Document Semantic Similarity

TIS Project

Alberto Pirovano Francesco Picciotti

Politecnico di Milano

5th May 2017

Outline

Introduzione

Le tecniche adottate attualmente per trovare la **similitudine semantica tra testi** si basano su tre approcci:

1. NLP Tradizionale
2. Vector Space Model
3. Deep Learning based

Outline

NLP Tradizionale

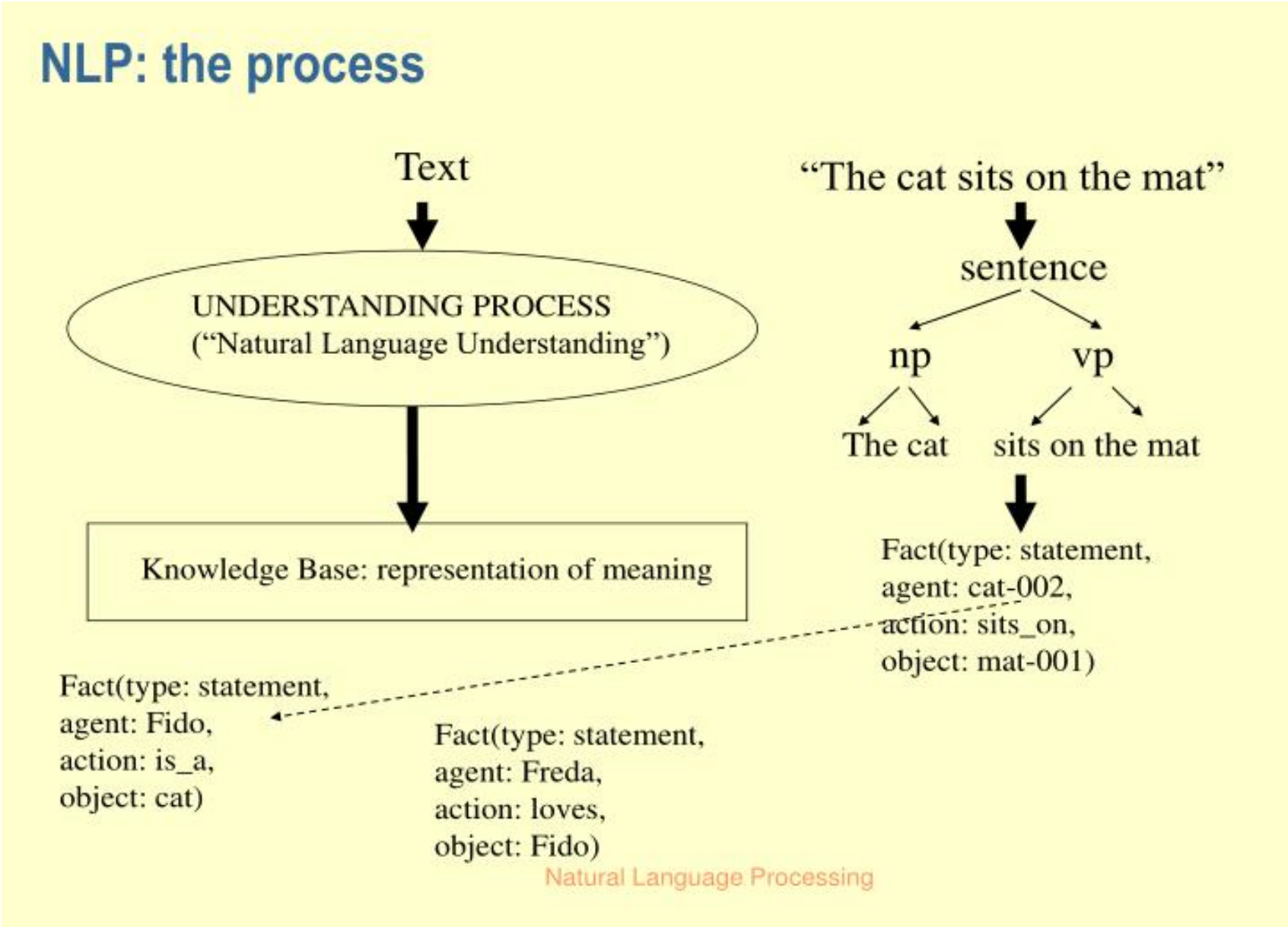
Questo approccio si basa sull'utilizzo delle tradizionali tecniche di **Natural Language Processing** e si costituisce dei seguenti step:

- Cleaning dei dati
- Pos-Tagging
- Stemming o Lemmatisation
- Parsing
- Ontologia

Tuttavia, dato che il nostro lavoro è molto **sensibile** e **dipendente** dalla qualità dei tool utilizzati, abbiamo trovato alcune consistenti **criticità** riguardanti:

- **L'affidabilità** del Pos-Tagger italiano di TreeTagger
- **Reperire** una Ontologia e un parsing tool per la lingua italiana

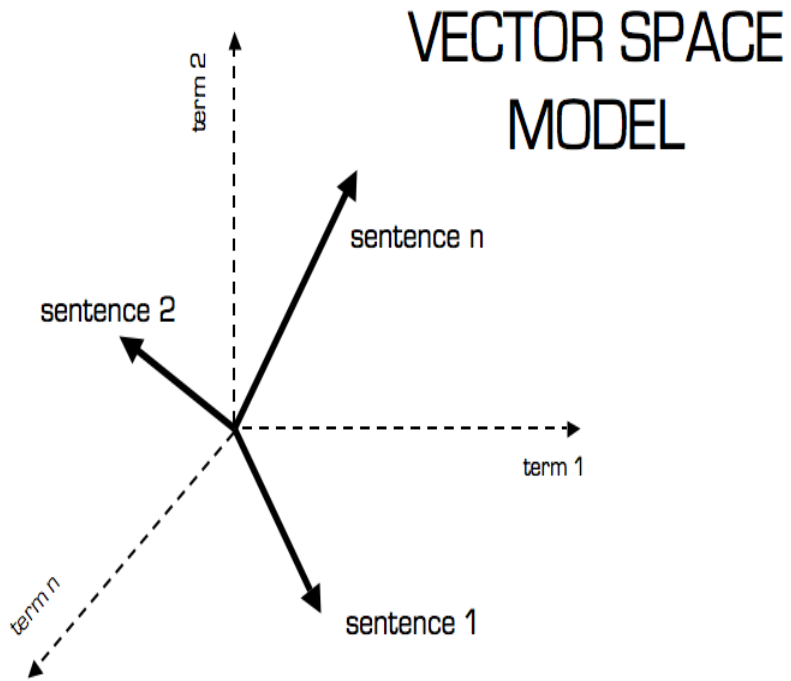
NLP Tradizionale



Outline

Vector Space Model - Explanation

Differentemente dal precedente, questo approccio ha le sue basi nello sviluppo di una **rappresentazione geometrica e vettoriale** per i documenti testuali. **Documenti e query** sono rappresentati da vettori con un numero di elementi pari al numero di termini presenti nel vocabolario. Tipicamente i termini sono le **parole distinte** presenti nell'insieme di documenti. A valle di questa rappresentazione vengono spesso utilizzate le **operazioni vettoriali** per confrontare due **documenti**.



Vector Space Model - Encoding

Nel vector space model proposto da **Salton, Wong and Yang** i vettori sono composti da **weights**, ognuna associata ad un termine del dizionario e calcolata tramite **tf-idf**.
Considerando un **documento** d_j, questo viene rappresentato tramite un **vettore** d_j:

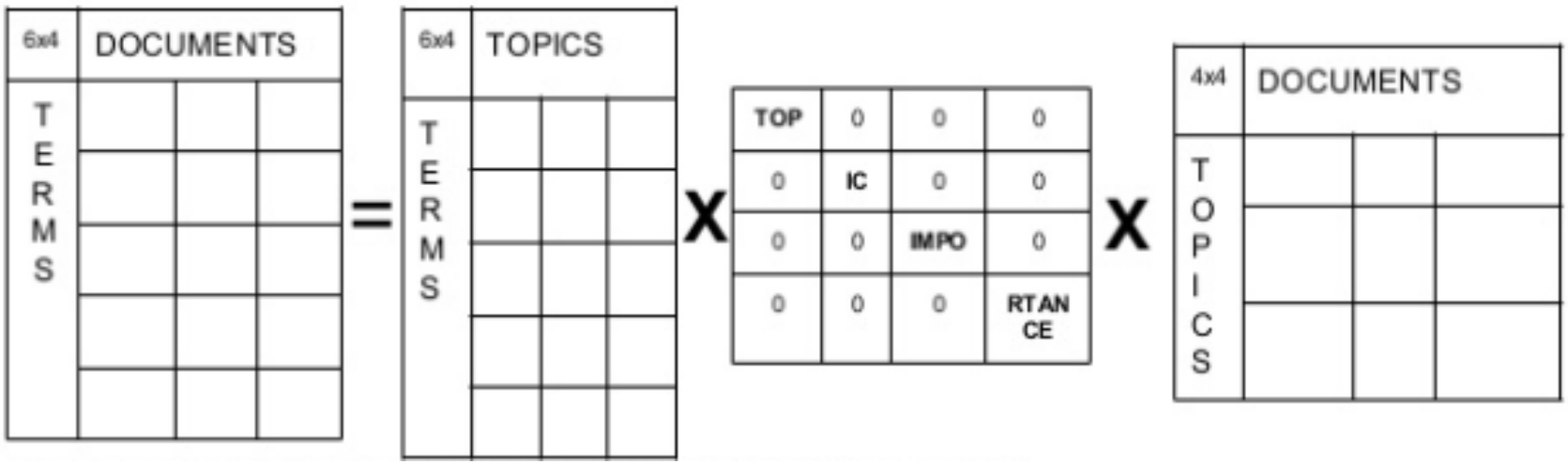
- $d_j = (w_{1,j}, w_{2,j}, ..., w_{t,j})$ **dove**: $w_{i,j} = tf_{t,d} * \log \frac{|D|}{|\{d' \in D | t \in d'\}|}$

Vector Space Model - LSA

Latent Semantic Analysis è una tecnica di **Topic Modelling** che si colloca a valle del **document encoding** con **tfidf**.
È una tecnica di **feature extraction** che permette di migliorare significativamente la qualità di un lavoro di **clustering**.
Questa procedura viene usata per generare una categorizzazione di un set di documenti in un set di topic o anche per osservare le parole che descrivono un certo topic.
Si basa sulla creazione di una **Document-Term Matrix** nella quale le **righe** rappresentano le parole del **Bag Of Words** e ha una **colonna** per **documento** nel corpora.
Il cuore di questa procedura sta nella **riduzione della dimensionalità** di questa matrice tramite **SVD**.

Vector Space Model - LSA - SVD

Find three matrices U, Σ and V so that: $X = U\Sigma V^t$



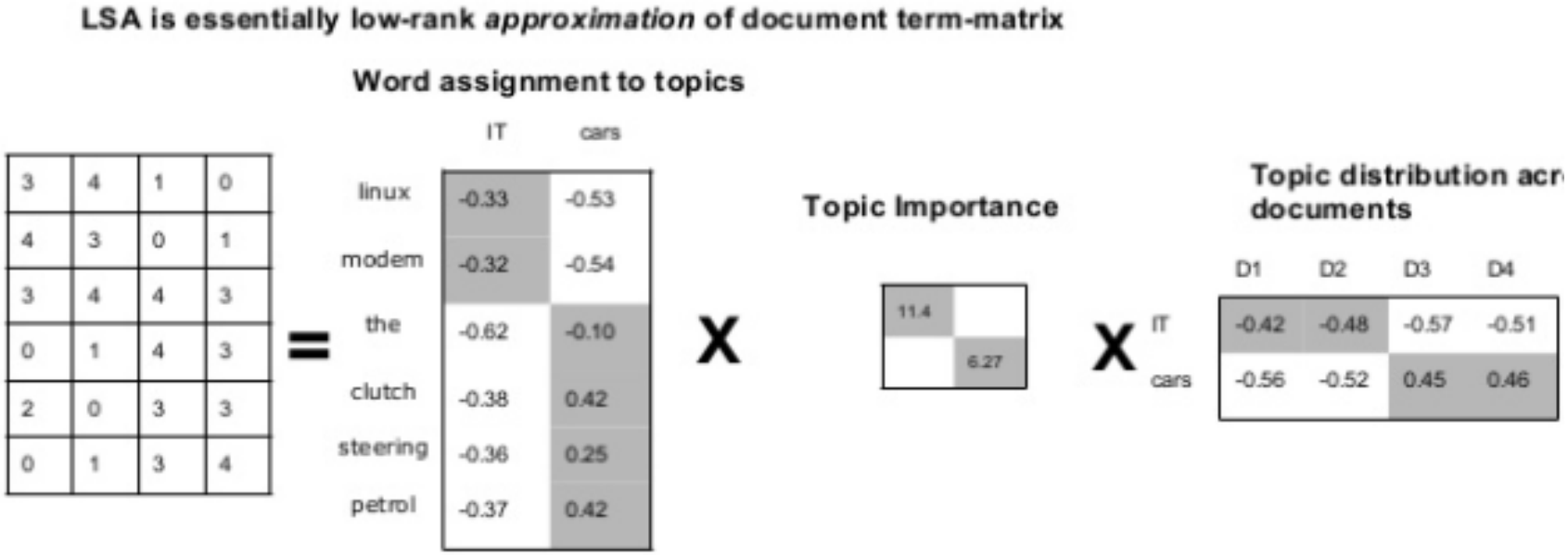
Vector Space Model - LSA

Questa procedura ci permette di:

- **Estrarre** quanti topic desideriamo da un set di documenti.
- **Conoscere** la rilevanza di un certo topic dopo averlo estratto, in questo modo siamo in grado di fermare il processo di estrazione quando i topic cominciano a diventare poco significativi.
- **Categorizzare** documenti in topic
- **Descrivere** topics con le parole del **Bag Of Words**.

Vector Space Model - LSA - Example

In questo esempio possiamo osservare la riduzione di dimensionalità:



Il processo di LSA permette di costruire le 3 matrici che vediamo sopra, ognuna con una sua utilità:

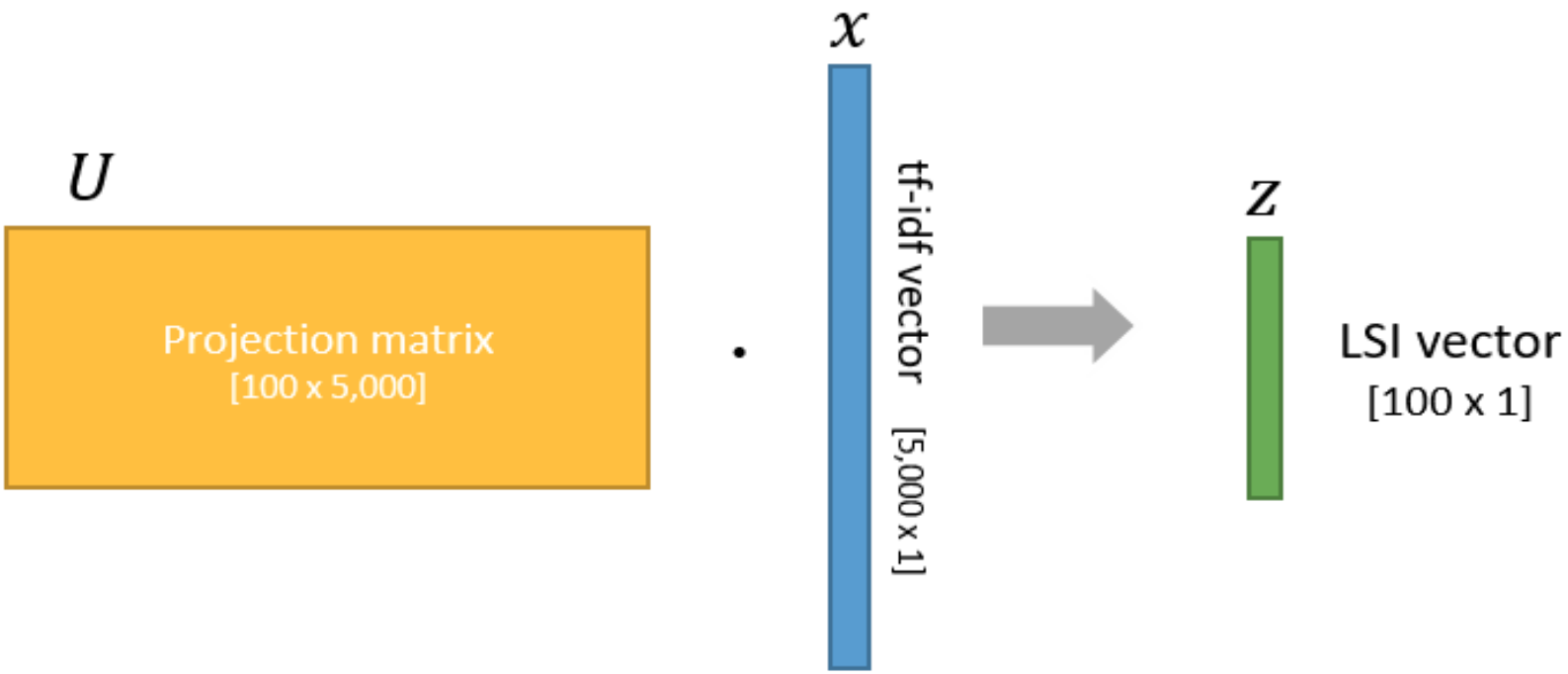
1. **Word assignment to topics**
2. **Topic importance**
3. **Topic distribution across documents**

Vector Space Model - LSA - Example

Ipotizzando un numero di topics pari a 100 e la cardinalità del **Bag of Words** pari a 5000, per ottenere la rappresentazione dei documenti in termini di topics, cioè un vettore **z**, dobbiamo:

1. **Vettorizzare** un documento in **x**.
2. **Proiettare** il **tfidf** vector **x** sul topic space.

Se definiamo la matrice $U = (Wordassignmenttotopics)^T$ possiamo visualizzare la **proiezione** in questo modo:



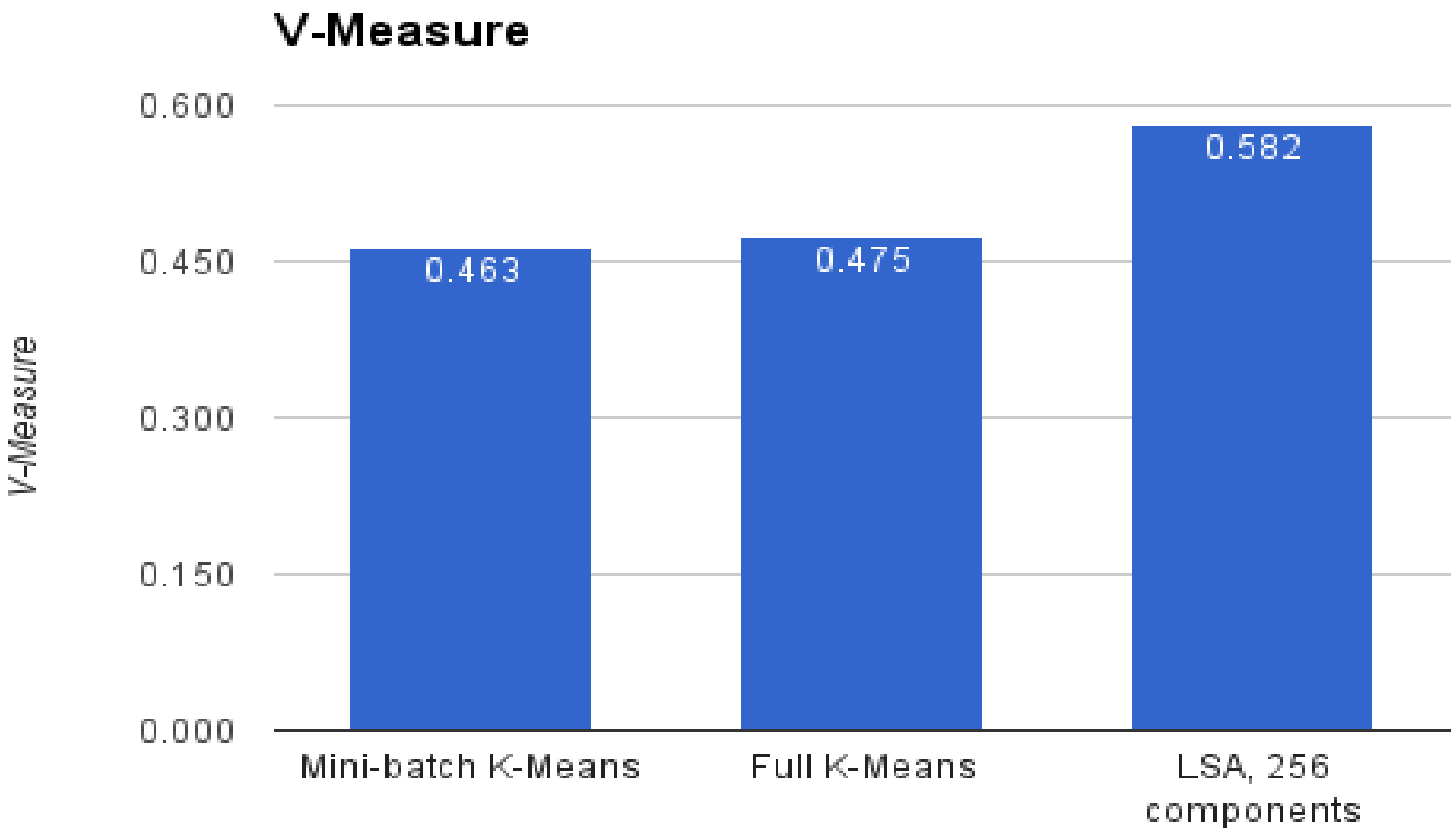
Vector Space Model - Algorithm

Riassumendo, possiamo individuare i seguenti **passaggi**:

1. Cleaning dei dati (Stemming o Lemmatisation)
2. Vectorization using TF/IDF (**x** nella figura nella slide precedente)
3. LSA (optional) (**z** nella figura precedente)
4. Clustering using similarity measure (Cosine, Pearson, ...) tra **x** vectors o **z** vectors

Vector Space Model - LSA - Performance

Come possiamo vedere nella figura, utilizzare le classiche tecniche di **clustering** senza **LSA** può portare una riduzione rilevante delle **performances**:



Vector Space Model (con LSA) - Pros and Cons

Pros:

- Modello semplice basato sull'**algebra**.
- Le **weights** non sono binarie.
- Permette di calcolare un grado di **similarità continuo**.

Cons:

- Non adatto a trattare **lunghi documenti**, infatti a causa della alta **dimensionalità** il valore **tfidf** dei componenti si riduce, riducendo di conseguenza il **dot-product**.(LSA fixes)
- **Fortemente sensibile a Falsi Negativi**, infatti documenti nello stesso **contesto** ma con diversa terminologia non saranno considerati simili.(LSA fixes)
- Perde **l'ordine delle parole**.
- **Pre-processing** dipendente.
- **Le dimensioni generate possono essere difficili da interpretare**, sensate matematicamente ma non dal punto di vista del natural language

Outline

Deep learning based

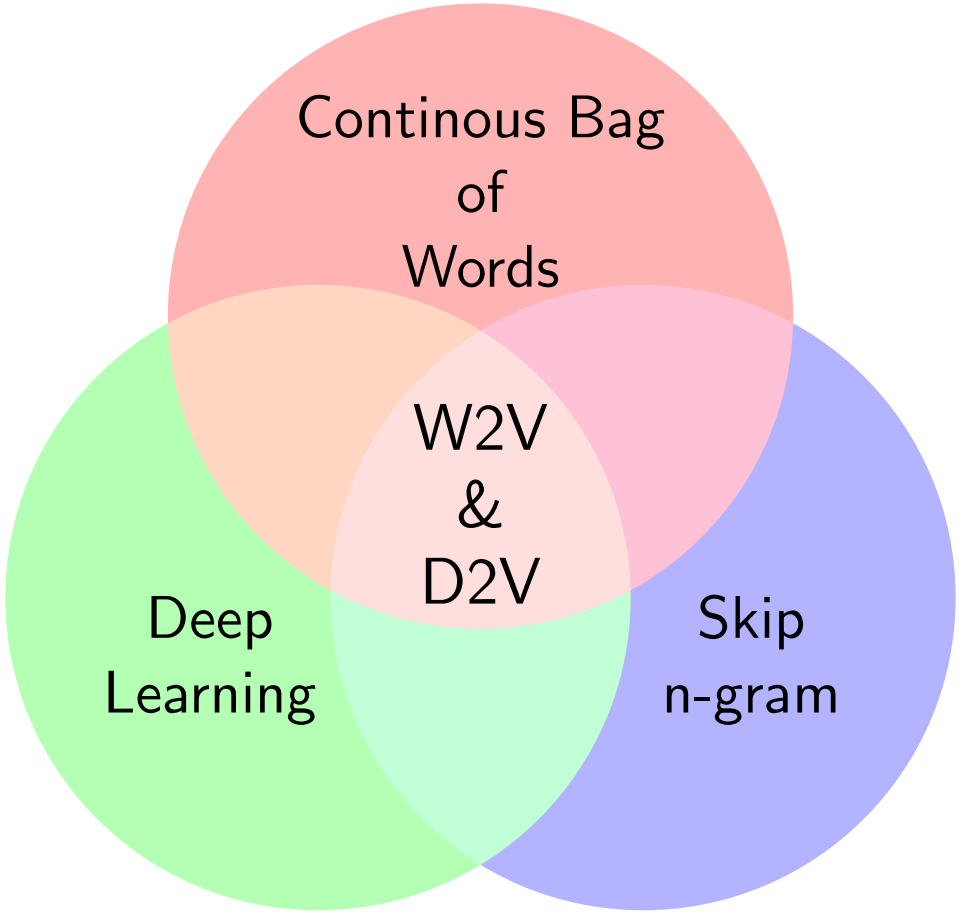
Il terzo approccio che proponiamo è molto diverso dai precedenti, infatti si basa sull'utilizzo di **neural networks**. La fondamentale differenza è che, mentre il secondo **definiva un algoritmo** per determinare le rappresentazioni vettoriali delle parole (tfidf), questo **definisce una neural network** con la task di imparare quell'algoritmo dai dati. Questa **NN** può essere **costruita** in due diversi modi, ognuno dei quali descrive **come** imparare la **word-representation** per ogni parola:

- **Continous Bag Of Words model**
- **Skip Grammar model**

Dato che il processo di apprendimento è **unsupervised**, questi modelli permettono di **determinare la task** tramite la definizione di un target per ogni input.

Deep Learning based

Questa tecnica quindi si basa su 3 ambiti:



Deep Learning based: Word2Vec & Doc2Vec

Questo metodo è stato implementato da **Google** sotto il nome di **Word2Vec** e **Doc2Vec**. Il primo permette di determinare le **relazioni semantiche** che un particolare **corpus di testi** assegna ad un **Bag Of Words** di parole. **Doc2Vec** invece è una tecnica che si configura come una **estensione di Word2Vec** la quale, preso in ingresso un set di documenti (corpora), genera un **grado di similarità**.

Deep Learning based - Pros and Cons

Dopo una ampia **discussione**, seguita da una approfondita **analisi critica** di questi due approcci, siamo giunti alle seguenti **conclusioni**, che in termini di pro e contro si possono riassumere nel seguente modo:

Pros:

- Molto meno dipende da un preprocessing
- Combina il metodo *Geometrico* con quello *NLP Tradizionale*
- **Non sfrutta una ontologia, ma la crea**
- **Language independent**
- **Tiene in considerazione anche l'ordine delle parole**, **Context-aware**

Cons:

- Tecnica **unsupervised**
- Necessità di un esperto per **validare** la similarità
- Può risultare in **GIGO** system (Garbage In Garbage Out)

Outline

Dataset

"Preprocessing is 80% of NLP work"

Lev Konstantinovskiy

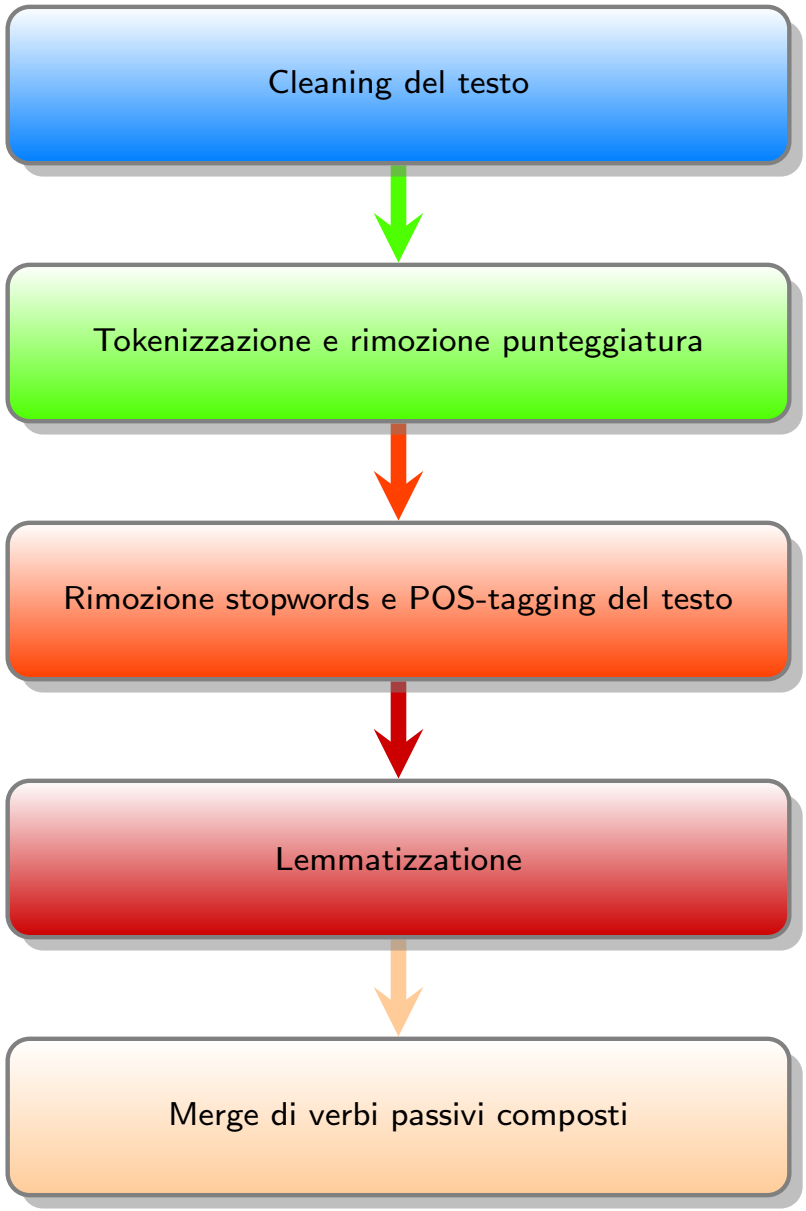
Il **dataset** si suddivide in due corpora:

- il corpus del **Sole 24 Ore** con 3265 articoli, di cui 31 non hanno body
- il corpus di **Radiocor** con 6916 articoli

Il corpus prima del **preprocessing** contiene quindi 10150 articoli.
Togliendo i **duplicati** otteniamo 9283 articoli, cioé ci sono 867 articoli duplicati.

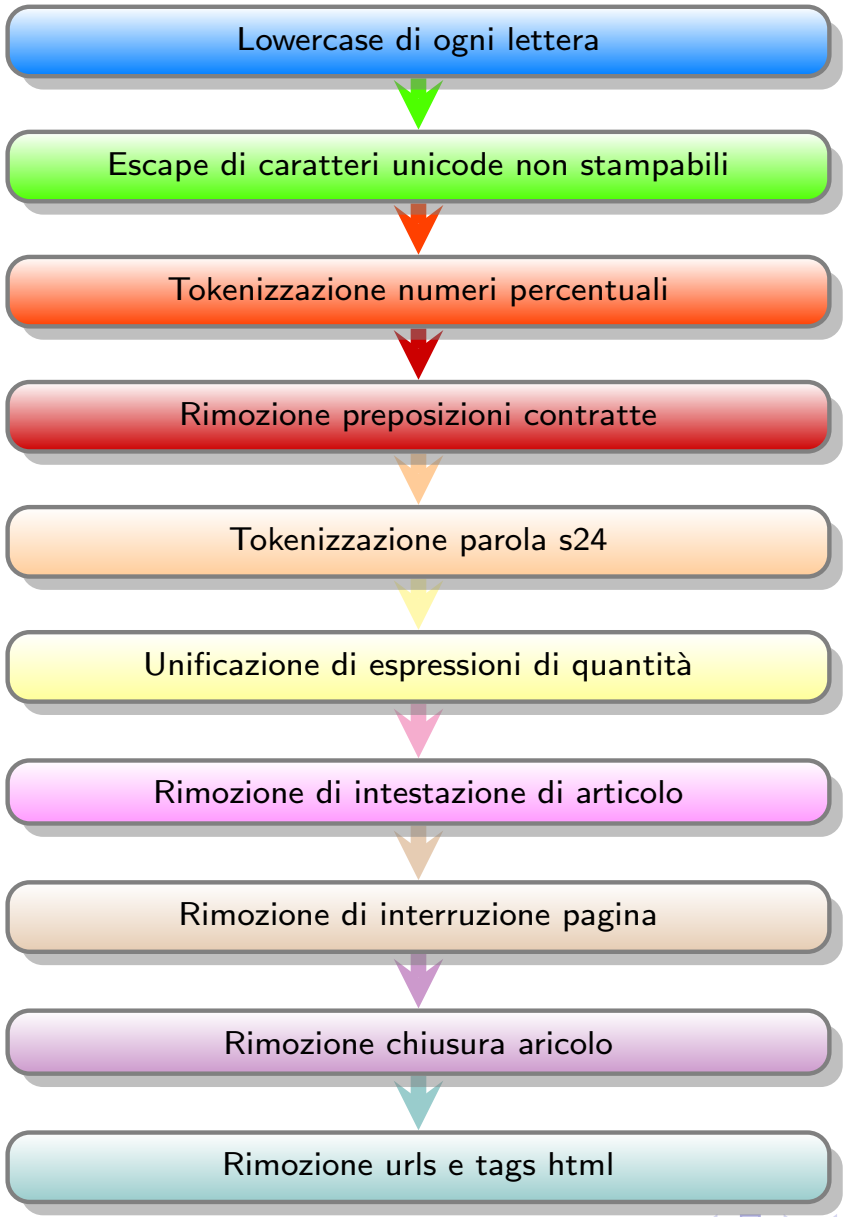
Outline

Pipeline completa



Outline

Cleaning pipeline



Outline

Word2Vec

Word2Vec è una **neural network** con un solo **hidden layer** progettata per elaborare linguaggio naturale in **unsupervised way**. L'algoritmo prende in ingresso un **corpus** e restituisce un insieme di vettori che rappresentano la **distribuzione semantica** delle parole nel testo. Questa task può essere definita in due modi:

- **Skip Grammar** - predire il contesto data la parola in input
- **Continous Bag Of Words** - predire la parola in input dato il contesto

Word2Vec - Skip Grammar - Fake task

La **training task** può essere descritta nel seguente modo:

- Data una parola in una frase, **input word**, campioniamo in modo casuale una parola *vicina* a questa; **il network modellizza** data una parola in ingresso, per ogni parola nel vocabolario, la probabilità di essere la parola che abbiamo campionato.

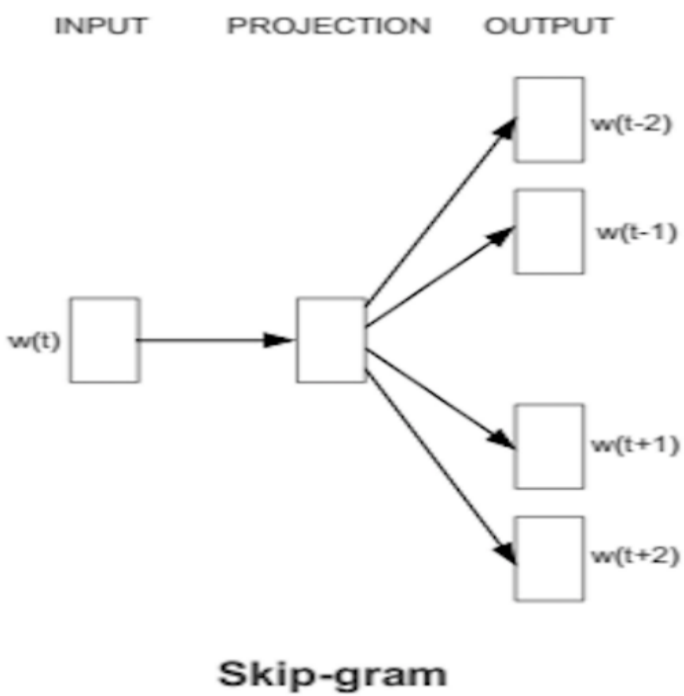
Gli **input-target pairs** sono rappresentati tramite **One Hot Encoding**, mentre l'**output** sarà un vettore di dimensione pari al numero di parole nel **vocabolario** contenente, per ogni parola, la probabilità di essere una di quelle campionate attorno alla parola di input. Riassumendo, **il network, passata una parola in input, restituisce per ogni parola nel dizionario la probabilià di capitare vicina a quella in input**, vale a dire nel suo contesto. Questo **output vector** sarà poi confrontato con il **target vector**, relativo ad una parola effettivamente vicina alla **input word**. Definendo una funzione di costo e seguendo il suo gradiente troveremo le weights che ci permettono di avere la miglior performance.

Word2Vec - Skip Grammar - Fake task

L'aspetto interessante di questa tecnica è che, dopo il training, **la rete non verrà utilizzata per la task su cui è stata addestrata**, e di conseguenza **non necessita di un test set**.

- Quello che ci interessa ottenere dopo la fase di training sono le weights dell'**hidden layer**, che rappresentano le **word-representations** delle parole del vocabolario, calcolate tenendo conto del contesto in cui vengono usate.

Word2Vec - Skip Grammar



Skip-gram

Come possiamo osservare nella figura, a partire da una parola in input vengono generati un numero di **(input, target) pairs** uguale al numero di parole che ci sono nelle "vicinanze" della parola in input. Per quantificare la "vicinanza" viene definito un **hyper-parameter** chiamato **window size** (in questo caso 2), tramite il quale vengono generati $2 * windowsize$ **(input, target) pairs**, che verranno usati per il training.

Outline

Hello

Qui speghiamo per bene come funziona doc2vec