d_3\} \\ R(d_3) = \{t_2, t_5\} &\rightarrow R(t_3) = \{d_1\} \end{aligned}$$

Esempio delle operazioni di exact matching:

$$\begin{aligned} q = t_1 &\rightarrow R(t_1) = \{d_1, d_2\} \\ q = t_1 \wedge t_2 &\rightarrow R(t_1) \cap R(t_2) = \{d_1\} \\ q = t_1 \vee t_2 &\rightarrow R(t_1) \cup R(t_2) = \{d_1, d_2, d_3\} \\ q = \neg t_1 &\rightarrow C - R(t_1) = \{d_3\} \end{aligned}$$

Una query viene rappresentata formalmente da un'espressione booleana di termini.

Ogni query può essere riscritta in forma normale disgiuntiva e ogni congiunzione rappresenta un insieme di documenti ideali.

Una query è soddisfatta da un documento se appartiene a uno dei documenti ideali identificati da una congiunzione.

$$\begin{aligned} q &= t_a \wedge (t_b \vee \neg t_c) \\ q_{dnf} &= (t_a \wedge t_b \wedge t_c) \vee (t_a \wedge t_b \wedge \neg t_c) \vee (t_a \wedge \neg t_b \wedge \neg t_c) \end{aligned}$$

L'ordine di valutazione di una query è importante, per questo è stata decisa una priorità degli operatori.

$$lowest \quad OR \rightarrow AND, NOT \rightarrow ADJ, NEAR \quad highest$$

Il matching mechanism applica delle operazioni insiemistiche.

La rilevanza è modellata come una proprietà booleana.

Bibliography

- [1] Davide Chicco. Computational algorithms to predict gene ontology annotations - scientific figure on researchgate, 04 2015. [visited on 11/11/2021].
- [2] Wikimedia Commons. File:cbow.png — wikimedia commons, the free media repository, 2020. [Online; accessed 12/11/2021].
- [3] Wikimedia Commons. File:skipgram.png — wikimedia commons, the free media repository, 2020. [Online; accessed 12/11/2021].
- [4] Wikimedia Commons. File:precisionrecall.svg — wikimedia commons, the free media repository, 2021. [Online; accessed 24/11/2021; modified].