Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Tecniche innovative di machine learning per la fruizione di contenuti digitali

Sabrina Celia, Massimo Guarascio, Marco Minici, Fabio Palopoli, Ettore Ritacco

RT- ICAR-CS-21-06 **Giugno 2021**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR)
– Sede di Cosenza, Via P. Bucci 8-9C, 87036 Rende, Italy, URL: *www.icar.cnr.it*
– Sezione di Napoli, Via P. Castellino 111, 80131 Napoli, URL: *www.icar.cnr.it*
– Sezione di Palermo, Via Ugo La Malfa, 153, 90146 Palermo, URL: *www.icar.cnr.it*

Indice

1. Introduzione

Obiettivo generale del documento è lo studio di soluzioni basate su tecniche di Artificial Intelligence e Machine Learning per migliorare l'esperienza dell'utente nella fruizione dei contenuti digitali. In particolare, saranno presi in esame i principali metodi, tecniche ed approcci di AI adottati ad oggi dalle piattaforme disponibili sul mercato evidenziandone pregi e limitazioni nell'ambito dello scenario di riferimento.

Il documento si focalizza su quali componenti caratterizzano un'interazione ottimale tra gli utenti e il sistema. Da questo punto di vista si è interessati a studiare metafore avanzate di presentazione e ricerca dei contenuti multimediali, meccanismi di interazione con gli utenti e delivery di contenuti personalizzati, sistemi di raccomandazione basati su contenuti e su collaborative filtering e loro integrazione.

Specificatamente viene fornita un'analisi critica delle limitazioni delle piattaforme attuali, focalizzandosi principalmente sui seguenti metodi e tecniche:

- Metodi ed architetture model-based per la raccomandazione
- Text mining e Natural Language Processing per l'analisi del testo e del parlato
- Multimedia Search e Information Retrieval

Nel dettaglio il documento si concentra su tre aspetti chiave nella fruizione dei contenuti digitali ossia (*i*) la raccomandazione di contenuti che incontrino i gusti degli utenti e quindi implicitamente la loro profilazione, (*ii*) l'utilizzo di strumenti di question/answering e di chatbot capaci di assistere l'utente nella ricerca di contenuti di suo interesse e (*iii*) l'impiego di strumenti di information retrieval avanzati capaci di suggerire all'utente concetti/entità e relazioni "inaspettati" partendo da item di suo interesse e l'arricchimento automatico dell'informazione contenuta catalogo tramite tagging automatico. In particolare per ognuno di questi aspetti sono stati analizzati i metodi, i modelli e gli algoritmi impiegati per risolvere questi task evidenziandone limitazioni e pregi.

Il documento è strutturato in 6 capitoli:

1. Il primo capitolo (*Introduzione*) fornisce una panoramica generale sul contenuto del documento;
2. Il secondo capitolo si focalizza sull'analisi di approcci ed algoritmi per la raccomandazione di contenuti che incontrino i gusti degli utenti e quindi implicitamente sulla loro profilazione;
3. Il terzo capitolo illustra metodi e strumenti per il text mining e il natural language processing che possono essere impiegati per lo sviluppo di chatbot e sistemi di question-answering;
4. Il quarto capitolo presenta approcci di information retrieval basati sull'impiego di Knowledge Graph capaci di suggerire all'utente concetti/entità e relazioni nuovi di item di suo interesse;
5. Il quinto capitolo si concentra sulle piattaforme e sulle tecnologie che sono attualmente impiegate in scenari reali rispetto alle tre tematiche affrontate;
6. Il sesto capitolo conclude il documento riassumendo gli studi svolti.

2. Metodi ed architetture model-based per la raccomandazione

Negli ultimi anni si è potuto constatare una crescita del mercato relativo all'erogazione di contenuti, sia in termini di numero di utenti, sia di numeri di prodotti a catalogo, sia di istanze di contenuto erogate. Le problematiche e le sfide che caratterizzano tale scenario applicativo sono in parte le stesse che affliggono altri settori, e sono legate quindi alla crescita enorme della disponibilità e della varietà di contenuti, che ostacola l'identificazione di elementi potenzialmente di interesse per l'utente. Infatti, numerosi sono gli ambiti nei quali tale sovraccarico di informazione pregiudica l'efficacia della ricerca delle informazioni dell'utente. Si consideri ad esempio piattaforme di streaming di film o di musica: in tali contesti, avere a disposizione strumenti che da un lato supportino l'utente nella ricerca dei contenuti voluti e, dall'altro, suggeriscano a quest'ultimo, proattivamente, prodotti a lui potenzialmente non noti ma di suo gradimento, è di importanza decisiva per garantire servizi soddisfacenti per l'utente e massimizzare i profitti per i provider, in termini di fidelizzazione della clientela e maggiore volume di vendite.

I sistemi di raccomandazione rappresentano lo strumento principe per arricchire l'esperienza dell'utente. In questo documento ci focalizzeremo sui modelli basati sul paradigma del Deep Learning (DL), che consente di apprendere modelli predittivi accurati anche da fonti dati di basso livello ma gestendo efficacemente la grande quantità di dati a disposizione.
La presente sezione è dunque organizzato come segue:

- Nella prima sottosezione sono introdotti i metodi e le architetture model-based (categoria alla quale appartiene il Deep Learning) per la raccomandazione, al fine di fornire una tassonomia all'interno della quale possono essere collocati i metodi e le architetture di deep learning per la raccomandazione, centrali per la presente trattazione.
- Nella seconda sottosezione saranno trattate quindi le tecniche di deep learning per i sistemi di raccomandazione.

2.1. Sistemi di raccomandazione

Nella presente sezione ci si pone l'obiettivo di presentare una panoramica sugli strumenti, i metodi e le tecniche utilizzati nell'ambito del supporto alla fruizione di contenuti. Le variabili da considerare in un processo di recommendation sono numerose e complesse, e la loro analisi richiede numerosi passi di knowledge discovery. In primo luogo si può considerare lo studio del profilo degli utenti in esame: ciò permette di caratterizzare gli individui in base alle loro proprietà demografiche, ma anche in base alle loro attitudini personali, evidenziandone le preferenze generali. Al di là dei gusti personali, gli individui possono essere sono spinti da fattori ambientali esterni, dovuti al fatto che le persone agiscono all'interno di un ambiente "sociale" che ne può influenzare il comportamento.

Infine, osservare lo storico delle azioni degli utenti, è di fondamentale importanza per la comprensione del loro comportamento. È possibile in tal modo definire dei modelli comportamentali che descrivono le azioni svolte in passato e che possono predire alcune azioni future.

Da quest'ultimo punto di vista, in letteratura è possibile identificare tre principali categorie di modelli per la raccomandazione:

- *Content-based system*. Sono modelli che basano il suggerimento sulle proprietà dei prodotti (*categoria, marca, modello, qualità, prezzo, descrizione, ecc.*) e sui profili utenti (*età, sesso, residenza, titolo di studio, professione, descrizioni caratteriali, ecc.),* creando un'associazione tra di essi.
- *Collaborative filtering*. È una famiglia di approcci alla recommendation basata sulla collezione ed analisi delle preferenze espresse dagli utenti, senza considerare le loro informazioni demografiche e le caratteristiche dei prodotti. L'idea, che governa il collaborative filtering, assume che utenti, che abbiano esibito comportamenti simili in passato, continueranno ad agire in modo simile anche in futuro.
- *Approcci ibridi*. Includono una serie di metodi e tecniche per combinare il collaborative filtering con i sistemi content-based. L'obiettivo è perciò quello di cercare di prendere il meglio dalle tecniche esistenti al fine di ottenere suggerimenti molto graditi agli utenti.

Questa sezione, benché miri ad offrire una panoramica su tutte le tecniche disponibili, porrà particolare attenzione al collaborative filtering, poiché gode di una caratteristica molto interessante ossia non richiede conoscenza semantica dei dati (spesso non disponibile) ma si limita ad utilizzare solo le preferenze esplicite degli utenti nei riguardi dei prodotti. Ciò consente di elaborare grandi moli di dati e, inoltre, di concentrarsi unicamente sulla definizione dei modelli comportamentali degli utenti, ignorando una serie di informazioni che in alcuni casi possono risultare fuorvianti. Inoltre, particolare attenzione sarà rivolta sulle tecniche che impiegano il deep learning per apprendere modelli di raccomandazione. Queste tecniche hanno ottenuto un forte interesse dalla comunità scientifica data la loro efficacia e la loro capacità di poter lavorare anche su dati *raw* senza la necessità di operazioni di preprocessing e feature engineering.

Prima di entrare nel caso specifico dei metodi e architetture model-based per la raccomandazione, si analizzano di seguito le fasi che compongono un processo di recommendation, tipicamente comuni a tutti gli approcci, al fine di mettere in evidenza le differenze con gli altri approcci.

2.1.1. Fasi di un processo di raccomandazione

Sebbene esistano diversi approcci per la definizione di un processo di raccomandazione, quest'ultimo è in genere composto da fasi condivise tra tutti. In particolare, è possibile suddividere tale processo nelle seguenti fasi:

- *Collezione delle informazioni*
- *Learning*
- *Predizione/Raccomandazione*

Collezione delle informazioni

In questa fase sono raccolte informazioni rilevanti riguardo gli utenti. Gli obiettivi perseguiti sono generare un profilo utente o un modello per i task di predizione. Tra le informazioni raccolte, gli attributi dell'utente, i suoi comportamenti o il contenuto delle risorse a cui esso accede. Un algoritmo di

raccomandazione non può funzionare in modo accurato finché il profilo/modello utente non è stato ben costruito. Il sistema ha bisogno di sapere quanto più possibile dall'utente e dell'utente per fornire una raccomandazione ragionevolmente efficace fin dall'inizio.

I sistemi di raccomandazione si basano su diversi tipi di input tra i quali, feedback espliciti forniti dagli utenti e riguardanti il loro interesse nei confronti di elementi, o viceversa input impliciti, ottenuti inferendo indirettamente le preferenze dell'utente attraverso l'osservazione del suo comportamento [Oar98]. Altra tipologia di feedback è costituita dai feedback di tipo ibrido. Essi possono essere ottenuti, tra l'altro, dalla combinazione di feedback impliciti ed espliciti.

In una piattaforma di E-learning, un profilo utente è costituito da una collezione di informazioni personali associate a un utente specifico. Esempi di tali informazioni sono gli skill cognitivi, le abilità intellettuali, lo stile di apprendimento, gli interessi, le preferenze e le interazioni con il sistema. Il profilo utente è tipicamente utilizzato per recuperare informazioni che serviranno per costruire un modello dell'utente. Quindi, semplificando, è possibile affermare che un profilo utente descrive un semplice modello dell'utente.

Il successo di un metodo di raccomandazione dipende largamente dalla qualità e dalla completezza della rappresentazione degli interessi correnti dell'utente. In altri termini, perché le tecniche di predizione possano fornire raccomandazioni rilevanti e accurate è indispensabile disporre di modelli accurati.

Come accennato, esistono essenzialmente tre tipologie di filtri:

- *Feedback espliciti*. Il sistema tipicamente richiede all'utente attraverso la sua interfaccia di fornire valutazioni per gli articoli al fine di costruire e migliorare il suo modello. L'accuratezza della raccomandazione dipende dalla quantità delle valutazioni fornite dall'utente. Uno dei limiti che si possono ascrivere a questo metodo è che richiede il coinvolgimento e la disponibilità pressoché continuativa dell'utente a fornire informazioni puntuali e complete. Malgrado ciò, il feedback esplicito è considerato il più affidabile, poiché non implica l'estrazione delle preferenze dalle azioni e fornisce anche trasparenza nel processo di raccomandazione. Ciò ha in genere come vantaggio il fatto che la qualità delle raccomandazioni sia in genere percepita come migliore rispetto ai casi in cui vengono utilizzate altre tipologie di feedback, e di conseguenza porta una maggiore fiducia nelle raccomandazioni.
- *Feedback impliciti*. Il sistema inferisce automaticamente le preferenze dell'utente, monitorando le varie azioni eseguite da quest'ultimo. Ad esempio, si tiene tipicamente traccia dello storico degli acquisti, quello della navigazione, il tempo passato sulle singole pagine web, i link seguiti, e i click. Una delle differenze sostanziali con i feedback impliciti risiede nel fatto che non viene richiesto nessun effort all'utente, tuttavia tali feedback risultano generalmente meno accurati rispetto al caso precedente. Tuttavia, in alcuni studi si sostiene invece che i dati relativi ai feedback impliciti potrebbero essere in realtà più oggettivi, in quanto non vi è alcun pregiudizio derivante dal fatto che gli utenti rispondano in un modo socialmente desiderabile e non ci sono problemi di immagine di sé o alcuna necessità di mantenere un'immagine per gli altri.
- *Feedback ibridi*. I due approcci prima presentati possono essere combinati, dando vita ai feedback ibridi, al fine di minimizzare le debolezze prima evidenziate, migliorando le perfomance del sistema. Ciò può essere realizzato ad esempio usando i feedback impliciti come

controllo per i feedback espliciti, oppure consentendo all'utente di fornire feedback espliciti solo quando egli proattivamente fornisce la disponibilità per farlo.

Learning

In questa fase, viene utilizzato un algoritmo di learning per filtrare i feedback raccolti nella fase precedente. Gli algoritmi utilizzati saranno discussi nel seguito del documento.

Fase di predizione/raccomandazione

In questa fase, viene effettuata la predizione o la raccomandazione su quale tipo elementi potrebbero essere preferiti dall'utente. Ciò può essere fatto o direttamente sulla base del dataset popolato nella fase della collezione delle informazioni, che potrebbe essere memory based o model based, o sulla base delle attività dell'utente osservate dal sistema. La seguente figura mette in evidenza le fasi della raccomandazione.

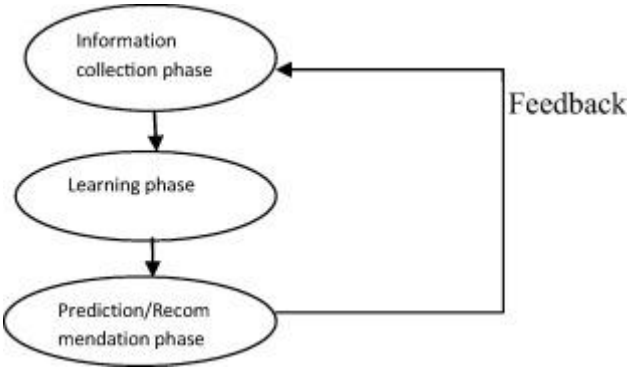

FIGURA 1 SCHEMA DI APPRENDIMENTO

2.1.2. Approcci alla raccomandazione

In questa sezione viene proposta una panoramica su varie tipologie di tecniche di Recommendation Filtering. La seguente figura mostra una tassonomia delle tecniche disponibili.

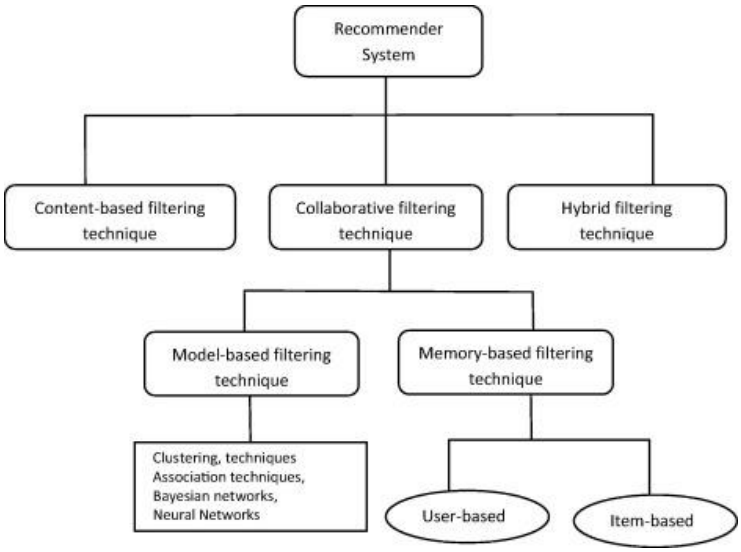

FIGURA 2 GERARCHIA DELLE TECNICHE DI RACCOMANDAZIONE

2.1.2.1. Content based filtering

Le tecniche content based sono basate su algoritmi dipendenti dal dominio, ed enfatizzano maggiormente l'analisi degli attributi degli item, al fine di generare predizioni. Quando documenti come ad esempio pagine web, pubblicazioni e news devono essere raccomandati, le tecniche di content based filtering sono quelle che ottengono un successo maggiore. Nell'ambito delle tecniche di content-based filtering le raccomandazioni vengono fatte basandosi sui profili utenti, usando le feature estratte dal contenuto degli item che gli utenti hanno valutato precedentemente [Bur02, Bob13]. Gli item che sono maggiormente correlati agli item valutati positivamente vengono raccomandati all'utente.

Le tecniche di content based filtering usano differenti tipi di modelli per trovare le similarità tra documenti, al fine di generare raccomandazioni significative. Tali tecniche possono usare, ad esempio, un Vector Space Model come un Term Frequency Inverse Document Frequency (TF/IDF) o modelli probabilistici come classificatori bayesiani Naive, gli alberi di decisione o le reti neurali per modellare la relazione tra differenti documenti all'interno di un corpus.
Tali tecniche generano raccomandazioni apprendendo il modello sottostante, sia per mezzo di analisi statistiche, sia per mezzo di tecniche di machine learning.

Le tecniche di content-based filtering non necessitano di conoscere il profilo di altri utenti, dal momento che essi non influenzano le raccomandazioni. Inoltre, se il profilo dell'utente cambia, le tecniche di content based filtering hanno ancora il potenziale di modificare le raccomandazioni generate in un tempo molto breve. Lo svantaggio maggiore di tali tecniche è il bisogno di disporre di conoscenza approfondita e della descrizione delle feature degli item nei profili.

2.1.2.2. Collaborative filtering

Dato un insieme di utenti U ed un catalogo di item I, l'input che un approccio di collaborative filtering si aspetta è una matrice di preferenze R. Questa matrice prevede per ogni riga le valutazioni di un utente su una serie di prodotti il cui codice identificativo è rappresentato da una colonna; la valutazione è

espressa in forma booleana (mi piace/non mi piace) oppure come valore numerico (indice di gradimento). Un esempio è:

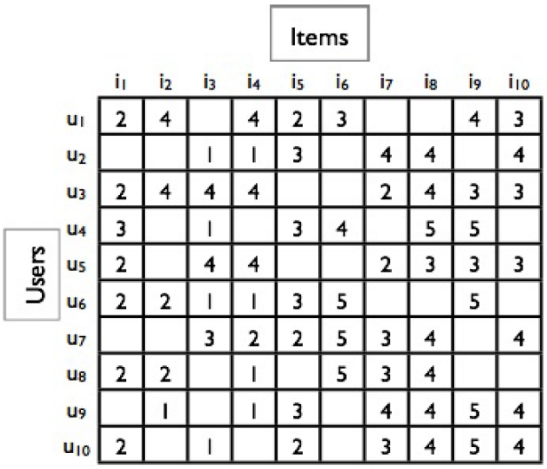

FIGURA 3: ESEMPIO DI MATRICE DEI RATING

La prima riga corrisponde alle preferenze dell'utente u_1, che ha espresso un voto, rating, pari a 2 sugli item (prodotti) i_1 e i_5, 4 per l'item i_2, i_4 e i_9, ed infine 3 per gli item i_6 e i_10. È da notare che nei casi reali la matrice delle preferenze è molto sparsa, con un fattore di sparsità molto spesso superiore al 95%. Ad esempio si consideri il sito di recensione film IMDb (http://www.imdb.com/), il suo sistema può vantare milioni di utenti e migliaia di film, il che significa che la sua matrice di preferenze ha milioni di righe e migliaia di colonne. Prendendo in analisi una sola riga, quindi un solo utente, ci si può chiedere: "quanti film ha potuto valutare questo utente?" Considerando un utente medio, si possono contare un numero di valutazioni inferiori alle centinaia, quindi il resto della riga contiene numerose celle vuote (senza valutazione).

Il processo di collaborative filtering viene diviso in due fasi: inferenza e predizione. La fase di inferenza genera un modello predittivo M a partire da una matrice di preferenze R. Tutti gli algoritmi di collaborative filtering costruiscono M sfruttando le preferenze espresse, che per comodità, nel seguito del documento, verranno indicate con r_u^i ovvero il rating indicato dall'utente u in merito all'item i. La fase di valutazione prende in analisi un altro set di preferenze T (ignoto alla fase di inferenza) di cui si vuole predire il contenuto in termini di rating o in termini di grado di apprezzamento generico. Nel seguito del documento, la predizione di un rating di un utente u per un item i, verrà indicata come r_u^i.

L'output atteso, nel corso degli anni, è stato modificato in base a varie interpretazioni del termine "*recommendation*". Inizialmente, per recommendation si è inteso l'atto di riempire le celle mancanti di R, con dei valori predetti di rating che fossero il più simile possibile ad un ipotetico valore reale espresso dagli utenti in analisi, per poi suggerire loro i prodotti con il rating predetto più alto. Per tal motivo, la recommendation è stata valutata in termini di precisione della predizione. Una tra le misure più diffuse, per questa misura, è la radice dell'errore quadratico medio, RMSE, definita come:

$$RMSE = \sqrt{\frac{1}{|T|}\sum_{(u,i)\in T} \left(r_i^u - \hat{r}_i^u\right)^2}$$

Supponendo di conoscere i rating all'interno di T. Quindi, generare una buona recommendation significa costruire M in modo da minimizzare l'RMSE in T.

Successivamente, ad affiancare l'RMSE è entrato in gioco un altro gruppo di valutazioni qualitative. L'idea di base si poggia sul concetto che buoni valori di RMSE non sempre corrispondono ad una recommendation. L'obiettivo della raccomandazione è quello di suggerire agli utenti brevi liste di item, L, di notevole interesse per loro. Per tal motivo, M deve essere valutato per la sua capacità di generare liste di raccomandazioni con item simili a quelli più graditi presenti in T: sotto questa prospettiva, si misura la qualità della raccomandazione e non la qualità della predizione di M. Le più diffuse misure di qualità della raccomandazione sono la precision, la recall e la F-Measure (e loro varianti):

$$precision = \frac{1}{|U|}\sum_{u\in U} \frac{L_u \cap T_u}{L_u};\ recall = \frac{1}{|U|}\sum_{u\in U} \frac{L_u \cap T_u}{T_u};$$
$$F - Measure = 2\frac{precision \cdot recall}{precision + recall}$$

Dove L_u è la lista di raccomandazione generata da M per l'utente u e T_u è l'insieme dei record in T relativi all'utente u.

2.1.2.3. Approcci ibridi

Sono una serie di metodi e tecniche per combinare il collaborative filtering con i sistemi content-based. L'obiettivo è quello di cercare di prendere il meglio dalle tecniche esistenti al fine di ottenere suggerimenti molto graditi agli utenti.

2.2. Modelli di raccomandazione

2.2.1. Baseline models

Questi modelli cercano di minimizzare l'RMSE per generare le recommendation list e rappresentano delle trivial baseline. Il modello più intuitivo che si possa pensare è definito come OverallMean: ad ogni osservazione (coppia utente/item) in T, si associa il rating medio calcolato su tutte le valutazioni in R. In alcuni contesti, in cui la varianza dei gradi di preferenza è piuttosto bassa, questo modello ottiene valori molto bassi di RMSE, ma il suo grado di personalizzazione è nullo, il che si riflette in liste di raccomandazione non soddisfacenti: un item sgradito ed uno gradito (così come ogni altro item) hanno lo stesso rating predetto.

I primi due modelli che introducono un minimo di personalizzazione sono noti come UserAvg e ItemAvg. UserAvg (risp. ItemAvg), calcola, per ogni utente (risp. item), la sua media di rating in R, $\underline{r}^u$ (risp. $\underline{r}_i$), ed assocerà ad ogni osservazione in T il corrispondente valore di media per utente (risp. item). Questi due modelli hanno una precisione che è al minimo è equivalente a quella dell'OverallMean (nel caso in cui la media di rating per utente/item sia una costante): l'RMSE rimane basso se la varianza di rating per ogni utente (risp. item) è bassa, rilassando il vincolo sulla varianza globale dell'OverallMean.

L'ultima baseline presentata in questo documento è nota come DoubleCentering. Il DoubleCentering è una combinazione lineare tra l'UserAvg e l'ItemAvg, la cui predizione di rating è definita come:

$$\hat{r}_i^u = \alpha \cdot \underline{r}^u + (1 - \alpha) \cdot \underline{r}_i$$

con $0 \leq \alpha \leq 1$ che può essere stimato attraverso classiche tecniche di inferenza, come ad esempio la stima ai minimi quadrati rispetto l'errore di predizione su R:

$$\alpha = arg\ arg \sum_{\langle u,i,r_i^u \rangle \in R} \left[r_i^u - \alpha \cdot \underline{r}^u - (1 - \alpha) \cdot \underline{r}_i \right]^2$$

Questo modello offre un grado di personalizzazione superiore alle altre baseline, ma ancora insufficiente per generare liste di raccomandazione di qualità.

2.2.2. Modelli Neighborhood-based

Una prima famiglia di modelli che ha dato buoni risultati in termini di RMSE e che allo stesso tempo ha dimostrato di essere utile per la raccomandazione è quella dei modelli per prossimità, detti anche Neighborhood-based. L'idea di fondo è di generare la raccomandazione in base alla similarità comportamentale degli utenti del sistema. Si supponga che gli utenti del sistema siano due, u_1 e u_2, e che gli item, in R, per cui abbiano espresso parere positivo siano rispettivamente {a,b,c,d} e {a,b,c,e}. Le due osservazioni sono identiche a meno di un solo item, il che fa supporre che i due utenti abbiano gusti simili: per tal motivo, è consigliabile suggerire l'item d all'utente u_2 dal momento che è piaciuto ad u_1, e l'item e all'utente u_1, perché gradito da u_2.
Questo esempio lo si può estendere al caso con più utenti, mediando il contributo alla predizione di tutti gli utenti. In questo caso la funzione di predizione è:

$$\hat{r}_i^u = \frac{\sum_{u' \in U \setminus \{u\}} sim_{u,u'} \cdot r_i^{u'}}{\sum_{u' \in U \setminus \{u\}} sim_{u,u'}}$$

dove $sim_{u,u'}$ rappresenta il grado di similarità tra l'utente u ed il generico utente u'. Questa formulazione è troppo generale e poco praticabile nei casi reali, poiché è quadratica nel numero di utenti non solo in fase di generazione del modello, si consideri il calcolo di tutte le similarità, ma anche in fase

di predizione, si notino le sommatorie nella formula. Quindi, da questa formulazione di base, nascono numerosi algoritmi in base a come semplificare le sommatorie (ad esempio l'algoritmo *k-NN* che per ogni utente seleziona solo i *k* utenti più simili, con *k* costante fornito da input) e a quale funzione di similarità si sceglie (Jaccard, similarità del coseno, coefficiente di Pearson…); inoltre, a questi modelli, si aggiungono le varianti che utilizzano fattori di regolarizzazione, per bilanciare il fatto che alcuni utenti possono essere più propensi ad dare rating alti, mentre altri tendono a dare rating bassi. In definitiva, considerando un set limitato di utenti simili per ogni utente u, S_u, ed il fattore di regolarizzazione, la formulazione diventa:

$$\hat{r}_i^u = \underline{r}^u + \frac{\sum_{u' \in S_u} sim_{u,u'} \cdot \frac{\underline{r}^u \cdot \left(r_i^{u'} - \underline{r}^{u'}\right)}{\underline{r}^{u'}}}{\sum_{u' \in S_u} sim_{u,u'}}$$

Il reale punto debole di questo modello è sulla difficoltà di scelta della funzione di similarità. Non esiste un criterio a priori che possa suggerire una funzione di similarità a discapito di altre e, per di più, diverse funzioni di similarità possono portare a risultati completamente diversi. Inoltre, come già accennato, questa famiglia di modelli ha problemi di scalabilità nel caso di grandi moli di dati e risente molto del fattore di sparsità della matrice delle preferenze (un palliativo è calcolare le similarità per item, invece di quelle per utente, dal momento che solitamente le colonne di R sono meno sparse delle sue righe).

2.2.3. Matrix Factorization

Per ottenere sistemi di raccomandazione soddisfacenti, il mondo scientifico si è rivolto all'algebra lineare. L'idea di base è quella di fattorizzare la matrice R [5] in due sottomatrici A e B, capaci di riassumere con buona approssimazione R, ma al tempo stesso dare generalità alla predizione ed essere scalabile in presenza di grandi moli di dati:

$$R \approx A \times B$$

La matrice A ha un numero di righe pari al numero di utenti ed un numero di colonne pari a K, fornito in input, mentre B ha K righe e colonne quanti sono gli item. Il numero K rappresenta il numero di fattori latenti da prendere in considerazione: un fattore lantente, nel caso della raccomandazione, può essere considerato semanticamente come un "motivo", detto topic, che spinge gli utenti ad apprezzare o meno un determinato prodotto, ovvero la risposta alla domanda "perché l'utente apprezza l'item?". Quindi la matrice A indica l'interesse degli utenti verso i topic, mentre B indica il grado di appartenenza di un item per un topic. Nell'ottica di questo modello, la predizione di rating, per l'utente u nei riguardi dell'item i, è intesa come il prodotto scalare tra la riga u di A e la colonna i di B:

$$\hat{r}_i^u = \sum_{k=1}^{K} A_{u,k} \cdot B_{k,i}$$

La stima delle matrici A e B può avvenire in diversi modi, in base ai diversi algoritmi proposti in letteratura. Si può utilizzare una stima ai minimi quadrati:

$$(A,B) = arg\ arg \sum_{\langle u,i,r_i^u \rangle \in R} \left[r_i^u - \sum_{k=1}^{K} A_{u,k} \cdot B_{k,i} \right]^2$$

tramite grandiente discendente:

$$A'_{u,k} = A_{u,k} + \eta\left[(\hat{r}_i^u - r_i^u) \cdot B_{k,i}\right] B'_{k,i} = B_{k,i} + \eta\left[(\hat{r}_i^u - r_i^u) \cdot A_{u,k}\right]$$

dove η rappresenta il tasso di apprendimento; oppure tramite altri metodi di stima. Anche per la matrix factorization si possono utilizzare varianti basate sulla regolarizzazione.
Sebbene questa famiglia di modelli abbia dimostrato in campo scientifico la sua validità, soprattutto in termini di RMSE (si veda il caso Netflix Prize, http://www.netflixprize.com/), porta con sé, comunque, alcuni effetti collaterali. In primo luogo, le matrici A e B non hanno una semantica chiara: è difficile dare un'interpretazione ai loro valori visto che spaziano su tutto il campo dei numeri reali. Ed in seconda analisi, risulta essere un modello molto prono all'overfitting.

2.2.4. Probabilistic models

Tra gli approcci di maggiore successo nel campo della raccomandazione, sia in termini di qualità predittiva sia in termini di qualità della lista dei suggerimenti, si trova la famiglia degli modelli probabilistici, noti con i termini Probabilistic Latent Factor Models o Probabilistic Topic Models, dal momento che, come per la matrix factorization, anche loro utilizzano il concetto di topic come fattore latente, che motiva il grado di apprezzamento degli utenti nei confronti degli item. Una pietra miliare, tra gli approcci probabilistici, risulta essere la Latent Dirichlet Allocation (LDA).

L'idea alla base di questo modello risiede nell'associare ad ogni utente una distribuzione di probabilità sul dominio dei topic, e per ogni topic una distribuzione di probabilità sullo spazio degli item.
Si consideri ad esempio la figura:

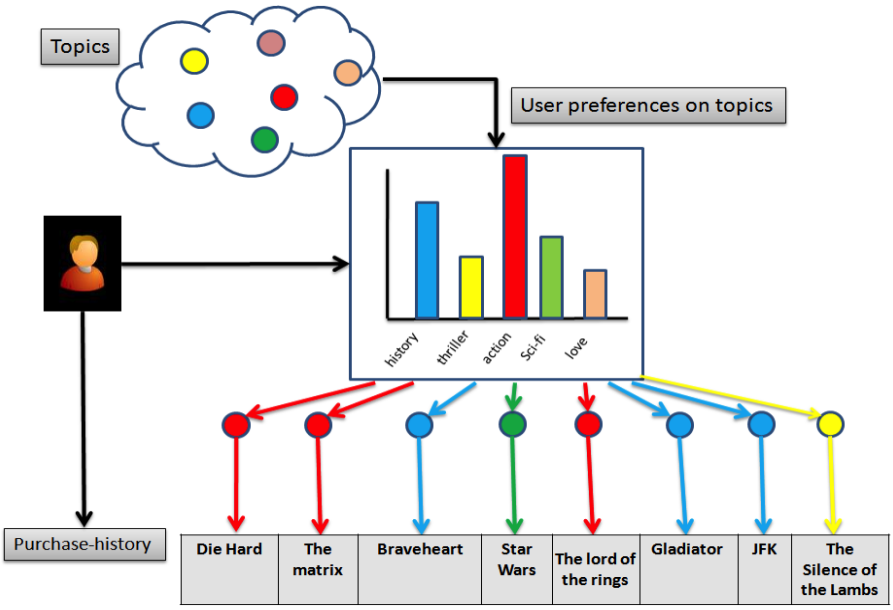

FIGURA 4: MODELLI PROBABILISTICI. ESEMPIO DI INTERAZIONE UTENTE - TOPIC.

L'utente esibisce una certa propensione verso alcuni topic, in questo caso, per comodità, sono associati dei generi cinematografici. Questa tendenza è indicata attraverso l'istogramma al centro: ogni barra dell'istogramma rappresenta la probabilità che l'utente possa scegliere un film da acquistare di un genere specifico. In generale si ha un istogramma per ogni utente u del sistema, che comodità sarà indicato con θ_u.

Parallelamente, si ha a disposizione un istogramma (distribuzione di probabilità) ϕ_k per ogni topic $k \in \{1, \dots, K\}$, che indica la probabilità dei vari item di appartenere al topic k.

La raccomandazione avviene secondo il seguente schema, che ricorda molto la matrix factorization:

FIGURA 5: MODELLI PROBABILISTICI. ESEMPIO DI INTERAZIONE DELL'UTENTE.

Alla domanda "Piacerà alla ragazza il film 'The Dark Knight'?" l'LDA risponde andando ad mettere in combinazione lineare tutti i topic, con il relativo grado di affinità con l'utente, pesato per l'attinenza del film al topic:

$$(i|u) = \sum_{k=1}^{K} \theta_{u,k} \cdot \phi_{k,i}$$

È da notare che nella sua versione base, l'LDA non restituisce un rating, bensì una probabilità di acquisto, il che implica che questo approccio punta a massimizzare la qualità della lista dei suggerimenti. I vantaggi dell'LDA rispetto ai modelli presentati finora sono:

- Buona qualità delle liste di suggerimenti
- Scalabilità
- Interpretabilità in termini probabilistici dei parametri
- Overfitting mitigato da un sistema di prior interne (nella fase di stima dei valori di θ e ϕ) che codifica l'eventuale conoscenza di backgroud del sistema

D'altro canto, porta con se molti difetti. Ad esempio:

- È molto sensibile all'inizializzazione dei parametri.
- Può soffrire di problemi di stabilità numerica, in presenza di alti gradi di sparsità della matrice delle preferenze.
- Si basa sulla Bag-of-Words assumption, ovvero non tiene conto della sequenza temporale delle preferenze.

L'attività di ricerca in quest'ambito è molto attiva ed ha condotto a numerose varianti e/o modelli affini che risolvono (in parte) questi problemi, tuttavia la ricerca è ancora aperta.

2.3. Approcci basati su Deep Learning per la raccomandazione

Come evidenziato nelle precedenti sezioni, i sistemi di raccomandazione sono utilizzati per stimare le preferenze degli utenti su item che non hanno ancora visionato. In generale abbiamo tre principali task nel campo della raccomandazione, basati sulla tipologia di output: predizione del rating, predizione della classificazione (raccomandazioni top-n) e classificazione. Il task di predizione del rating ha lo scopo di riempire le voci mancanti della matrice di valutazione degli item-user. Invece, la raccomandazione Top-n produce un elenco ordinato in base alla raccomandazione, con *n* item per ogni utente. Infine, l'attività di classificazione mira a classificare gli elementi candidati nelle categorie corrette per la raccomandazione.

Come visto in precedenza, i modelli di raccomandazione sono generalmente classificati in tre categorie: collaborative filtering, content based e hybrid recommender system. Il collaborative filtering formula raccomandazioni apprendendo dalle interazioni storiche utente-item, con feedback esplicito (ad es. valutazioni precedenti dell'utente) o feedback implicito (ad es. cronologie di navigazione). La raccomandazione content-based si basa principalmente sul confronto tra item e informazioni ausiliarie degli utenti. Gli approcci ibridi combinano il meglio dei precedenti. Si noti tuttavia che in un'ottica più generale è possibile prendere in considerazione una vasta gamma di informazioni ausiliarie come testi, immagini e video per migliorare ulteriormente la capacità predittiva del modello.

Nel seguito verranno prese in esame diverse architetture di DL learning evidenziando il task affrontato ed il tipo di approccio adottato.

Prima di proseguire introduciamo la notazione che verrà impiegata in questa sezione. Si supponga di avere M utenti e N item, e che R denoti la matrice di valutazione. Utilizziamo un vettore $r^{(u)} = \{r^{u1}, \ldots r^{uN}\}$ per rappresentare ogni utente u, e un vettore $r^{(i)} = \{r^{1M}, \ldots r^{Mi}\}$ per rappresentare ogni item i. Il successivo elenco sintetizza le notazioni e le corrispondenti descrizioni che sono usate nel seguito della presente trattazione.

- R: matrice dei rating
- M: numero di utenti
- r_{ui}: rating dell'item i dato dall'utente u
- $r^{(i)}$: vettore parzialmente osservato per l'item i
- U: fattore latente dell'utente
- $\mathcal{O}$, $\mathcal{O}^-$: Insiemi dei rating osservato o non osservato
- W_*: Matrici dei pesi per rete neurale
- $\hat{R}$: Matrice dei rating predetti
- N: Numero di item
- $\hat{r}_{ui}$: Rating predetto dell'item *i*, dato dall'utente *u*
- $r^{(u)}$: Vettore parzialmente osservato per l'utente *u*
- *V:* Vettore dei fattori latenti
- *K:* Il massimo valore di rating (esplicito)
- b_*: i termini Bias per le reti neurali

2.3.1. Deep Learning

I modelli tradizionali di machine learning consentono di gestire dati strutturati in maniera efficace e sono stati ampiamente utilizzati dalle aziende per il credit scoring, la previsione del tasso di abbandono, il targeting dei consumatori e così via. Il successo di questi modelli dipende in larga misura dalle prestazioni della fase di ingegnerizzazione delle feature: più si lavora vicino all'azienda per estrarre la conoscenza rilevante dai dati strutturati, più tipicamente efficace sarà il modello.

Il deep learning è un campo del machine learning basato sull'apprendimento di diversi layer di rappresentazione, tipicamente utilizzando reti neurali artificiali. Attraverso la gerarchia dei livelli di un modello di deep learning, i concetti di livello superiore sono definiti dai concetti di livello inferiore [Den14].

In Hinton et al. [Hin06] è stato introdotto un modo efficiente di addestrare deep models, e in Bengio [Ben09] viene dimostrata le capacità di architetture deep in task complicati di intelligenza artificiale. Attualmente, gli approcci di deep learning rappresentano soluzioni all'avanguardia per molti problemi nei campi della computer vision, natural language processing nel riconoscimento vocale [Den14] e text mining.

Sebbene le reti neurali e l'intuizione alla base dei modelli deep non siano recenti, l'ascesa degli approcci e dei modelli basati su architetture deep è avvenuta solo nell'ultimo decennio. I principali fattori che hanno promosso l'ascesa del deep learning come tecnica machine learning all'avanguardia possono essere riassunti in:

- ***Big data***. Un modello di deep learning è in grado di apprendere rappresentazioni migliori sfruttando enormi moli di dati, caratteristica principe dei Big Data.
- ***Potenza computazionale***. Le Graphical Processing Units (GPU) di ultima generazione soddisfano le necessità di potenza di calcolo richieste per calcoli complessi nei modelli di deep learning.

2.3.1.1. Rete neurale

L'elemento costitutivo fondamentale del Deep Learning è il Perceptron, che rappresenta il singolo neurone di una rete neurale.

Dato un insieme finito di m input (ad es. m parole o m pixel), moltiplichiamo ogni input per un peso ($\theta_1,\ldots,\theta_m$), quindi sommiamo la combinazione pesata degli input, aggiungiamo un bias e infine li passiamo attraverso una funzione di attivazione non lineare.

Questo produce l'output $\hat{y}$.

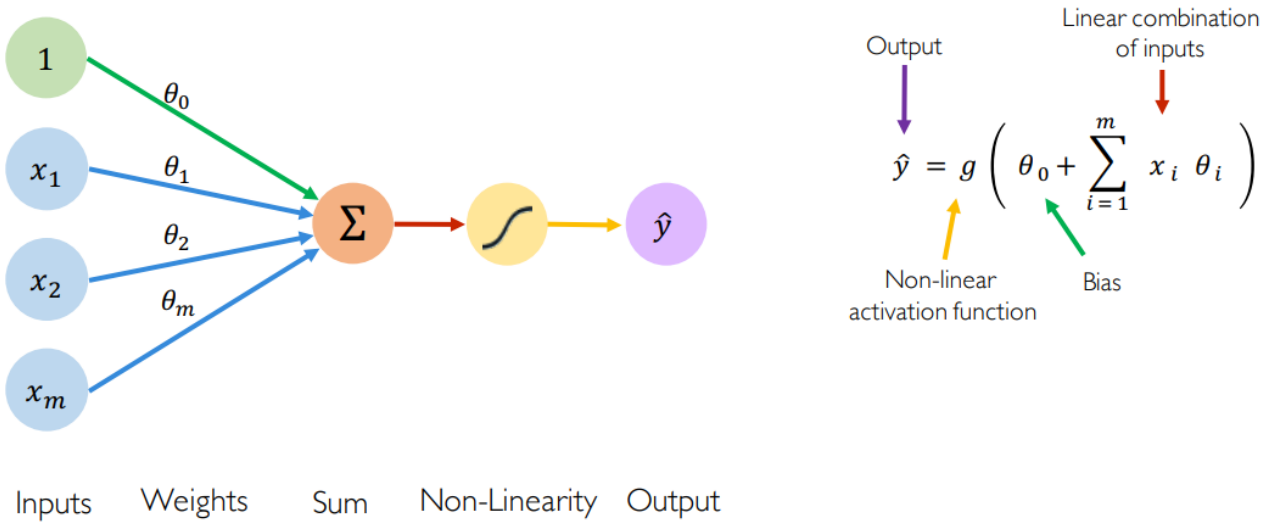

FIGURA 7: MECCANISMO DI APPRENDIMENTO DI UNA RETE NEURALE

- Il bias *theta* θ consente di aggiungere un'altra dimensione allo spazio degli input. Quindi, la funzione di attivazione fornisce ancora un output in caso vi sia un vettore degli input con tutti zero.
- Lo scopo delle funzioni di attivazione è quello di introdurre non-linearità nella rete. Infatti, le funzioni di attivazione lineare producono decisioni lineari, indipendentemente dalla distribuzione degli input. La non linearità consente di approssimare meglio funzioni arbitrariamente complesse. Di seguito, alcuni esempi di funzioni di attivazione:

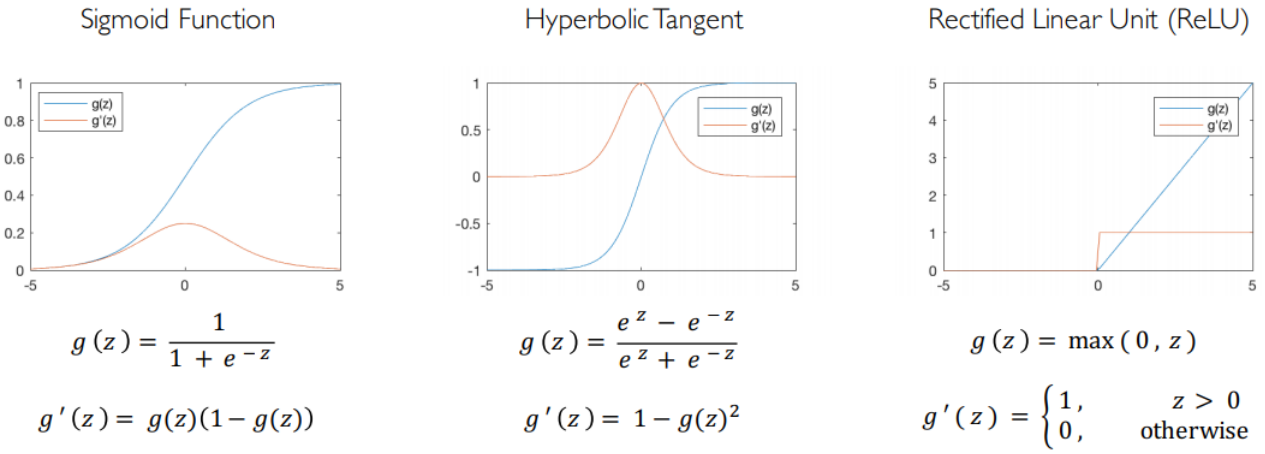

FIGURA 8: PRINCIPALI FUNZIONI DI ATTIVAZIONI

In buona sostanza una deep neural network non è altro che una rete neurale nella quale sono impilati molteplici hidden layer al fine di modellare funzioni complesse.

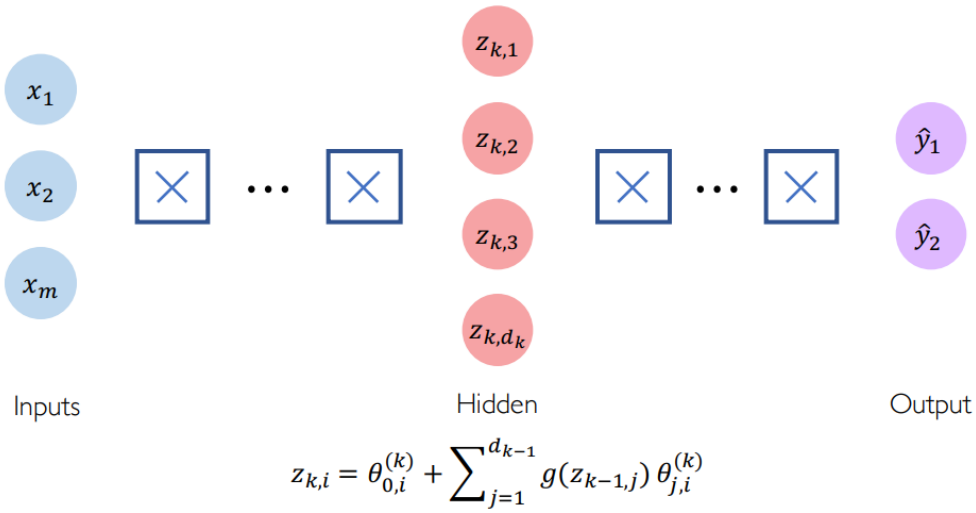

FIGURA 9: SCHEMA DI UNA RETE NEURALE

2.3.1.2. Architetture di Deep Learning

Il Deep Learning è sottoramo di ricerca dell'apprendimento automatico. Apprende più livelli di rappresentazioni ed astrazioni dai dati, che possono risolvere task di apprendimento supervisionati e non supervisionati. Nell'elenco di seguito sono presentate le principali architetture proposte in letteratura e loro varianti:

- Multilayer Perceptron (MLP) è una rete neurale feedforward con più (uno o più) livelli nascosti tra il layer di input e il layer di output. In questo caso, il perceptron può impiegare una funzione di attivazione arbitraria e non rappresenta necessariamente un classificatore strettamente binario.
- Autoencoder (AE) è un modello non supervisionato che tenta di ricostruire i suoi dati di input nel layer di output. In generale, il layer di compressione nascosto (il livello più intermedio) viene utilizzato come rappresentazione delle caratteristiche salienti dei dati di input. Sono state proposte in letteratura diverse varianti di autoencoder, tra cui: autoencoder denoising, autoencoder denoising marginalizzato, autoencoder sparso, autoencoder contrattivo e autoencoder variazionale (VAE).
- Convolutional Neural Network (CNN) è un tipo particolare di rete neurale feedforward con strati convoluzionali e operazioni di pooling. È in grado di catturare le caratteristiche globali e locali e di migliorare significativamente l'efficienza e la precisione. Rappresenta una scelta ideale quando sia necessario elaborare dati matriciali, come ad esempio immagini.
- Recurrent Neural Network (RNN) è un'architettura con consente di modellare dati sequenziali. A differenza della rete neurale feedforward, ci sono loop e "memorie" per tenere traccia degli stati precedenti. Varianti come le LSTM (Long Short Term Memory) e le Gated Recurrent Unit (GRU) sono spesso utilizzate nella pratica per superare il problema del gradiente evanescente.
- Deep Semantic Similarity Model (DSSM), o più specificamente, Deep Structured Semantic Model, è una deep neural network che mira ad apprendere rappresentazioni semantiche di entità in uno spazio semantico continuo comune e misurare le loro somiglianze semantiche.

- Una Restricted Boltzmann Machine (RBM) è una rete neurale a due layer composta da un layer visibile e uno nascosto. Può essere facilmente impilato su un deep network. Restricted qui significa che non ci sono comunicazioni intra-layer nel layer visibile o nel layer nascosto.
- Una Neural Autoregressive Distribution Estimation (NADE) è una rete neurale non controllata costruita su un modello autoregressivo e una rete neurale feedforward. È uno stimatore trattabile ed efficiente per modellare distribuzioni e densità dei dati.
- La Generative Adversarial Network (GAN) è una rete neurale generativa che consiste in un discriminatore e un generatore. Le due reti neurali vengono addestrate simultaneamente competendo tra loro in un framework di gioco minimax.

2.3.2. Metodi e modelli di Deep Learning per la raccomandazione

In questa sezione, presentiamo brevemente i modelli di deep learning che sono stati maggiormente impiegati nelle attività di raccomandazione. L'applicazione delle tecniche di deep learning al dominio dei sistemi di raccomandazione è di grande rilevanza e tendenza. Il deep learning è utile per analizzare i dati da più fonti e scoprire caratteristiche nascoste. Poiché la capacità di elaborazione dei dati delle tecniche di deep learning è in aumento a causa dei progressi nelle tecnologie per big data e nei supercomputer, i ricercatori hanno già iniziato a trarre vantaggio dalle tecniche di deep learning nei sistemi di raccomandazione.

In particolare, sono state utilizzate tecniche di deep learning per produrre soluzioni efficaci per alcune delle problematiche note nell'ambito dei sistemi di raccomandazione, come la scalabilità e la scarsità di dati. Inoltre, nel campo dei recommender systems il deep learning è stato utilizzato altresì per la riduzione di dimensionalità e per l'estrazione di feature rilevanti da diverse sorgenti dati.

Le tecniche di deep learning sono utilizzate nei sistemi di raccomandazione per modellare la matrice delle preferenze user-item o le informazioni content/side, e talvolta entrambe. Di seguito, si analizzano come e per quale scopo architetture di DL sono state utilizzate nei sistemi di raccomandazione. Le architetture di riferimento descritte in questa sezione includono RBM, DBN, autoencoder, RNN e CNN. Infine, sono introdotti alcuni modelli di frontiera in questo contesto.

2.3.2.1. Multilayer Perceptron based Recommender System

Il Multilayer Perceptron è un modello conciso ma efficace. Da ciò deriva il fatto che esso è utilizzato in numerose aree, in particolare in quella industriale. È stato mostrato come queste reti siano capaci di approssimare ogni funzione misurabile con ogni livello desiderabile di accuratezza. È la base di molti modelli avanzati.

2.3.2.2. Neural Collaborative Filtering

Nella maggior parte dei casi, la raccomandazione è considerata un'interazione a due vie tra le preferenze degli utenti e le caratteristiche degli item. Ad esempio, la fattorizzazione di matrice scompone la matrice di rating in uno spazio utente latente a bassa dimensionalità e in uno spazio latente degli item a bassa dimensione.

2.3.2.3. Content-based recommender

Il sistema genera elenchi di suggerimenti basati sulle somiglianze tra i profili utente e le caratteristiche degli elementi.
Quindi, è naturale costruire un doppio network per modellare l'interazione a due vie tra utenti e oggetti. Il Neural Collaborative Filtering (NCF) è un framework di questo tipo che mira a catturare la relazione non lineare tra utenti e oggetti. La seguente figura (a) presenta l'architettura NCF.

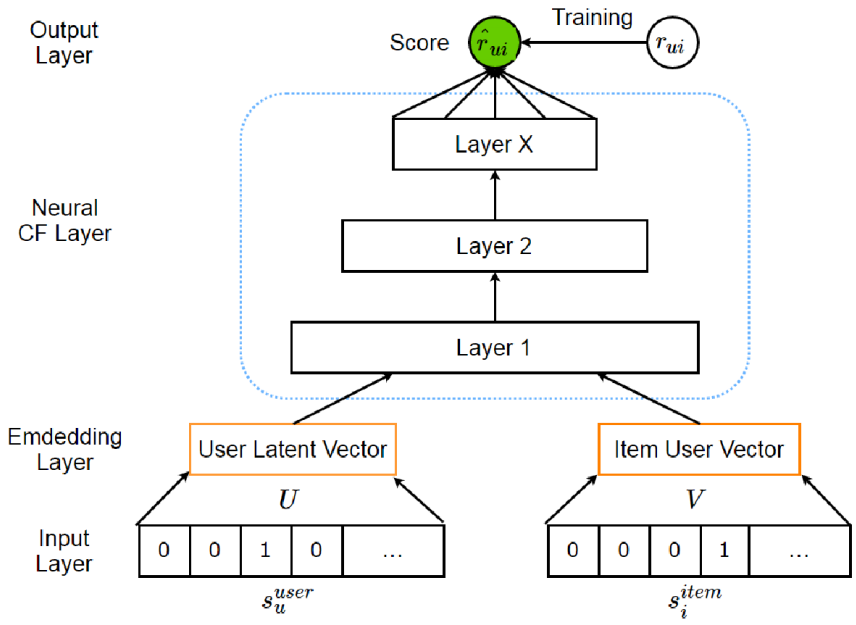

FIGURA 10: NEURAL COLLABORATION FILTERING

Si assuma che s_u^{user} e s_i^{item} denotino le informazioni secondarie (ad esempio, i profili utente e le caratteristiche degli articoli), o solo un identificativo one-shot dell'utente e dell'item rispettivamente. La regola di previsione di NCF è formulata come segue:

$$\hat{r}_{ui} = f(U^T \cdot s_u^{user}, V^T \cdot s_i^{item} | U, V, \theta)$$

Dove la funzione f(.) denota il multilayer perceptron, e θ è il parametro di questo network. La funzione di loss per predire la valutazione esplicita è definita come errore quadrato pesato:

$$\mathcal{L} = \sum_{(u,i) \in \mathcal{O} \cup \mathcal{O}^-} w_{ui}(r_{ui} - \hat{r}_{ui})^2$$

Dove w_{ui} denota il peso della istanza di training (u,i). Per le valutazioni binarie o il feedback implicito (ad es. 1 rappresenta un like e 0 rappresenta antipatia) gli autori hanno proposto un approccio

probabilistico (es. Funzione logistica o Probit) per limitare l'output r_{ui} nell'intervallo [0,1] e hanno rivisto la loss un una forma di entropia incrociata:

$$\mathcal{L} = -\sum_{(u,i)\in\mathcal{O}\cup\mathcal{O}^-} r_{ui}\log\hat{r}_{ui} + (1-r_{ui})\log(1-\hat{r}_{ui})$$

Poiché esiste un gran numero di istanze non osservate, NCF utilizza il campionamento negativo per ridurre la dimensione dei sempling data, che migliora notevolmente l'efficienza di apprendimento. La fattorizzazione di matrice tradizionale può essere vista come un caso speciale di NCF. Pertanto, è conveniente fondere la fattorizzazione matriciale con NCF per formulare un modello più generale che fa uso sia della linearità di MF, sia della non linearità di MLP, per migliorare la qualità delle raccomandazioni.

In He et al. [Xia17] è stato esteso il modello NCF alle raccomandazioni sociali interdominio, vale a dire, raccomandando item agli utenti dei social network, ed è stato presentato un neural social collaborative ranking recommender system. Un'altra estensione è CCCFNet (Cross-domain Content-boosted Collaborative Filtering rete neurale), mostrato nella seguente figura.

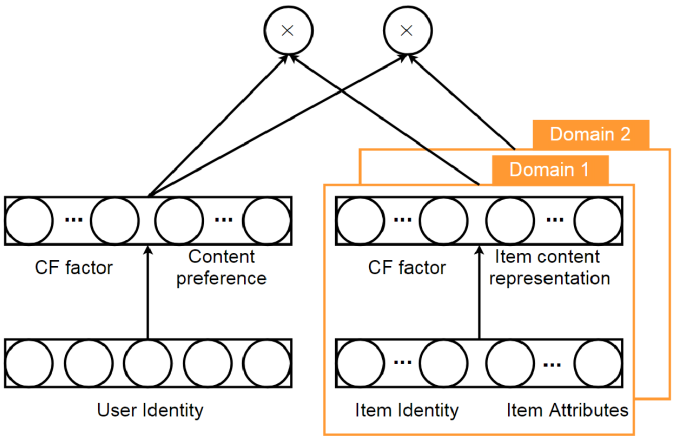

FIGURA 11: CCCFNET

Anche la componente di base di CCCFNet è una doppia rete (rispettivamente per utenti e item). Essa, modella le interazioni utente-item nell'ultimo layer tramite un prodotto scalare. Per incorporare le informazioni sul contenuto, gli autori hanno scomposto ulteriormente ciascuna rete (appartenente alla doppia rete) in due componenti:

- collaborative filtering factor (utente e item latente);
- informazioni sui contenuti (preferenze dell'utente sulle caratteristiche degli item).

Un multi-view neural framework è definito a partire dal modello base per eseguire raccomandazioni cross-domain.

2.3.2.4. Wide and Deep Learning

Questo modello è rappresentato nella seguente figura.

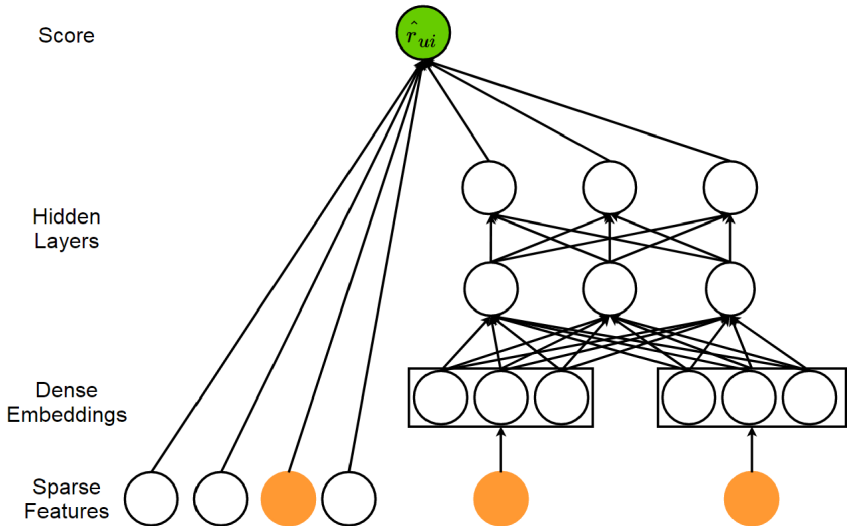

FIGURA 12: WIDE & DEEP LEARNING COMBINATION

Il modello è in grado di risolvere sia problemi di regressione che di classificazione, ma inizialmente è stato introdotto per la raccomandazione di app in Google Play. La componente di apprendimento *Wide* è composta da un singolo layer composto da diversi perceptron, il quale può anche essere visto come un modello lineare generalizzato; la componente di deep è una mlp multistrato. La logica della combinazione di queste due tecniche di apprendimento è consentire al sistema di raccomandazione di catturare sia la memorizzazione che la generalizzazione. La memorizzazione ottenuta dalla componente di apprendimento wide rappresenta la capacità di cogliere le caratteristiche dirette dai dati storici. Mentre, la componente di deep cattura la generalizzazione producendo rappresentazioni più generali e astratte.
Questo modello può migliorare la precisione e la diversità delle raccomandazioni.

Deep Factorization Machine

Una DeepFM viene rappresentata nella seguente figura.

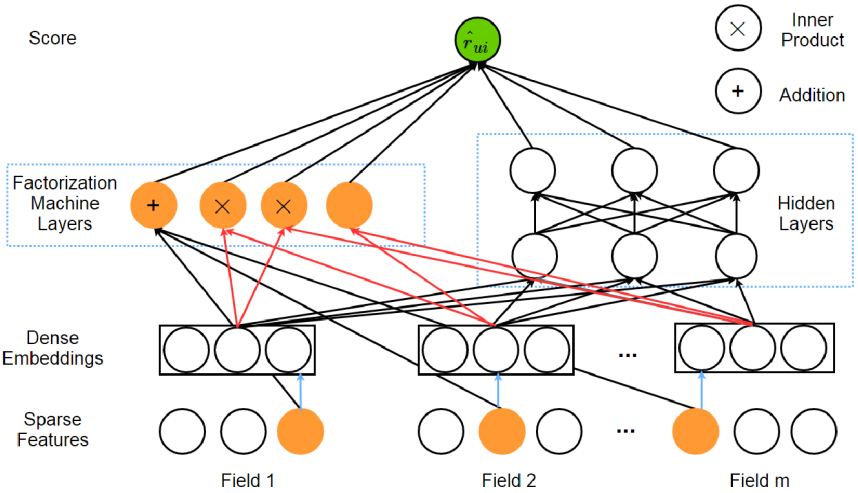

FIGURA 13 DEEPFM

DeepFM è un modello end-to-end che può anche essere visto come una generalizzazione che integra perfettamente la factorization machine e MLP. È in grado di modellare le interazioni delle caratteristiche di ordine elevato tramite un deep neural network e le interazioni di basso ordine tramite la factorization machine. La factorization machine utilizza operazioni di addizione e di prodotto interno.
MLP sfrutta le attivazioni non lineari e la struttura profonda per modellare le interazioni di ordine elevato. Il modo di combinare questi due modelli è evidenziato da un wide & deep network. Rispetto al modello wide & deep, DeepFM non richiede l'ingegnerizzazione delle funzionalità. L'input di DeepFM è costituito da dati con *m* campi che sono costituiti da coppie *(u, i)* (l'identità e le caratatteristiche di user e item). Per semplicità, l'output di FM e di MLP sono denotati come yFM(x) e yMLP(x) rispettivamente. Lo score di predizione è calcolato da:

$$\hat{r}_{ui} = \sigma(y_{FM}(x) + y_{MLP}(x))$$

Dove σ(.) è l'attivazione sigmoidea.

2.3.2.5. Attentive Collaborative Filtering

Il meccanismo di attenzione è ispirato dall'attenzione visiva umana. Ad esempio, le persone devono concentrarsi solo su parti specifiche degli input visivi per comprenderli o riconoscerli. Il meccanismo di attenzione è in grado di filtrare le caratteristiche non informative dagli input grezzi e ridurre gli effetti collaterali dei dati rumorosi. Il modello basato sull'attenzione ha mostrato risultati promettenti su attività come il riconoscimento vocale e la traduzione automatica.
Alcuni lavori recenti [Che17, Gon16, Seo17] dimostrano la sua capacità di migliorare le prestazioni delle raccomandazioni, incorporandole nelle tecniche di deep learning (ad esempio MLP, CNN e RNN) per i modelli di raccomandazione. Chen et al. [Che17] hanno proposto un modello di collaborative filtering di attenzione, introducendo un meccanismo di attenzione a due livelli al modello a fattori latenti. il modello dell'attenzione è un MLP che consiste nell'attenzione a livello di elemento e a livello di componente.
L'attenzione a livello di item viene utilizzata per selezionare gli item più rappresentativi per caratterizzare gli utenti. L'attenzione a livello di componente mira a catturare le caratteristiche più informative dalle informazioni ausiliarie multimediali per ogni utente.

2.3.3. Autoencoder-based Recommender Systems

In letteratura gli approcci più comuni per l'impiego di autoencoder nei sistemi di raccomandazione sono:

1. utilizzare l'autoencoder per apprendere rappresentazioni a bassa dimensionalità dalle features di partenza;
2. "riempire" i valori mancanti della matrice di rating direttamente nel layer di ricostruzione.

Un semplice autoencoder comprime i dati forniti la sotto-rete encoder, e ricostruisce i dati compressi per mezzo del decoder. Gli autoencoder provano a ricostruire i dati iniziali cercando di riprodurre i dati in ingresso alla rete. Questi modelli sono stati usati nei sistemi di raccomandazione per apprendere una rappresentazione non lineare della matrice user-item, e ricostruirla determinando i valori mancanti [Oiy14, Sed15].
Gli autoencoder sono anche impiegati per la riduzione di dimensionalità e l'estrazione delle feature in vettori latenti, usando i valori di output dell'autoencoder [Den17, Zuo16, Ung16].

Inoltre, lo sparse coding è applicato agli (sparse) autoencoder per apprendere feature più efficaci [Zuo16]. A differenza degli autoencoder sparsi incompleti che limitano la rappresentazione, i Denoising Autoencoder (DAE) consentono di estrarre una buona rappresentazione modificando il criterio di ricostruzione. In pratica, i DAE prendono un input parzialmente danneggiato e vengono addestrati in modo tale da ricostruire l'input originale non distorto. In pratica, l'obiettivo di questi modelli è quello di ripulire l'input corrotto.

Gli SDAE sono costituiti da molti autoencoder che formano uno stack. Tali autoencoder consentono di estrarre molte feature nascoste. I denoising autoencoder (DAE) sono stati altresì usati nei sistemi di raccomandazione per predire i valori mancanti da dati corrotti [Wu16], e gli SDAE supportano i sistemi di raccomandazione nella scoperta di una forma più densa della matrice di input [Str15].

Inoltre, sono utili per integrare e fornire informazioni ausiliarie, consentendo la raccolta di dati da più fonti [Wan15b, Str16].

Gli studi in questo campo attestano che gli autoencoder dimostrano la capacità di questi modelli di fornire raccomandazioni più accurate rispetto a quelle fornite dalle RBM. Una delle ragioni di ciò è che le RBM producono predizioni massimizzando la loglikelihood, mentre gli autocencoder le producono minimizzando l'RMSE. Inoltre, la fase di training degli autoencoder è più veloce di quella degli RBM, a causa dei metodi usati, come la backpropagation basata su gradiente per gli autoencoder e la divergenza contrastiva per gli RBM [Sed15]. Gli auteconder appartenti a stack generano previsioni più accurate di quelli che non appartengono a stack, in quanto i primi consentono di apprendere con maggiore profondità feature nascoste [Li15].

2.3.3.1. AutoRec

AutoRec utilizza i vettori parziali dell'utente $r^{(u)}$ o i vettori parziali dell'elemento $r^{(i)}$ come input, e mira a ricostruirli nel layer di output. Apparentemente, ha due varianti: Item-based AutoRec (I-AutoRec) and User-based AutoRec (U-AutoRec), corrispondenti ai due tipi di input. Qui, introduciamo solo I-

AutoRec, mentre U-AutoRec può essere facilmente derivato di conseguenza. La seguente figura illustra la struttura di I-AutoRec.

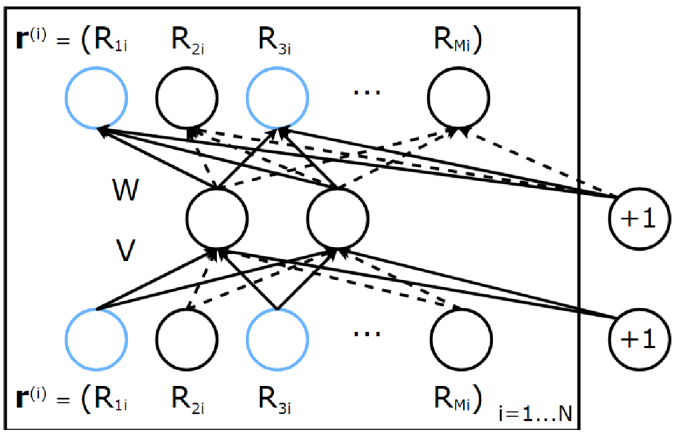

FIGURA 14: AUTOREC

Dato un input $r^{(i)}$, la riconstruzione è $h(r^{(i)}; \theta) = f(W \cdot g(V \cdot r^{(i)} + \mu) + b)$,dove f(.) e g(.) sono le funzioni di attivazione, il paramentro $\theta = \{W, V, \mu, b\}$. La funzione obiettivo di I-AutoRec è formulata come segue:

$$\arg\min_{\theta} \sum_{i=1}^{N} \| \mathbf{r}^{(i)} - h(\mathbf{r}^{(i)}; \theta) \|_{\mathcal{O}}^2 + \lambda \cdot Regularization$$

Qui $\| \cdot \|_{\theta}^2$ sta a significare che vengono presi in considerazione solo i rating osservati. La funzione obiettivo può essere ottimizzata mediante propagazione resiliente (converge più velocemente e produce risultati comparabili) o L-BFGS (algoritmo Broyden Fletcher Goldfarb Shanno a memoria limitata). Ci sono quattro punti importanti su AutoRec che vale la pena notare prima del deployment:

- I-AutoRec esibisce prestazioni migliori rispetto ad U-AutoRec dovute alla maggiore varianza dei vettori parziali osservati dall'utente
- Diverse combinazioni di funzioni di attivazione f(.) e g(.) influenzeranno notevolmente le prestazioni
- L'aumento moderato della dimensione dell'unità nascosta migliorerà il risultato poiché l'espansione della dimensionalità del livello nascosto conferisce ad AutoRec una maggiore capacità di modellare le caratteristiche dell'input
- L'aggiunta di più livelli per formulare una rete profonda migliora ulteriormente le prestazioni

2.3.4. Restricted Boltzmann machines per la raccomandazione

Le RBM sono particolari tipi di Boltzmann Machine. Hanno due tipi di layer, uno strato softmax visibile e uno strato invisibile. In una RBM non esiste la comunicazione tra layer. Esse sono usate per estrarre

feature latenti relative alle preferenze degli utenti o ai rating degli item nel dominio della raccomandazione.

Le RBM vengono utilizzate anche per la modellazione congiunta di entrambe le correlazioni tra gli elementi votati da un utente e la correlazione tra gli utenti che hanno votato un particolare elemento, per migliorare l'accuratezza di un sistema di raccomandazione [Geo13]. Le RBM sono anche utilizzate nei sistemi di raccomandazione group-based, per modellare le preferenze di gruppo modellando congiuntamente caratteristiche collettive e profili di gruppo[Hu14].

In [Tru09] le macchine di Boltzmann sono utilizzate per estrarre sia la relazione tra un elemento valutato e il suo rating, sia le correlazioni tra gli elementi valutati, grazie alle connessioni tra lo strato hidden e lo strato softmax, e le connessioni tra la i layer softmax, rispettivamente.

Le Boltzmann Machine sono utilizzate nei sistemi di raccomandazione per modellare la correlazione a coppie tra item o utenti. Inoltre, [Gun08] impiegano macchine di Boltzmann non solo per modellare la correlazione tra utenti e item, ma anche per integrare le informazioni sui contenuti. In particolare, vengono correlati i parametri delle macchine alle informazioni sui contenuti.

Le RBM vengono utilizzati principalmente per fornire una rappresentazione low-rank delle preferenze dell'utente. D'altra parte, le Boltzmann Machine sono utilizzate per integrare le correlazioni tra le coppie di utenti o le coppie di item, e la formazione di "vicinati" all'interno di strati visibili. È possibile combinare le correlazioni utente-utente e item-item tramite RBM generando un modello ibrido in cui i livelli nascosti sono collegati a due livelli visibili (uno per gli item e uno per gli utenti).

Infine, sia gli RBM che le Boltzmann Machines consentono l'integrazione di informazioni ausiliarie da diverse fonti di dati.

2.3.5. Approcci di raccomandazione basati su deep belief networks

Una delle applicazioni dei DBN (Deep belief networks) nel dominio dei sistemi di raccomandazione è relativa all'estrazione di feature nascoste e utili dal contenuto audio, per la raccomandazione di musica basata sul contenuto [Wan14].

Le DBN sono anche impiegate nei sistemi di raccomandazione basati su dati testuali [Kyo14, Zao15]. Inoltre, le DBN vengono usate nei sistemi di raccomandazione content-based basati su dati testuali come classificatori [Kyo14].

Anche la rappresentazione semantica delle parole è ottenuta utilizzando DBN [Zha15]. Inoltre, le DBN consentono di estrarre feature di alto livello da feature di basso livello dalle preferenze dell'utente [Hu14]. Questi studi mostrano che le DBN sono particolarmente adatte per la fase di feature extraction, principalmente da dati testuali e audio.

2.3.6. Recurrent Neural Networks per la raccomandazione

Le RNN (Recurrent Neural Networks) sono specializzate nell'elaborazione di sequenze di informazioni. In un sistema di e-commerce, la browsing history corrente di un utente influenza i suoi comportamenti negli acquisti. Comunque, la maggior parte dei sistemi di raccomandazione creano le preferenze dell'utente all'inizio di una sessione, il che si traduce nel trascurare la cronologia corrente e l'ordine nelle sequenze delle azioni dell'utente. Le RNN sono usate nei sistemi di raccomandazione per integrare la cronologia di visualizzazione corrente della pagina Web e l'ordine delle visualizzazioni, per fornire raccomandazioni più accurate [Wu16a].

In [Wu16a] vengono uniti i risultati di un RNN con il risultato di una rete neurale feed-forward, al fine di considerare le correlazioni user-item nella produzione di predizioni. In [Ko16] le reti neurali ricorrenti vengono utilizzate per rappresentare aspetti temporali e contestuali dei comportamenti degli utenti, al fine di migliorare la qualità delle predizioni, combinando tali rappresentazioni con i fattori di latenza relativi alle preferenze degli utenti.

[Dev17] utilizzano le RNN al fine di integrare l'evoluzione dei gusti degli utenti nel processo di raccomandazione, considerando il problema come una sequenza di problemi di predizione. Si noti infine che le RNN rappresentano scelte efficace specialmente nell'ambito dei sistemi di raccomandazione session-based e nell'integrazione dei comportamenti impliciti degli utenti nella modellazione delle loro preferenze.

2.3.7. Utilizzo delle reti Convoluzionali nel processo di raccomandazione

Una rete neurale convoluzionale usa un livello convoluzionale in almeno un layer che la compone, e viene impiegata principalmente in task, come la classificazione di oggetti e il riconoscimento di immagini. Anche i sistemi di raccomandazione possono trarre beneficio dall'adozione delle reti neurali convoluzionali.
In [Oor16] sono state utilizzate per estrarre fattori latenti dai dati relativi agli audio, quando tali fattori non possono essere ottenuti dai feedback degli utenti. In [she16] le reti neurali convoluzionali sono usate per estrarre fattori latenti da dati testuali. In [Zho16] le reti neurali convoluzionali sono state impiegate per estrarre feature visuali, al fine di generare profili utenti, per produrre successivamente raccomandazioni.

In [Lei16] le CNN sono usate per estrarre feature latenti dalle immagini, al fine di ottenere un mapping tra le feature e le preferenze dell'utente nello stesso spazio latente. Il significato semantico delle informazioni testuali estratte per mezzo delle reti neurali convoluzionali viene utilizzato anche nei sistemi di raccomandazione, specialmente nei sistemi di raccomandazione context-aware, per fornire raccomandazioni maggiormente qualificate [Wu17b].

Ciò ha come conseguenza il fatto che le reti neurali convoluzionali sono usate maggiormente per l'estrazione di fattori latenti e feature dai dati, specialmente da immagini e testo, allo scopo di generare raccomandazioni. In ultimo una linea di ricerca emergente riguarda l'impiego di CNN per analizzare la sequenza dei prodotti valutati o visionati dall'utente.

3. Text mining e Natural Language Processing per l'analisi del testo e del parlato

L'obiettivo di questa sezione è offrire una panoramica sulle tecniche che possono essere di supporto alla realizzazione di chatbot e risponditori vocali. In altri termini l'analisi sarà orientata alle tecniche che possono trovare impiego nell'ambito del riconoscimento conversazionale (sia parlato che testuale). Due i macro scenari di utilizzo.
Nel primo, l'utente interagisce a voce con un risponditore automatico, e quest'ultimo lo supporta nella ricerca dei contenuti. Si presentano quindi due problematiche. La prima, si riferisce alla comprensione delle singole parole, ovvero della trasposizione del parlato in scritto. Tale problematica è stata già ampiamente affrontata e risolta dal punto di vista tecnologico, e su di essa, quindi, non si è inteso investigare oltre.
La seconda tematica è relativa alla comprensione di frasi in linguaggio naturale, ed è condivisa con il secondo macro scenario di seguito presentato. In tale scenario, l'utente interagisce, tramite chat, con un bot (da qui il termine chatbot), può porre a quest'ultimo domande ed essere assistito nella ricerca di contenuti. Da quanto detto è possibile concludere che, fatto salvo uno step preliminare le cui funzione è quella di tradurre l'audio relativo al parlato in testo scritto (che, come detto, non risulta in questa sede di interesse), si tratti del medesimo problema, che quindi, dati tali scenari applicativi, può essere sintetizzato come la comprensione del significato di frasi o documenti in linguaggio naturale. Tale problema viene tipicamente affrontato grazie all'impiego di tecniche di text mining e le tecniche di NLP, che saranno quindi oggetto del seguito della presente trattazione.

3.1. Knowledge Discovery in Text

Tecniche, metodi ed approcci per l'analisi automatica e semi-automatica del testo hanno trovato largo impiego in diversi scenari applicativi quali analisi di frodi, studio del comportamento dell'utente in reti sociali, classificazione documentale, etc.

Questi approcci sono in genere utilizzati per raggruppare informazione non strutturata o semi-strutturata in base all'argomento, per cui è naturale pensare di impiegare tali tecniche nell'ambito della classificazione automatica dei ticket. L'apertura e la gestione di un ticket porta con sé diverse tipologie di informazioni testuali, come ad esempio la descrizione del problema riscontrato fornita dall'utente, le note dell'operatore durante la gestione dello stesso e la descrizione di come l'esperto abbia risolto il problema.

Da questo punto di vista, le tecniche di *Knowledge Discovery* e *Machine Learning* possono essere impiegate per apprendere modelli predittivi in grado di classificare automaticamente il ticket in base alla tipologia di problema e suggerire *best practices* per la sua rapida risoluzione. Il Knowledge Discovery in Text (KDT) è un insieme di tecniche che sono il risultato di una generalizzazione di quelle utilizzate nel Knowledge Discovery in Databases (KDD) a domini informativi disponibili in formato testuale, come ad esempio le descrizioni delle problematiche in uno ticket management system.

Il processo di KDT, definito compiutamente da Feldman nel 1996, è caratterizzato da una struttura, ormai definitivamente accettata dalla comunità scientifica, che è composta da 4 fasi: Document

Acquisition, Document Preprocessing, Text Mining, Result Interpretation and Refinement. Le descriviamo brevemente:

1. **Document Acquisition**. Questa fase viene realizzata attraverso tecniche ed algoritmi di crawling e wrapping ed ha l'obiettivo di acquisire collezioni di documenti di potenziale interesse di vario formato, provenienti da differenti sorgenti (Web, Intranet, Banche Dati testuali). I documenti acquisiti, solitamente ricondotti ad un formato standard, vengono memorizzati in un repository;
2. **Document Pre-Processing**. In questa fase, ogni documento viene analizzato al fine di estrarne le features che lo caratterizzano. In questo modo i documenti memorizzati nel repository assumono una forma "strutturata" dipendente dalla natura delle features estratte. La tipologia delle features dipende, generalmente, da due fattori principali: gli algoritmi di mining che si intende utilizzare per l'analisi, nonché tipologia e la forma della conoscenza che si intende estrarre. Gli estrattori di features si caratterizzano in ragione delle tecnologie di base che utilizzano, ad esempio, reti neurali, espressioni regolari, analisi statistiche, ecc., e dalla precisione e completezza che possono garantire al processo di estrazione. Contrariamente a quanto accade col KDD, nel KDT la fase di pre-processing assume importanza fondamentale, perché deve tenere in considerazione le problematiche correlate al linguaggio naturale;
3. **Text Mining**. Questa fase sarà ampiamente dettagliata nel seguito. Preliminarmente si può dire che con il termine Text Mining si indica un insieme di metodi, tecniche e strumenti destinati alla scoperta di regolarità all'interno di sorgenti informative semi o non strutturate (testuali). Vi è una forte correlazione con la fase precedente, che deve predisporre un insieme di informazioni adatte agli algoritmi di text mining, attraverso l'estrazione di features adeguate. Si può notare che, in presenza di una fase di pre-processing che sia in grado di garantire un buon grado di strutturazione al suo output, le tecniche adoperabili per il text mining non sono necessariamente distinguibili da quelle di data mining. Di conseguenza, algoritmi di classificazione, clustering e generazione di regole possono essere utilmente applicate alle versioni "strutturate" dei testi prodotte dalla fase di pre-processing. In ragione del livello di precisione di localizzazione determinato dagli algoritmi di pre-processing, le tecniche di text mining possono garantire un livello di analisi più o meno fine, essendo in grado di classificare solo interi documenti, ovvero singoli paragrafi o frasi di questi;
4. **Results Interpretation and Refinement**. In questa fase viene visualizzata la conoscenza estratta, che può presentarsi sotto varie forme (cluster di documenti con contenuti simili, liste di concetti contenuti nei documenti, associazioni tra documenti, trend temporali sui contenuti dei documenti ecc.). La visualizzazione può, anche, avvenire dopo processi di raffinamento compiuti attraverso apposite interfacce o moduli automatici in grado di mostrare i risultati finali secondo le esigenze dell'utente. È bene sottolineare come il ruolo del KDT non si riduca alla risoluzione del pur importante problema della selezione di documenti rilevanti ad una data esigenza informativa, ma possa potenzialmente svolgere un ruolo rilevante per affrontare la più generale problematica della gestione dei contenuti informativi all'interno delle organizzazioni.

3.1.1. Text Mining

Con il termine "Text Mining" si indica un'elaborazione automatizzata di un insieme di documenti testuali, volta ad effettuare una classificazione degli elementi che ne fanno parte in base all'argomento trattato o ad altre caratteristiche distintive, ed a comporre una descrizione dei raggruppamenti creati. Si tratta di un caso particolare di data mining, ma si discosta notevolmente dalla maggior parte delle tecniche che vengono generalmente indicate con questo termine. È facile comprendere l'utilità di un software in grado di svolgere questo tipo di compito se si pensa ad una situazione nella quale ci si trovi ad avere a che fare con enormi quantità di testo. Leggerlo tutto sarebbe improponibile, inoltre le informazioni interessanti contenute al suo interno sarebbero probabilmente non equamente distribuite, potrebbe esserci una lunga serie di documenti inutili per gli scopi fissati. Sfruttando il text mining, sarebbe possibile separare i documenti in base all'argomento trattato, così da potersi concentrare soltanto su quelli più significativi. In un altro scenario, potrebbe non interessare conoscere con precisione cosa sia scritto all'interno dei documenti coi quali si ha a che fare, si potrebbe aver semplicemente bisogno di un riassunto degli argomenti trattati. Ancora una volta, affidandosi al text mining sarebbe possibile ottenerne in maniera automatizzata una breve descrizione.

Lavorando con informazione non strutturata il text mining si presenta come un problema impegnativo in diversi campi applicativi. È un campo multidisciplinare, che impiega modelli e metodi mutuati, principalmente, dai seguenti settori:

- *Information Retrieval* (IR): L'Information Retrieval è l'insieme di tecniche per l'individuazione dei documenti rilevanti rispetto ad una determinata esigenza informativa dell'utente. Più precisamente tratta problemi e tecniche di rappresentazione, memorizzazione, organizzazione e accesso all'informazione non-strutturata. Solitamente col termine "Information Retrieval" si identifica la raccolta di testi tra quelli che ipotizziamo trattare lo stesso argomento, ma più genericamente possiamo intendere anche la semplice raccolta di informazioni testuali per una successiva analisi. I sistemi di IR devono essere in grado di interpretare il contenuto informativo dei documenti, selezionare quelli il cui contenuto riguarda l'area di interesse dell'utente e riordinarli secondo criteri di rilevanza. I documenti che un Information Retrieval System (IRS) tratta sono spesso in un linguaggio naturale, quindi non presentano nessuna struttura e utilizzano forme lessicali semanticamente ambigue. Per far fronte a questi problemi i sistemi di IR effettuano diverse operazioni che gli permettono di raggiungere l'obiettivo comune: restituire documenti rilevanti per una data query.
- *Natural Language Processing* (NLP): Il Natural language processing è un settore di ricerca integrato all'Intelligenza artificiale (IA) e alla linguistica computazionale che si occupa del trattamento automatico del linguaggio naturale. Si tratta di una serie di pratiche volte alla realizzazione di strumenti informatici per analizzare, produrre, comprendere, tradurre testi o corpora del linguaggio naturale. In generale i sistemi di NLP sono sistemi per calcolatori ideati affinché le macchine possano esibire comportamenti linguistici intelligenti. I compiti possono variare molto negli scopi e nelle ambizioni: dai programmi di correzione automatica, sintattica ed ortografica, all'analisi e traduzione automatica di testi, fino al language understanding (comprensione del linguaggio). Molte problematiche concernono questi sistemi, in particolare l'ambiguità e la polisemia intrinseca al linguaggio naturale che sono difficilmente trattabili da un calcolatore. La comprensione del linguaggio è ciò che desta più problemi, in quanto l'interpretazione delle frasi richiede una vasta e articolata conoscenza del mondo e del contesto coinvolto, con un ricco apparato di presuppozioni e credenze che costituiscono lo sfondo comune nel quale avviene ogni scambio comunicativo.

- *Information Extraction* (IE): Per "information extraction" si intende la creazione di una rappresentazione strutturata dell'informazione rilevante presente in un documento machine-readable non strutturato. Se applicata a documenti testuali, l'Information Extraction è dunque una tecnologia che punta ad estrarre elementi salienti relativi ad un particolare contesto, come entità o relazioni. Esso si distingue dall'IR in quanto l'enfasi in IR è trovare documenti che già contengono la risposta alla domanda formulata dell'utente: data una collezione di documenti, il sistema di IR che riceve in input una query (set di parole chiave) seleziona un sottoinsieme i documenti che ritiene rilevanti per la query. L'utente poi navigherà la lista di documenti e cercherà l'informazione che più gli interessa. Il sistema di IE, invece, data una selezione di documenti, cerca invece di estrarre in modo strutturato l'informazione rilevante secondo le esigenze fornite in input. IR e IE sono quindi tecnologie complementari che condividono lo stesso obiettivo finale: l'estrazione di informazione implicita contenuta in un insieme di documenti. Il processo relativo ad un sistema di Information Extraction si suddivide generalmente in due parti principali:
    - In primo luogo il sistema estrae fatti individuali dal documento attraverso un'analisi locale del testo;
    - Poi i fatti estratti vengono integrati con l'analisi di coreferenza e di inferenza.

Infine, dopo tale fase di integrazione, i fatti pertinenti vengono tradotti nel formato di output richiesto.

3.1.1.1. Text Classification

La *text classification* mira ad assegnare un valore booleano ad ogni coppia $\langle d_j, c_i \rangle \in D \times C$, dove D è il domino dei documenti e $C = \lfloor c_1, \ldots, c_{|c|} \rfloor$ è l'insieme delle categorie da assegnare. Un valore positivo per la coppia $\langle d_j, c_i \rangle$ indica che il documento d_j è stato assegnato alla categoria c_i, un valore negativo che quel documento non è appartiene a quella categoria. Più formalmente, l'obiettivo è quello di approssimare una funzione, a priori ignota, $\breve{\phi} : D \times C \to \{T, F\}$ (che descrive come i documenti dovrebbero essere classificati) con una funzione $\phi : D \times C \to \{T, F\}$ chiamata classificatore (o modello) in modo tale che $\breve{\phi}$ e ϕ "coincidano quanto più possibile".

Uno dei problemi che si affronta nella classificazione dei documenti è stabilire quante possibili classi la funzione di classificazione può assegnare ad un certo documento. In generale si può pensare di assegnare un numero di categorie k tale che $a \leq k \leq b$. Variando i valori dei coefficienti a e b si possono ottenere i seguenti diversi scenari:

- *Single-labelled* o non overlapping categories: i coefficienti a e b sono entrambi posti uguali ad 1. In questo modo si impone il fatto che ad ogni documento d_j del corpus sia assegnata una e una sola categoria tematica c_i.

- *Multi-labelled* o overlapping categories: i coefficienti a e b vengono scelti in modo tale che $a < b$. Così facendo imponiamo che più categorie possano essere assegnate ad uno stesso documento.
- *Binary-labelled*: è un caso speciale del caso Single-labelled, in cui un documento dj viene assegnato alla categoria ci oppure al suo complemento.

Dei tre casi appena descritti, quello binario può essere considerato il più generale in quanto un qualsiasi problema di classificazione di tipo multi-labelled sull'insieme di categorie $C = \{c_1, \ldots, c_{|c|}\}$ può essere trasformato in c problemi indipendenti di classificazione *binary-labelled* sulle categorie $\{c_i, \bar{c}_i\}$, per *i = 1, ..., c*.

Negli ultimi anni sono stati proposti diversi metodi di generazione di classificatori dei testi basati su tecniche di machine learning (ossia mediante l'uso di dati di training etichettati), fra cui classificatori k-nearest neighbor (k-NN) e metodi basati su modelli probabilistici, tra i quali classificatori Bayesiani, classificatori lineari, alberi di decisione, reti neurali, Support Vector Machines. In letteratura sono stati proposti interessanti approcci per la generazione di classificatori basati su regole. Questi presentano la desiderabile proprietà di essere leggibili e di agevole interpretazione, al contrario di altri approcci (come ad esempio SVM, reti neurali ecc.).

Tuttavia, i sistemi di TC esistenti presentano ancora importanti limitazioni di varia natura. In particolare:

- Molti approcci (classificatori Bayesiani, SVM, reti neurali) sono caratterizzati da un modello di tipo black box, ovvero non possono essere interpretati da un lettore umano.
- Sistemi tradizionali per l'apprendimento di regole di classificazione, quali RIPPER, generano classificatori che spesso includono diverse centinaia di regole, quindi di difficile interpretazione; inoltre, essi sono complessi e generalmente poco efficienti.
- I classificatori associativi applicati a problemi di TC (basati essenzialmente su estensioni del popolare algoritmo di rule mining "a priori"), risultano altresì "leggibili", ma la qualità delle regole generate, a differenza di quelle costruite per dati strutturati, risulta piuttosto bassa.

3.1.1.2. Rappresentazione e pre-processing di dati testuali

La disponibilità di grandi quantità di informazioni in formato testuale implica la necessità di disporre di soluzioni che consentano l'accesso efficace alle informazioni; in particolare, sono necessari strumenti per convertire tali sorgenti non strutturate in sorgenti strutturate, in modo da facilitarne la memorizzazione, l'accesso e la navigazione.

Nel campo del text mining gli input sono costituiti fondamentalmente da contenuti testuali, in genere provenienti da documenti di varia natura, e sono comunemente raggruppati in insiemi detti corpora. Un documento d_j, cioè un insieme di parole appartenenti ad un dizionario, può essere rappresentato come

un vettore le cui componenti indicano la presenza o l'assenza di una feature (es. parole, punteggiatura, stile del testo, struttura del testo, N-grammi) all'interno del documento. Più in generale un documento d_j viene solitamente rappresentato come un vettore di pesi $\vec{d}_j = \lfloor w_1, \ldots, w_{|T|j} \rfloor$ dove T è l'insieme dei termini che occorrono almeno una volta in almeno un documento del training set e $0 \leq w_{kj} \leq 1$ rappresenta quanto un termine t_k contribuisce alla semantica del documento d_j.

Questo tipo di rappresentazione prende il nome di Bag-Of-Word (BOW). Supponiamo ad esempio di avere due documenti:

- D_1 = "ingredienti pizza: farina, acqua, lievito, olio"
- D_2 = "descrizione computer: CPU, RAM, Hard disk"

Il dizionario sarà quindi l'unione dei due insiemi:

- T = {ingredienti, pizza, farina, acqua, lievito, olio, descrizione, computer, CPU, RAM, Hard Disk}
- n = 11 dimensione dello spazio

La rappresentazione BOW dei due documenti è quindi D_1(1,1,1,1,1,1,0,0,0,0,0), D_2 (0,0,0,0,0,0,1,1,1,1,1).

Si tratta di una rappresentazione molto semplice e veloce da realizzare, con bassa complessità computazionale e che esibisce buoni risultati in fase classificazione. Tuttavia proprio per la sua semplicità mostra dei limiti, in primo luogo parole uguali che assumono nel documento significati differenti sono trattate come la stessa parola, non tiene conto dell'ordine delle parole, infine si tratta di una rappresentazione molto "rumorosa". Col termine rumore si intende qualunque cosa che possa disturbare il buon comportamento del sistema. Nel caso in esame il rumore si presenta come parole poco informative sul topic del documento (articoli, congiunzioni, avverbi), parole diverse con significati simili (sinonimi), parole uguali con significati diversi (es. àncora e ancòra) e verbi coniugati (vado e andare).

Per limitare alcuni di questi problemi sono stati studiati metodi di pre-processing quali:

- Tokenization;
- Conversion to lowercase;
- Lemmization e Stemming;
- Part of speech (POS) tagging;
- Eliminazione delle stop-word;
- Generazione degli N-Grammi;
- Feature Selection.

Tali fasi saranno descritte in dettaglio nei paragrafi successivi.

Tokenization

Il primo passo da compiere nel manipolare il testo, consiste nel segmentare l'insieme di caratteri in word o più precisamente in token. I token sono separati l'uno dall'altro da appositi caratteri, i separatori. Il processo di tokenization è banale per un umano ma può divenire anche molto complesso per un calcolatore. Il problema risiede nel fatto che determinati caratteri possono essere Token Delimiters oppure no in dipendenza della specifica applicazione considerata. Caratteri come Spazio, Tab, Newline vengono assunti essere sempre e comunque Token Delimiters. Per altri caratteri la situazione è differente e in molti casi controversa.

Caratteri come: *( ) < > ! ? "* sono sicuramente Token Delimiters ma occasionalmente possono essere Tokens. A seconda del contesto caratteri come *. , : - '* possono essere o no Token delimiters; ad esempio il punto di fine periodo è diverso dal punto che separa le cifre decimali in un numero, oppure il segno "-" è diverso se usato come trattino in un testo o come separatore tra prefisso e numero di telefono o ancora come segno davanti ad un numero.

Un tokenizer ottimizzato deve essere realizzato in base alla tipologia di testo che deve essere trattata. Inoltre, appare evidente che il tokenizer dipende dalla lingua dei testi che devono essere trattati.

Conversion to lowercase

Si tratta di un ulteriore fase di pre-processing dei dati. È necessaria in quanto consente di evitare che termini scritti totalmente o parzialmente in maiuscolo e in minuscolo vengano considerati in maniera differente dal classificatore.

Lemmization e Stemming

Lo scopo principale della fase di lemmization è quello di ridurre drasticamente il numero di termini da trattare. Questa riduzione ha un doppio effetto benevolo sull'analisi del testo. In primo luogo il minor numero di termini agevola la fase di addestramento limitando la necessità di risorse sia in termini di tempo che di spazio; un secondo effetto che si può ottenere da tale riduzione risiede nel fatto che accorpando parole con radice simile e quindi presumibilmente con significato simile, un'eventuale analisi statistica sarebbe da considerare più affidabile. Rispetto agli algoritmi di rimozione del suffisso, la differenza chiave è che la lemmization utilizza (come passo preliminare) l'identificazione della parte del discorso rappresentata da un termine (*part of speech tagging*): l'idea è quella di usare regole di stemming relativizzate alla parte del discorso identificata (sostantivo, verbo, ecc.).

Lo stemming è il processo di riduzione della forma flessa di una parola alla sua forma radice, detta tema. Il tema non corrisponde necessariamente alla radice morfologica (lemma) della parola: normalmente è sufficiente che le parole correlate siano mappate allo stesso tema (ad esempio, che andare, andai, andò mappino al tema and), anche se quest'ultimo non è una valida radice per la parola. Quello che ci aspettiamo da un buono stemmer è che riconosca (con la maggiore accuratezza possibile) la correlazione tra termini a cui attribuiamo una semantica comune, e sfrutti questa correlazione per sostituire tutti i termini correlati con il tema corrispondente. Ad esempio:

- 'gatto', 'gatta', 'gattino', 'gattaccio', ecc. saranno mappati sul tema 'gatto', o 'gatt';
- 'fish', 'fished', 'fishing', 'fisher', ecc. saranno mappati sul tema 'fish'.

In base all'approccio utilizzato, gli algoritmi di stemming si possono dividere in alcune famiglie:

- Algoritmi a forza bruta;
- Algoritmi che operano per rimozione dei suffissi (suffix stripping);
- Algoritmi di lemmatizzazione;
- Algoritmi stocastici;
- Algoritmi ibridi;
- Algoritmi di matching.

Part of speech tagging

Se lo scopo dell'analisi è l'estrazione di informazioni dettagliate, è opportuno operare un'analisi grammaticale dei token. Dopo aver diviso il testo in frasi, si può quindi procedere con un'assegnazione del ruolo grammaticale (*part of speech*), attribuendo ad ogni token il ruolo di nome, verbo, aggettivo, avverbio ecc.

Naturalmente, per questa fase di pre-processing l'apporto linguistico è fondamentale. Il riconoscimento del ruolo grammaticale di un termine si basa in gran parte sulla costruzione (manuale) di dizionari, detti proprio grammatiche, che riportano un elenco di termini associati ai ruoli. Esistono perciò molti casi ambigui. In tal caso, è possibile costruire degli appositi classicatori supervisionati, ancora una volta basati sui token presenti in un intorno del termine di interesse. La difficoltà maggiore risiede nella scarsità di testi già annotati che fungano da training set.

Le parti del discorso possono essere categorizzate come classi chiuse, ovvero quelle in cui la condizione di appartenenza è relativamente fissa, ad esempio le proposizioni, e classi aperte, in cui è possibile di volta in volta trovare nuovi elementi, dovute a parole di recente conio. Un esempio di tagset è quello del Penn Treebank, adottato anche per l'annotazione di diversi importanti corpora.

Ad esempio, si consideri la frase "Quella porta non è chiusa bene.". In questo caso il PoS-Tagging genererà il seguente risultato:

PAROLA	CATEGORIA SINTATTICA
Quella	*Aggettivo*
Porta	*Nome comune*
Non	*Avverbio*
È	*Verbo ausiliare*
Chiusa	*Verbo P.P.*
Bene	*Avverbio*
.	*Punteggiatura*

L'esempio precedente è caratterizzato dalla presenza di un certo numero di parole cui, secondo la grammatica italiana, è possibile assegnare più categorie sintattiche. Infatti:

- la parola "quella" può essere usata sia come pronome che come aggettivo dimostrativo;
- la parola "porta" può essere usata sia come sostantivo (nome comune) che come verbo ("egli porta il libro in biblioteca");

- la parola “chiusa” può essere usata come verbo al participio passato, come aggettivo e come sostantivo;
- la parola “bene” può essere usata sia come avverbio sia come sostantivo.

Da queste osservazioni si evidenzia che uno dei problemi principali che deve essere affrontato nel PoS-Tagging è la risoluzione dell’ambiguità delle parole. A tal proposito, tutti i metodi esistenti in letteratura per eseguire il PoS-Tagging concordano sul fatto che:

- ogni parola deve avere un set di tag limitato che può essere ricercato in un lessico o attraverso un’analisi morfologica della parola;
- quando una parola ha più di un possibile tag, bisogna avere a disposizione delle regole che determinino il tag corretto in relazione al contesto.

I PoS-Tagger eliminano l’ambiguità della categoria sintattica di una parola prendendo in esame il contesto in cui essa si colloca.

Eliminazione delle stop-words

La stop-word list è l’insieme delle parole con peso zero rispetto al documento. In questa fase quindi sono eliminate le parole comuni con un grado di informazione minimo del documento rispetto al topic. Esempi in questo sono possono essere articoli, congiunzioni, avverbi, verbi ausiliari e verbi che non portano informazione (potere, fare, ecc…). Ovviamente, come si può intuire, l’insieme delle stop-word può cambiare in base alla lingua del documento. I principali vantaggi di questa fase risiedono nella riduzione delle dimensioni del documento originale e nel miglioramento della precisione ma con effetti indesiderati quali possibile perdita di informazione e riduzione del potere di richiamo.

Generazione degli n-grammi

Un N-gramma è una sotto sequenza di n elementi di una data sequenza. Secondo l'applicazione, gli elementi in questione possono essere fonemi, sillabe, lettere, parole, ecc. Un n-gramma è di lunghezza 1 prendi il nome di "unigramma", di lunghezza 2 "digramma", di lunghezza 3 "trigramma" e, da lunghezza 4 in poi, "n-gramma". Spesso rappresentano costruzioni sintattiche comuni come ad esempio <preposizione + articolo> e sono particolarmente utili nel caso in cui i testi trattati siano in lingue non note al sistema, o in ricerche che coinvolgano problemi di authorship analysis. Riescono altresì a modellare dipendenze che derivano da regole grammaticali (es. ad un aggettivo segue probabilmente un nome), restrizioni semantiche (es. mangio un panino vs. mangio una chiave) e restrizioni culturali (es. mangio un gatto).

Feature Selection

Come detto, ogni documento corrisponde ad un'osservazione, quindi ad un vettore del data set. Ciò che si osserva nei documenti sono le parole, più precisamente i token individuati nel preprocessing. Un vettore di feature è dunque costruito in modo che ogni componente corrisponda ad un token. In particolare, una componente singola del vettore prenderà valori diversi da zero se il token corrispondente è presente nel documento. Ogni vettore sarà perciò composto da p componenti, dove p è il numero di token ritrovati in tutta la collezione di testi analizzati. Delle p componenti, solo poche saranno

diverse da zero in un singolo vettore, poiché un documento riporta solo un numero limitato di parole, rispetto a tutte quelle ritrovate nella collezione. I vettori saranno perciò sparsi, e sarà necessario, nella fase di implementazione, disporre di buoni algoritmi per la gestione ottimale di vettori sparsi.

La sparsità dei vettori non è il solo problema. In generale, è facile aspettarsi che p sia molto grande. Il linguaggio naturale è infatti tale da richiedere, anche in ambiti specifici, l'impiego di un ampio vocabolario. Se allora n è il numero di documenti osservati (e quindi di vettori costruiti), saremo nel caso delicato in cui $p >> n$.

In letteratura sono stati studiati diversi metodi per ridurre il numero di feature andando a selezionare soltanto quelle più significative, ad esempio basate sulla *mutual information*, sull'*information gain* e sulla metrica del *Chi-Square*.

Document Frequency

Come accennato, una funzione di feature selection semplice e sorprendentemente efficace è proprio la *document frequency* #(tk) di un termine tk, usata per la prima volta in [Apt94] e successivamente studiata sistematicamente in [Yan97]. In questo contesto per spazio degli eventi si intende l'insieme di tutti i documenti nel training set;. In termini probabilistici, pertanto, la document frequency si può anche scrivere come P(tk).

Yang e Pedersen [Yan97] hanno mostrato che, senza tener conto del classificatore adottato e del corpus iniziale usato, usando #(tk) come tecnica di selezione delle feature è possibile ridurre la dimensionalità dello spazio dei termini di un fattore di 10 senza perdita in efficacia (mentre una riduzione di un fattore 100 comporta una piccola perdita).

Questo risultato sembra indicare, preliminarmente, che i termini più rilevanti per la classificazione sono quelli che occorrono più frequentemente nella collezione e pertanto, è incompatibile con uno dei principi fondamentali dell'Information Retrieval, per cui i termini a maggior contenuto informativo sono quelli con una frequenza nel documento medio-bassa [Yan97]. In realtà i due risultati non sono in contraddizione, perché è noto (vedi [Apt94]) che la maggior parte delle parole che occorrono almeno una volta in un dato corpus hanno una document frequency estremamente bassa; ciò vuol dire che, eseguendo una TSR con un fattore 10 usando la document frequency, solo queste parole sono rimosse, mentre le parole con una document frequency medio-alta sono preservate.

Infine, si osservi che una forma leggermente più empirica di document selection basata sulla document frequency è adottata da vari autori ([Itt95,Li98]), che escludono tutti i termini che occorrono in almeno x documenti di training (valori tipici di x variano da 1 a 3), sia quale unica forma di riduzione dimensionale o prima di applicare una forma più sofisticata.

Function	Denoted by	Mathematical form
Document frequency	$\#(t_k, c_i)$	$P(t_k, c_i)$
Information gain	$IG(t_k, c_i)$	$P(t_k, c_i) \cdot log\frac{P(t_k, c_i)}{P(c_i) \cdot P(t_k)} + P(\overline{t}_k, c_i) \cdot log\frac{P(\overline{t}_k, c_i)}{P(c_i) \cdot P(\overline{t}_k)}$
Chi-square	$\chi^2(t_k, c_i)$	$\frac{g \cdot [P(t_k, c_i) \cdot P(\overline{t}_k, \overline{c}_i) - P(t_k, \overline{c}_i) \cdot P(\overline{t}_k, c_i)]^2}{P(t_k) \cdot P(\overline{t}_k) \cdot P(c_i) \cdot P(\overline{c}_i)}$
Correlation coefficient	$CC(t_k, c_i)$	$\frac{\sqrt{g} \cdot [P(t_k, c_i) \cdot P(\overline{t}_k, \overline{c}_i) - P(t_k, \overline{c}_i) \cdot P(\overline{t}_k, c_i)]}{\sqrt{P(t_k) \cdot P(\overline{t}_k) \cdot P(c_i) \cdot P(\overline{c}_i)}}$
Relevancy score	$RS(t_k, c_i)$	$log\frac{P(t_k \mid c_i) + d}{P(\overline{t}_k \mid \overline{c}_i) + d}$

Tabella 1: le principali funzioni proposte per la riduzione di spazio.

L'Information gain è chiamata anche informazione mutuale attesa in Lewis [Lew92]. Nelle formule $\chi 2$ e CC, g è in genere la cardinalità del training set. Nella formula RS(tk,ci)d è una costante del fattore di camping.

Frequency analysis

Un documento può essere rappresentato come un insieme di features (words, n-grammi, concetti), eventualmente pesati con misure che esprimono la loro presenza quantitativa nel documento. In genere, per determinare il peso w_{jk} del termine t_k nel documento d_j, può essere utilizzata una qualunque tecnica di indicizzazione che rappresenta un documento come un vettore di termini pesati. La maggior parte delle volte, si utilizza la funzione standard per la valutazione dei pesi *tf-idf* ([Sal75]), definita come tf-idf(t_k,d_j) = #(t_k,d_j)·log $|T_r|$ #(t_k), ove #(t_k,d_j) indica il numero di volte in cui t_k compare in d_j (Term Frequency), #(t_k) indica il numero di documenti di T_r in cui t_k compare almeno una volta (detta anche *Document Frequency* del termine t_k), e g è la cardinalità del training set T_r. Questa funzione si basa sulle seguenti osservazioni: (i) tanto più un termine occorre in un documento, tanto più esso rappresenta il contenuto del documento stesso; (ii) tanto più alto è il numero di documenti in cui un certo termine appare, tanto meno il termine è utile per eseguire una discriminazione tra i documenti stessi. Per garantire che il valore restituito per il peso sia compreso tra 0 e 1 e che tutti i documenti siano rappresentati con un vettore della stessa lunghezza, i pesi calcolati attraverso la funzione *tfidf* sono in genere normalizzati attraverso la funzione coseno, come segue:

$$w_{jk} = \frac{tfidf(t_k, d_j)}{\sqrt{\sum_{s=1}^{|T|}(tfidf(t_s, d_j))^2}} \quad (2)$$

Nonostante *tf-idf* sia la funzione più popolare, in letteratura sono state utilizzate anche altre funzioni per l'indicizzazione dei documenti ([Seb02]).

Clustering terminologico

Il *clustering terminologico* ha lo scopo di raggruppare le parole che hanno un elevato grado di correlazione semantica in cluster che possano essere usati al posto dei termini quali dimensioni dello spazio vettoriale. Ogni metodo per il clustering terminologico deve specificare:

- un metodo per raggruppare le parole in cluster;
- un metodo per convertire la rappresentazione originale di un documento d_j in una nuova rappresentazione basata su dimensioni ridotte.

Questo approccio è stato usato da Li e Jan [Li98], che guardano alla correlazione semantica tra i termini secondo la loro co-occorrenza e la co-assenza all'interno dei documenti di training. Usando questa tecnica nel contesto di un algoritmo di clustering gerarchico si ottiene solo un miglioramento marginale: comunque le ridotte dimensioni degli esperimenti rendono impossibile trarre opportune conclusioni.

Latent semantic indexing (LSI)

Il *latent semantic indexing* è una tecnica per la riduzione dimensionale sviluppata originariamente nell'ambito dell'Information Retrieval al fine di risolvere i problemi derivanti dall'uso di sinonimie e polisemie quali dimensioni dei documenti e delle rappresentazioni di query. Questa tecnica comprime i vettori rappresentando documenti o query in altri vettori in uno spazio a dimensioni più piccole le cui dimensioni sono ottenute come combinazione delle dimensioni originali considerando i pattern di co-occorrenze. La funzione che mappa i vettori originali in nuovi vettori ottenuti è ottenuta applicando la singular-variable decomposition ad una matrice di incidenza formata dai vettori del documento originale. Nell'ambito della classificazione documentale, questa tecnica è applicata derivando la funzione di mapping dal training set ed applicandola ad ogni documento di test in modo da produrre una rappresentazione in uno spazio a dimensione più piccola.

Una caratteristica del LSI quale funzione di riduzione dimensionale è che le nuove dimensioni ottenute non sono direttamente interpretabili, al contrario dei casi della fetaure selection e del clustering terminologico.

Comunque, queste tecniche tendono a lavorare tenendo conto della struttura semantica "latente" del vocabolario usato nel corpus. Ad esempio, Schutze et al. [Sch98], discutono il caso della classificazione in categorie demografiche. Nell'esperimento riportato, si discute il caso di un documento che costituisce un'istanza positiva di test per la categoria, e che contiene, tra gli altri, la frase rilevante "*The nation grew to 249.6 million people in the 1980s as more Americans left the industrial and agricultural heartlands for the South and West*". La classificazione è errata se la riduzione dimensionale è eseguita con il metodo di selezione delle feature basato su $\chi 2$ rispetto ai 200 termini originali principali, mentre è corretta se lo stesso metodo è applicato a valle del LSI. In questo modo si spiega il funzionamento di LSI: la fase descritta in precedenza non contiene nessuno dei 200 termini più rilevanti delle categorie selezionate da $\chi 2$, ma in tutte le prove le parole contenute nel documento concorrono a generare una o più delle feature principali del LSI che generano lo spazio delle categorie del documento.

Pertanto, come osservato da Schutze et al., "se esiste un grande numero di termini che individuano una piccola quantità di informazioni critiche, allora la combinazione delle prove è uno dei problemi principali per un classificatore basato su termini" [Sch98]. Uno svantaggio del LSI è che se qualche

termine originale è particolarmente adatto a discriminare una categoria, questa caratteristica si può perdere nel nuovo spazio vettoriale.

Wiener et al. [Wie95] usano LSI in due modi alternativi: i) per la riduzione dimensionale locale, creando diverse rappresentazioni LSI specifiche per singole categorie, e ii) per la riduzione dimensionale globale, creando una singola rappresentazione LSI per tutto l'insieme delle categorie. In ogni caso, entrambi gli approcci bastai su LSI si dimostrano più efficaci di una tecnica di selezione delle feaure basata su una misura di rilevanza.

Latent Dirichlet Allocation (LDA)

LDA (Latent Dirichlet Allocation) [Ble03] Gibbs è un modello probabilistico utilizzato per la generazione di corpora documentali.

Un corpus viene rappresentato da un insieme di frasi o parole estratte da diversi contesti. Le parole di cui è permettono di stabilire l'appartenenza di un documento ad un contesto (*topic*). Per cui i documenti possono fare parte di uno o più corpora distinti.

Un documento può essere rappresentato, in linea di massima, dagli argomenti di cui tratta. Ogni categoria è caratterizzata da una definita distribuzione di parole.

La LDA permette di stimare l'appartenenza delle parole di un vocabolario, finito e condiviso dai diversi documenti, ad un numero finito di corpora. Tramite la LDA, è possibile stimare quindi l'appartenenza di un documento ad un corpus, nonché di rappresentare per mezzo di una stima il documento con i diversi argomenti di cui tratta.

La costruzione di un modello predittivo LDA Gibbs avviene in due passi:

- Inizializzazione: Ogni parola viene associata ad un topic in maniera random.
- Sampling: Le parole vengono ridistribuite nei vari topics in maniera tale da ridurne l'entropia. Il nuovo topic viene valutato basandosi su una probabilità multinomiale.

Basandosi sulla distribuzione delle parole è possibile stabilire l' "allocazione" dei documenti ai vari gruppi. Il modello predittivo si basa sulla probabilità di appartenenza delle parole nelle diverse categorie per stimare l'appartenenza di un qualsiasi documento ai *topic* definiti per il modello.

L'algoritmo LDA, così come tutti i topic model, stima un'associazione debole tra i topic latenti e i campioni osservati. Queste associazioni possono essere usate in svariati modi nella *knowledge discovery*. Uno scenario applicativo diretto della scoperta dei topic model è il clustering dei gruppi della collezione. La LDA fornisce esplicitamente un soft clustering dei gruppi, in cui i cluster sono rappresentati dai topic. Dato un gruppo d ed un topic k, la probabilità condizionale del topic dato il gruppo è:

$$p(z = k|\theta_d, \vec{\alpha}) = \theta_{d,k}^{n_{d,(\cdot)}^k}$$

Quindi il vettore $p_d = \{p(z = k|\theta_d, \vec{\alpha})\}_{k=1}^{K}$rappresenta la distribuzione di probabilità dei topic, ovvero dei cluster, all'interno del gruppo. L'insieme $\{p_d\}_{d=1}^{M}$ rappresenta dunque il soft clustering. Una versione di hard clustering può essere implementata associando ad ogni gruppo il topic (cluster) più probabile.

Un secondo scenario applicativo di interesse, per l'utilizzo della LDA ed in generale dei topic models, è la *feature selection*. Spesso i gruppi di una collezione possono contenere molti campioni, inoltre, il numero di campioni distinti in tutta la collezione può essere molto elevato. Ad esempio, si pensi ad un corpus documentale, i cui gruppi sono rappresentati dai documenti. Ogni documento contiene le parole che rappresentano i campioni: l'insieme delle parole distinte del corpus potenzialmente potrebbe essere uguale a tutto il vocabolario. È quindi evidente il problema della dimensionalità dei dati.

I topic scoperti, per loro natura, rappresentano una compressione delle informazioni fornite dai campioni. Per tal motivo, una possibile strategia per la riduzione della dimensionalità dei dati, e quindi di *feature selection*, è la sostituzione dei campioni con i topic. Questa trasformazione porta alla definizione di una nuova collezione di gruppi di dati discreti simile per contenuto informativo a quella originale, ma dalle dimensioni notevolmente ridotte.

3.1.1.3. Analisi Semantica del Testo

Esiste un ulteriore modo per organizzare globalmente gli elementi del testo: le parole "casa", "edificio" e "abitazione" sono in qualche modo intuitivo più collegate di quanto non lo siano "casa", "cane" e "cavo" e questa "vicinanza" è chiaramente non evidente osservando l'ordine alfabetico o quello di frequenza. In effetti tra "casa" ed "edificio" sussiste una relazione semantica chiamata *iponimia*: "casa" è un termine più specifico di "edificio", cioè il termine "casa" è applicabile solo ad un sottoinsieme di oggetti per cui si può propriamente dire che sono "edifici" (la relazione inversa si definisce di *iperonimia*: "fiore" è un iperonimo di "orchidea"). Tra "casa" ed "abitazione" sussiste invece una relazione di *sinonimia* (tale relazione è ovviamente bidirezionale, esiste comunque anche una relazione di "polarità inversa" definite di *antinomia* che lega ad esempio l'aggettivo "vuoto" e "pieno"). Altre relazioni interessanti possono essere quella di parte-di (*meronimia*) che lega ad esempio la parola "braccio" a "corpo" o di diversa modalità di un'azione/evento (*troponimia*): "correre" è un troponimo di "camminare".
Si può immaginare una rete che connette in vario modo gli elementi del nostro lessico e che risulta invisibile partendo dalla forma ortografica, dagli indici di frequenza e dalle informazioni morfosintattiche associate alle parole. Sfruttando queste relazioni si può definire una struttura globale del lessico adatta a rispondere in modo psicolinguisticamente plausibile alla domanda: quale è il significato di una parola? La risposta è data dalla serie di link che legano il significato associato ad una parola ad altri significati di altre parole. La risorsa sviluppata dal gruppo di Fellbaum e Miller (Princeton University) parte da questa idea e si chiama WordNet. Attualmente, giunta alla sua versione 3.0, si presenta come una struttura dati di circa 155.000 parole che esprimono quasi 118.000 significati distinti (synset) organizzati in circa 207.000 relazioni semantiche. WordNet è spesso considerato un'importante base di conoscenza (Lexical Knowledge Base, LBK) e ha avuto un notevole successo presso la comunità scientifica che ha cercato di riprodurlo in diverse lingue e di allinearlo alla versione inglese. Da segnalare, in questo senso, è il progetto EuroWordNet: progetto finanziato dall'Unione Europea che ha coinvolto 12 partner europei e ha permesso di riprodurre una rete completa di lemmi sul modello inglese

per Ceco, Estone, Fiammingo, Francese, Italiano, Spagnolo e Tedesco. Si è quindi tentato di allineare ogni distinta struttura con un'interlingua semantica basata sul WordNet inglese (Inter-Lingual Index). In modo simile, il progetto MultiWordNet dell'ITC-IRST (Istituto per la Ricerca Scientifica e Tecnologica dell' Istituto Trentino di Cultura, adesso Fondazione Bruno Kessler, FBK) ha permesso di sviluppare una risorsa multilingue basata su questa rete semantica (Italiano, Spagnolo, Portoghese, Ebraico, Rumeno e Latino). In MultiWordNet si è dato priorità alla struttura inglese e su questa si è iniziato il processo di traduzione e individuazione dei gap cross-linguistici in ciascuna lingua (ad esempio, il concetto di "mollica" è lessicalizzato in Italiano, ma non in Inglese): in generale la possibilità di allineamento tra i vari wordnet in lingue diverse si basa sull'ipotesi che la struttura concettuale sia effettivamente universale anche se gap linguistici, cioè concetti/synset non lessicalizzati in una lingua, porterebbero a pensare altrimenti. Guardando all'italiano in termini quantitativi, le risorse presenti in EuroWordNeted in MultiWordNet essenzialmente si equivalgono: ItalWordnet, parte di EuroWordNet, comprende 47.000 lemmi, 50.000 synsets e circa 130.000 relazioni semantiche. La parte italiana di MultiWordNet comprende invece circa 46.000 lemmi, quasi 39.000 synsets allineati con il WordNet inglese e circa 120.000 relazioni semantiche (i dati si riferiscono alla versione 1.42).

La complessità del problema deriva in effetti dal fatto che la mappatura tra "parole" ed "etichette" non è sempre univoca; ad esempio una famosa frase ambigua:

la vecchia legge la regola

Le due interpretazioni possibili sono le seguenti:

a) una legge antica controlla qualcosa (ad esempio un determinato contesto)
b) una vecchia signora sta leggendo una certa direttiva

Assegnare a questa frase una struttura richiede prima di tutto l'assegnazione ad ogni token di un Part of Speech (o PoS). Assumendo che questa sia la lista di PoS:

D (articolo), Pro (pronome), A (aggettivo), N (nome), V (verbo).

Si può facilmente notare che tutte le parole della frase sono ambigue tra due PoS:

la = D/Pro

vecchia = N/A

legge = N/V

regola = N/V

Si parla in questo caso di *ambiguità lessicale*, cioè una stessa parola (o token) è ambigua tra due categorie morfologiche distinte (o PoS).

L'ambiguità lessicale si ripercuote chiaramente anche sulla struttura della frase, ad esempio scegliere di interpretare "vecchia" come nome o come aggettivo determina una suddivisione in costituenti ben precisa a livello sintattico.

Ci sono casi, come il seguente, in cui non basta rimuovere l'ambiguità lessicale per avere accesso a soluzioni strutturali distinte:

Gianni ha visto Maria con il cannocchiale

La frase potrebbe significare sia che Gianni ha usato un cannocchiale per vedere Maria, sia che Maria aveva un cannocchiale quando Gianni l'ha vista.

Questa *ambiguità* si definisce *sintattica*, poiché legata alla diversa assegnazione dei rapporti di inclusione/modificazione tra i costituenti, è stata estensivamente studiata sia da un punto di vista psicolinguistico che da un punto di vista computazionale, ed è nota come il problema di attaccamento dei sintagmi preposizionali (PP, PrepositionalPhrase, attachment).

C'è infine una terza tipologia di ambiguità, *l'ambiguità semantica*. La parola "pianta", può significare sia "parte del piede" che "vegetale". La categoria grammaticale (PoS) è la stessa nei due casi (N), quindi c'è un'ambiguità lessicale; d'altra parte nelle espressioni "mostrami [la pianta [del piede]]" e "annaffia [la pianta]", il sintagma "la pianta" è in entrambi i casi un *sintagma nominale* (NP), quindi non si può parlare neppure di ambiguità sintattica. Questo approccio alla struttura globale del lessico è molto interessante perché permette di gestire in modo soddisfacente il problema dell'ambiguità semantica attraverso il livello di ambiguità lessicale (alla parola "vecchia" si può associare sia il PoS "aggettivo" che il PoS "nome"), quello dell'ambiguità sintattica (alla frase "ho visto un uomo con il cannocchiale" si possono assegnare due strutture soggiacenti: [ho visto [un uomo [con il cannocchiale]]] Vs. [ho visto [un uomo] [con il cannocchiale]]). Il terzo livello di ambiguità, quello semantico, appunto, trova un'esplicita rappresentazione in WordNet: parole polisemiche, che hanno cioè significati distinti, sono associate a synset distinti in posizioni diverse della rete di relazioni di wordnet (rete che tecnicamente si chiama grafo).

Thesauri

Altri strumenti a supporto dell'analisi semantica dei testi sono i thesauri. Un thesauro è un vocabolario controllato. Nella sua forma più semplice, un vocabolario controllato è un sottoinsieme di un linguaggio che rappresenta un sapere specialistico, per esempio un elenco (indice) dei termini specifici di una disciplina (arte, medicina, economia, ecc.). Un vocabolario controllato di questo tipo può essere:

- deciso da uno o più esperti, o
- costruito automaticamente scartando dai testi del settore le parole cosiddette "non-stop" (articoli, preposizioni, pronomi, ecc.).

Un vocabolario controllato diventa uno schema di classificazione, (schema organizzativo) o tassonomia, quando i termini vengono organizzati in una gerarchia.
In particolare, Un thesaurus è un vocabolario controllato in cui vengono esplicitate relazioni semantiche fra termini. Precisamente:

- **relazioni di equivalenza** fra i termini (anelli di sinonimi);
- **relazioni gerarchiche** fra i termini preferiti (schemi di classificazione);
- **relazioni associative** fra i termini. Associativa è per esempio la relazione di contestualità fra termini, come "forchetta" e "coltello", "autostrada" e "casello", "Waterloo" e "Napoleone".

I concetti rappresentati dai termini di un thesaurus possono appartenere a diverse categorie:

- entità concrete:
  - oggetti e loro parti fisiche;
  - materiali;
- entità astratte:
  - azioni e avvenimenti;
  - entità astratte e proprietà degli oggetti, dei materiali o delle azioni;
  - discipline o scienze;
  - unità di misura;
- entità individuali o "classi di uno" analoghe a nomi propri.

In fase di costruzione del thesaurus è di fondamentale importanza il controllo dell'appartenenza dei termini a queste categorie, poiché esse possono influenzare determinate procedure, come ad esempio la scelta del plurale o del singolare, o verificare la validità delle gerarchie (non può esistere rapporto gerarchico fra termini appartenenti a categorie diverse).

3.1.2. Tecniche di Document Embedding

Il word embedding, ovvero il mapping di parole in vettori numerici, è una tecnica di NLP che ha acquisito negli anni sempre maggiore interesse, consentendo l'uso di vari modelli di machine learning che sfruttano una rappresentazione vettoriale dell'input. Tali rappresentazioni preservano maggiormente le informazioni sintattiche e semantiche delle parole, e ciò consente di ottenere migliori prestazioni nei task di NLP.
L'efficacia e l'innovatività dell'approccio sopra introdotto ha spinto i ricercatori a considerare il problema di come sfruttare tali vantaggi anche nel caso di grandi quantità di testo, anche delle dimensioni di libri, portando alla definizione di nuovi metodi per la definizione di tali mapping.
Si noti che in questa sede per documento si intende una qualsiasi sequenza di parole. Si noti inoltre che non ci si riferirà esclusivamente a tecniche di word enbedding, ma anche ad altre tecniche purché esse producano un mapping tra i documenti e vettori in $\mathbb{R}^n$.

Sebbene il problema del document embedding sia relativamente vecchio, molte delle soluzioni migliori sono recenti, essendo ritornato un notevole interesse sull'argomento da pochi anni. I nuovi approcci sono in larga misura influenzati dalle tecniche di word embedding basate sul paradigma encoder-decoder.

È possibile raggruppare tali nuovi approcci come segue:

- *Sommarizzazione dei word vector*. Si tratta dell'approccio classico. Uno dei metodi utilizzati è il Bag-of-words (su cui si ritornerà più avanti), e che viene utilizzato tra l'altro nell'ambito del Natural Language Processing. In sintesi, esso gestisce ed elabora i documenti senza tenere in considerazione l'ordine con cui le parole in essi contenuti compaiono. Da qui il nome Bags of Words, ovvero borse di parole.
- *Modellazione dei topic*. Malgrado non si tratti dell'applicazione principale per le tecniche di topic modelling come LDA e PLSI, esse proiettano intrinsecamente un documento in uno spazio a bassa dimensionalità (*document embedding*), che consente di modellare e spiegare la distribuzione delle parole nel corpus e dove le dimensioni possono essere viste come strutture semantiche latenti nascoste nei dati, e sono quindi utili nel nostro contesto.

- *Modelli encoder-decoder*. Malgrado tale approccio sia stato proposto a partire dai primi anni del 2000, esso ha riscosso maggiore interesse ultimamente, ed è stato con successo applicato alla generazione di word embedding.
- *Supervised representation learning*. Tale approccio deve la sua introduzione alla grande crescita della disponibilità di modelli di Neural Network, e la loro abilità di apprendere ricche rappresentazioni di dati in input, usando vari operatori non lineari multi-layer, che possono approssimare un ampio insieme di mapping.

3.1.2.1. Question Answering e chatbots

Nei task di question answering, il modello riceve una domanda riguardante il contenuto testuale, e gli viene richiesto di marcare l'inizio e la fine della domanda nel testo.
Il Question Answering è una interazione uomo-macchina atta a estrarre informazioni dai dati, usando query in linguaggio naturale. Una macchina tipicamente non raggiunge il grado di comprensione del linguaggio naturale che caratterizza un essere umano dotato di un'intelligenza media. Un sistema di Question Answering di buona qualità consente di ridurre tale gap, consentendo agli esseri umani di estrarre conoscenza dai dati in un modo che ci risulta naturale, ovvero ponendo domande.

I sistemi di Question Answering accettano domande in linguaggio naturale (tipicamente scritte, ma esistono sistemi che riconoscono il linguaggio parlato, come Alexa di Amazon o Siri di Apple), e generano in output una risposta concisa. Il search engine di Google, ad esempio, ha affiancato una forma di question answering in aggiunta alla ricerca tradizionale. Una volta posta la domanda, Google restituisce oltre un milione di documenti (non mostrati) rilevanti per i termini di ricerca. Google mostra anche una porzione del testo (snippet) che dovrebbe rispondere al quesito con poche parole, sopra al link al relativo documento (quello che ritiene più pertinente), evidenziando nella relativa pagina le porzioni di testo prima riferite.

Il Question Answering va oltre le capacità standard di un search engine, che tipicamente restituisce solo una lista di documenti e siti web rilevanti. Google recentemente ha spiegato che tali risultati sono ottenuti per mezzo di un approccio innovativo al NLP, su cui si tornerà a breve.

Numerosi sono i campi applicativi che sfruttano efficacemente il QA e tra questi vi è senz'altro quello delle chatbots, di interesse nella presente sezione. In tale ambito, i sistemi di Question Answering possono migliorare le performance delle tecnologie esistenti, fornendo una comprensione più profonda, al fine di migliorare l'esperienza dell'utente. Ad esempio, un sistema di Question Answering con conoscenza delle domande frequenti relative a un prodotto può semplificare l'esperienza del cliente, mentre i sistemi di QA costruiti sulla documentazione interna di un'azienda potrebbero fornire ai dipendenti un accesso più semplice a registri, rapporti, rendiconti finanziari o documenti di progettazione.

3.1.2.2. BERT

Si introduce in questa sezione BERT, un algoritmo che può essere opportunamente adottato nell'ambito dei sistemi di question answering, chatbots, ed in generale per il NLP.

Algoritmo

BERT (Bidirectional Encoder Representations from Transformers) è un algoritmo attualmente utilizzato da Google nel proprio motore di ricerca. Si tratta di un natural language processor che consente di comprendere meglio il contesto delle query di ricerca, al fine di mettere a disposizione risultati più rilevanti nella comprensione di forme linguistiche che prima di esso non erano comprensibili al computer mezzo di altri algoritmi. Nello specifico, BERT può comprendere parti di parlato come pronomi, fornendo un meccanismo per analizzare in maniera più efficace query scritte in linguaggio naturale. È necessario tuttavia precisare che BERT non estende attualmente la raggiungibilità ai siti web scritti in linguaggio meno comprensibile. Questo algoritmo infatti attualmente si applica solo alle query di ricerca degli utenti, aumentando notevolmente la pertinenza dei risultati. Google Bert è stato rilasciato in open source, al fine di consentire agli sviluppatori di personalizzarlo e promuovere nuove tipologie di utilizzo.

Prima di introdurre Bert, si introducono i seguenti modelli di NLP, che sono utili per la comprensione del funzionamento di Bert:

- Bags of words. Già introdotto brevemente prima, Bags of words rappresenta un approccio semplificato all'NLP. Ogni testo è rappresentato come una "*borsa"* di parole, ignorando il contesto, la grammatica e l'ordine delle parole. Si prenda in considerazione il seguente esempio, in lingua inglese: "EdLab is a research, design, and development unit at Teachers College, Columbia University". In tale frase, la Bag of Words è la seguente: {"EdLab" : 1, "is" : 1, "a" : 1, "research" : 1, "design" : 1, "and" : 1, "development" : 1, "unit" : 1, "at" : 1, "Teachers" : 1, "College" : 1, "Columbia" : 1, "University" : 1}. Un ulteriore esempio è il seguente (sempre in lingua inglese): "John went to the river bank today, and he visited the bank to withdraw his salary on his way back". Bag of words: {"John" : 1, "went" : 1, "to" : 2, "the" : 2, "river" : 1, "bank" : 2, "today" : 1, "and" : 1, "he" : 1, "visited" : 1, "withdraw" : 1, "his" : 2, "salary" : 1, "on" : 1, "way" : 1, "back" : 1 }. Dagli esempi sopra, è possibile notare che:
    - Il modello bag of words mantiene intatta la molteplicità delle parole;
    - Il modello non prende in considerazione il contesto.
- Word2Vec. Word2Vec prende in input grandi quantitativi di dati testuali e rappresenta le parole come vettori, sulla base del contesto delle parole. Word2Vec identifica il contesto sulla base di come le parole vengono usate nel testo. La posizione di una parola rispetto a un'altra parola definisce la relazione che intercorre tra esse. Ad esempio, può capire il contesto delle parole "King", "Queen", "Man" e "Woman", tale per cui King - Man + Woman = Queen. Il modo in cui quanto detto prima viene calcolato è rappresentato nella seguente figura:

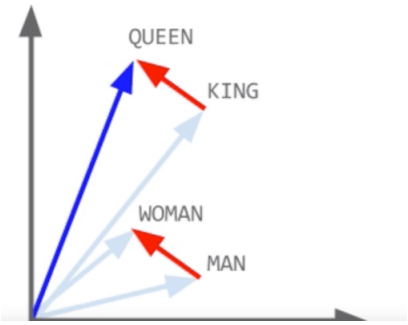

FIG. 15: ESEMPIO DELLA RAPPRESENTAZIONE DEI TERMINI NELLO SPAZIO

Come è possibile osservare dalla figura, l'algoritmo di Word2Vec rappresenta le parole come vettore.

Di seguito si introduce l'algoritmo di BERT, per dare subito una prima intuizione su di esso. BERT è l'acronimo di Bidirectional Encoder Representations from Transformers. L'innovazione principale risiede nella capacità del modello di considerare i termini da entrambe le direzioni (ovvero, le parole presenti prima e dopo una data parola) nella fase di building e di training dell'algoritmo. In buona sostanza il singolo termine acquista semantica in base al contesto in cui ricade.

Più in dettaglio l'algoritmo in esame è costituito sostanzialmente da due passi principali. Per rendere più chiaro il funzionamento dell'algoritmo si utilizza innanzitutto un esempio, come prima in lingua inglese: "*The man goes to the store. He buys a Gallon of Milk*". L'algoritmo, come detto, è costituito da due step:

- *Word Masking*. Prima di lanciare l'algoritmo, una certa percentuale delle parole (circa il 15% usualmente) in ogni sequenza viene nascosto (o mascherato). Il modello successivamente tenta di predire il valore originale della parola mascherata basandosi sul contesto fornito dalle altre parole nella sequenza (in entrambe le direzioni). Nell'esempio: *"The man goes to the [Mask1]. He buys a [Mask2] of Milk"*. L'algoritmo quindi predice [Mask1] come "Store" e [Mask2] come "Gallon". Questo abilita l'algoritmo ad apprendere la relazione tra parole in una frase.
- *Prediction*. Nel secondo step, i modelli aiutano a capire le relazioni che intercorrono tra frasi. Apprende se la frase B viene dopo la frase A o viceversa. Si prendano in considerazione i seguenti due esempi:

Frase A: "*The man goes to the* Store*"*

Frase B: *"He buys a Gallon of Milk"*

Label: IsNextSentence

Frase A*: "The man goes to the* Store

Frase B*: "Penguins are Flightless*"

Label : NotNextSentence

Come è possibile osservare, è molto probabile che nel primo esempio la frase B possa venire dopo la frase A, ma è molto improbabile che sia possibile nell'esempio 2. Ciò consente a BERT di imparare quale frase può seguire un'altra frase.

Tale training viene eseguito dall'algoritmo su grandi corpus di dati. BERT usa l'intero corpus di wikipedia per costruire il modello. Google ha messo a disposizione BERT in una forma preaddestrata. I task di NLP maggiormente interessati dalle innovazioni di Bert sono diversi; tra questi, il Question Answering (SQuAD v1.1), il Natural Language Inference (MNLI) e altri.

Si approfondisce ora l'innovazione tecnica chiave di BERT ovvero l'applicazione del training bidirezionale del transformer, un modello basato sul meccanismo dell'attention molto utilizzato nella modellazione del linguaggio.

BERT si distingue sostanzialmente dagli approcci precedenti, che sono caratterizzati dall'esaminare una sequenza di testo da sinistra a destra o un training combinato da sinistra a destra e da destra a sinistra. I risultati ottenuti mostrano che un modello linguistico addestrato in modo bidirezionale può abilitare alla comprensione di un senso più profondo del contesto e del flusso linguistico rispetto ai modelli linguistici unidirezionali. Sarà inoltre discussa in maniera più approfondita la tecnica denominata Masked LM (MLM) che consente l'addestramento bidirezionale in modelli in cui prima era impossibile.

Nel campo della visione artificiale, ad esempio, è stato più volte dimostrato il valore del transfer learning, che consiste nel pre-addestramento di un modello di rete neurale su un task o dominio (ad esempio ImageNet) e successivamente, tramite *fine-tuning*, l'utilizzo dello stesso modello addestrato come base di un nuovo modello *purpose-specific* (eventualmente appartenente anche ad un dominio differente). Negli ultimi anni, è stato mostrato come tale approccio possa essere proficuamente impiegato in molte attività di NLP.

Un approccio diverso, utilizzato anche nei task di PNL, è il feature-based training. In questo approccio, una rete neurale pre-addestrata produce *word embeddings* che vengono poi sono impiegati come feature nei modelli NLP.

Di seguito viene descritto il funzionamento dell'encoder di un *transformer*. L'input è una sequenza di token, che vengono prima incorporati nei vettori e quindi elaborati nella rete neurale. L'output è una sequenza di vettori di dimensione H, in cui ogni vettore corrisponde a un token di input con lo stesso indice.

Quando si addestrano modelli linguistici, è difficile definire un obiettivo di previsione (ad esempio per mancanza di dati etichettati). L'idea è quello di utilizzare un task surrogato (Self-learning) per estrarre un modello iniziale che sarà poi raffinato non appena più istanze saranno rese disponibili. Molti modelli prevedono la parola successiva in una sequenza (ad esempio "Il bambino è tornato a casa da ___"), un approccio direzionale che limita intrinsecamente l'apprendimento contestuale. Per superare tale problematica, BERT utilizza due strategie:

- *Masked LM (MLM)*. Prima di inserire le sequenze di parole in BERT, il 15% delle parole in ciascuna sequenza viene sostituito con un token [MASK]. Il modello tenta quindi di prevedere il valore originale delle parole mascherate, in base al contesto fornito dalle altre parole non mascherate nella sequenza. In termini tecnici, la previsione delle parole in uscita richiede:
    - L'aggiunta di un layer di classificazione sopra l'output dell'encoder.
    - La moltiplicazione dei vettori di output per la matrice di incorporamento, trasformandoli nella dimensione del vocabolario.
    - Il calcolo della probabilità di ogni parola del vocabolario con softmax.

  La funzione di loss di Bert prende in considerazione solo i valori mascherati ed ignora la predizione dei valori non mascherati. In conseguenza di ciò, il modello converge più lentamente rispetto ai modelli direzionali, una caratteristica che è compensata dalla sua maggiore awareness del contesto.
- *Next Sentence Prediction (NSP)*. Nel processo di training di BERT, il modello riceve coppie di frasi come input ed impara a predire se la seconda frase nella coppia è la frase successiva nel documento originale. Durante il training, il 50% degli input sono una coppia in cui la seconda frase è la frase successiva nel documento originale, mentre nell'altro 50% viene scelta come seconda frase una frase casuale dal corpus.
- L'assunzione è che la frase casuale verrà scollegata dalla prima frase. Per supportare il modello a distinguere tra le due frasi in addestramento, l'input viene pre-elaborato nel modo seguente:
    - Un token [CLS] viene inserito all'inizio della prima frase e un token [SEP] viene inserito alla fine di ogni frase.
    - A ogni token viene aggiunta una frase che indica la frase A o la frase B.
    - Un incorporamento posizionale viene aggiunto a ciascun token per indicarne la posizione nella sequenza.
- Per predire se la seconda frase è effettivamente collegata alla prima, vengono eseguiti i seguenti step:
    - L'intera sequenza di input passa attraverso il modello transformer.
    - L'output del token [CLS] viene trasformato in un vettore 2×1, utilizzando un semplice layer di classificazione (matrici apprese di pesi e bias).
    - Viene calcolata la probabilità di IsNextSequence (ossia il prossimo elemento della sequenza) tramite una funzione di attivazione softmax.

Nella fase di training del modello di BERT, Masked LM e Next Sentence Prediction vengono addestrati insieme, con l'obiettivo di minimizzare la funzione di loss combinata.

Fine-tuning
L'utilizzo di BERT per un task specifico è relativamente semplice.
BERT può essere utilizzato per un'ampia varietà di task linguistici, aggiungendo solo un ulteriore layer al modello core:

- Task di classificazione (ad es. sentiment analysis) sono eseguiti in modo simile alla Next Sentence classification, aggiungendo un layer di classificazione sopra l'output di transformer per il token [CLS].
- In task di Question Answering (ad esempio, SQuAD v1.1), il software riceve una domanda riguardante una sequenza di testo e viene richiesto contrassegnare la risposta nella sequenza. Utilizzando BERT, un modello di Question Answering può essere addestrato apprendendo due vettori aggiuntivi che identifichino l'inizio e la fine della risposta.
- Nel caso della Named Entity Recognition (NER), il software riceve una sequenza di testo ed è richiesto di contrassegnare i vari tipi di entità (*persona, organizzazione, data, ecc.*) che appaiono nel testo. Utilizzando BERT, un modello NER può essere addestrato inserendo il vettore di output di ciascun token in un layer di classificazione che predice l'etichetta NER.

Word Masking
L'addestramento del modello linguistico in BERT viene effettuato predicendo il 15% dei token nell'input, scelti casualmente. Questi token vengono pre-elaborati come segue: l'80% viene sostituito con un token "[MASK]", il 10% con una parola casuale e il 10% utilizza la parola originale. L'intuizione che ha portato gli autori a scegliere questo approccio è la seguente:

- Se usassimo [MASK] per il 100% delle volte, il modello non produrrebbe necessariamente buone rappresentazioni simboliche per parole non mascherate. I token non mascherati sarebbero ancora utilizzati per il contesto, ma il modello sarebbe ottimizzato per predire parole mascherate.
- Se usassimo [MASK] per il 90% delle volte e parole casuali per il 10% delle volte, questo insegnerebbe al modello che la parola osservata non è mai corretta.
- Se usassimo [MASK] per il 90% delle volte e mantenessimo la stessa parola il 10% delle volte, il modello potrebbe semplicemente copiare l'embedding non contestuale.

Di seguito si forniscono alcuni concetti importanti per la comprensione dell'architettura adottata dal modello BERT.

Encoders. I pretrained text encoders prendono in input una sequenza di testo tokenizzato, che viene codificato da un modello neurale multi-livello. La rappresentazione di ogni (sub) token, è un insieme di pesi nascosti, $\{h_t^{(l)}\}$ per ogni layer l, o il suo peso solo sul top layer, $\{h_t^{(-l)}\}$. Diversamente dalle parole a lunghezza fissa, le frasi, o le rappresentazioni dei paragrafi e le rappresentazioni contestualizzate prodotte del testo dipendono dalla lunghezza del testo di input. La maggior parte degli encoder impiegano un'architettura transformer. Esse sono particolari tipi di architetture tipicamente utilizzate per risolvere problemi di traduzione automatica del testo.

Pretrain-Finetune framework. Mentre le rappresentazioni del testo possono essere apprese in una qualunque maniera, esse sono poi sfruttate nella risoluzione di specifici task obiettivo. In particolare, in Dai and Le (2015) viene proposto per la prima volta l'uso dell'output di modelli di linguaggi preaddestrati come preprocessing o fase di inizializzazione. In Howard and Ruder (2018) e Radford et al. (2018) viene dimostrata l'efficacia del finetuning per diversi task obiettivo, aggiornando l'intero modello (preaddestrato) per ogni task.

Architettura

Fondamentalmente, BERT è uno stack di transformer encoder layers, che consistono in self-attention "heads" multiple. Per ogni token di input in una sequenza, ogni head calcola i vettori di chiavi, valori e query, usati per creare una rappresentazione pesata. Gli output di tutte le head nello stesso layer vengono combinati e propagati ad un layer fully-connected. Ogni livello e poi ulteriormente connesso tramite una skip connection e seguito da un livello di Batch Normalization. Il flusso di funzionamento di BERT consiste di due fasi principali come visto accennato prima: il pre-training e il fine-tuning. Il pre-training usa due task supervisionati: la modellazione di linguaggi "masked" (MLM, predizione di token in input "mascherati") e la predizione della successiva frase (NSP, acronimo di next sentence prediction, ovvero predire se due frasi in input sono tra loro adiacenti). Nel fine-tuning, uno o più layer fully connected sono tipicamente agganciati al layer finale dell'encoder. Le rappresentazioni dell'input sono calcolate come segue: ogni parola nell'input viene prima tokenizzata in segmenti di parole, e successivamente sono combinati tre embedding layer (token, position, e segment), al fine di ottenere un vettore di lunghezza fissata. Esistono numerose varianti di BERT, inclusa la versione originale "base" e varianti "large". Esse variano nel numero di heads, layer, e dimensione degli stati nascosti. Di seguito in figura viene mostrata in dettaglio il building block dell'architettura:

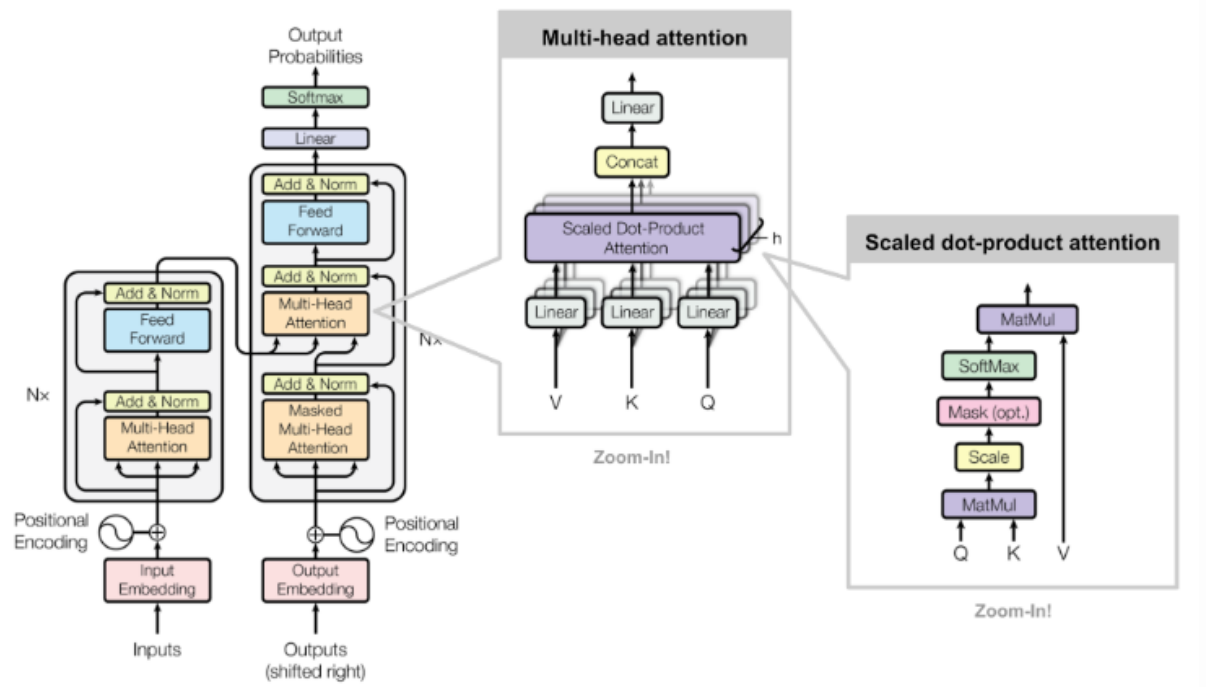

FIG. 16. COMPONENTE BASE DEL MODELLO BERT

4. Multimedia Search ed Information Retrieval

Il significato che si attribuisce al termine information retrieval può cambiare notevolmente a seconda del contesto di riferimento. Tuttavia, in ambito scientifico, l'information retrieval potrebbe essere definito come segue: l'Information Retrieval (IR) consiste nel recuperare materiale (solitamente documenti) di natura non strutturata (solitamente testo) che soddisfa un bisogno di informazioni, all'interno di grandi raccolte (solitamente archiviate sui computer).

L'information retrieval sta rapidamente diventando la forma dominante di accesso alle informazioni, superando la tradizionale ricerca con query su database. Il termine "*dati non strutturati*" si riferisce genericamente a dati che non hanno una struttura chiara, semanticamente aperta e immediatamente comprensibile per un computer. È l'opposto di dati strutturati, il cui esempio canonico è un database relazionale. In realtà, quasi nessun dato è veramente "non strutturato".

I dati multimediali stanno diventando onnipresenti. Dalle foto ai video, a numerose altre tipologie di dati, i dati multimediali stanno diventando parte di tutte le applicazioni. La maggior parte delle applicazioni sono fornite di dati multimediali che sono parte integrante delle loro sorgenti dati. In numerosissime aree applicative, che vanno dalla sicurezza nazionale all'assistenza sanitaria, ecc., fanno largo uso di dati multimediali. Sulla base delle attuali tendenze attuale, è facile prevedere che gran parte dei dati gestito in database sarà multimediale.

Si definisce di seguito il concetto di multimedia information retrieval. Data una raccolta di documenti multimediali, l'obiettivo del MIR (Multimedia Information Retrieval) è trovare documenti rilevanti per una necessità di informazione dell'utente. Un documento multimediale è un oggetto informativo complesso, con componenti di diverso tipo, come testo, immagini, video e suono, il tutto in forma digitale. Il MIR è una disciplina scientifica, caratterizzata da molti approcci diversi, che affrontano sostanzialmente lo stesso problema, ma attraverso un aspetto diverso dei documenti multimediali.

I documenti possono essere suddivisi dal punto di vista dell'utente in due categorie principali: semplici e complessi. Un documento è semplice se non può essere ulteriormente scomposto in altri documenti. Le immagini e parti di testo sono in genere documenti semplici. Un documento semplice è una disposizione di simboli che trasportano informazioni attraverso il loro significato, concorrendo così a formare quello che viene chiamato il contenuto del documento.

Nel caso del testo, i simboli sono parole (o le loro frazioni semanticamente significative, come prefissi, etc.), mentre per le immagini i simboli sono colori e trame. Documenti semplici possono quindi essere caratterizzati come aventi due dimensioni parallele: quella della forma (o sintassi, o simbolo) e quella del contenuto (o semantica, o significato).
La forma di un documento semplice dipende dal supporto che trasporta il documento. Al contrario, il significato di un documento semplice è il contesto in cui esso è verificato, ed è quindi indipendente dal supporto. Ad esempio, il significato di una porzione di testo può dipendere da ciò che la circonda o un'immagine dipenda dalla scena in cui si presenta.

I documenti complessi (o semplicemente documenti) sono insiemi strutturati di documenti semplici. Questo porta all'identificazione della struttura come terza dimensione dei documenti. La struttura del

documento è tipicamente una relazione binaria, che può essere rappresentato come un albero avente i documenti semplici come foglie.

Possono esistere strutture più complesse, ad esempio quelle che richiedono un ordinamento tra i “figli” di uno stesso “genitore” (come tra i capitoli di un libro), o quelle aventi un'arità maggiore di 2 (come la sincronizzazione tra diversi flussi di un documento audiovisivo).
Infine, i documenti, semplici o complessi, esistono come entità indipendenti caratterizzate da (meta-) attributi (spesso chiamati metadati), che descrivono le proprietà rilevanti di tali entità. L'insieme di tali attributi viene solitamente chiamato profilo di un documento e costituisce la quarta e ultima dimensione del documento.

In corrispondenza delle quattro dimensioni dei documenti appena introdotte, possono esserci quattro categorie di retrieval, ciascuna delle quali è una proiezione del problema generale del MIR su una dimensione specifica. Inoltre, è possibile, e in alcuni casi desiderabile, combinare diversi tipi di retrieval all'interno della stessa operazione.

Il retrieval basato sulla struttura del documento non porta realmente a una vera scoperta, poiché l'utente deve aver già visto (o essere altrimenti a conoscenza) dei documenti ricercati per poter affermare un predicato sulla loro struttura. Il retrieval basato sul profilo del documento, da un punto di vista puramente logico, non è diverso dal retrieval basato sul contenuto, e in effetti molti schemi di metadati utilizzati per la descrizione del documento (tra questi, il Dublin Core Metadata Set) includono attributi di entrambi i tipi.

4.1. Knowledge Graph

Uno degli approcci che sta attirando l’attenzione di diverse imprese operanti nel campo del Multimedia Information Retrieval, e più in generale all’Information Retrieval, prevedere l’utilizzo di Knowledge Graph (KG). Fondamentalmente un KG può essere immaginato come un repository di identità e delle loro relazioni e attributi e rappresentato tramite l’impiego di una struttura a grafo. Nei moderni approcci di accesso alle informazioni, i KG sono ubiqui. Nello specifico, nel recupero delle informazioni (IR) i KG sono fondamentali per consentire la ricerca semantica. Ci sono due tratti distintivi della ricerca semantica in un contesto IR: (1) andare oltre i "*dieci link blu*" forniti da un motore di ricerca tradizionale e restituire risultati rilevanti di qualsiasi tipo (come risposte dirette, actionable entities o relazioni) e (2) comprendere le domande e i documenti, migliorando la corrispondenza tra loro con le relazioni pertinenti. Idealmente, un motore di ricerca è in grado di rispondere direttamente al bisogno di informazioni di un utente, o almeno di generare possibili interpretazioni del bisogno di informazioni espresso attraverso la query.

Per raggiungere questo obiettivo, sono necessari vari componenti entity-oriented che risolvono problemi specifici in diverse fasi della pipeline di recupero delle informazioni, inclusa l'identificazione delle entità nella query, l'identificazione delle entità nei documenti e metodi che sfruttano le informazioni sull'entità e sulle relazioni per identificare gli elementi rilevanti da recuperare. Nonostante il fatto che IR e KG siano sempre più correlate nel contesto del web e dei motori di ricerca specifici per dominio, non c'è un approccio ampiamente condiviso nella letteratura relativamente all’utilizzo dei KG nell’ambito dell’IR e viceversa.

Prima di entrare nel cuore della trattazione, è importante introdurre un insieme unificato di definizioni e terminologia. Innanzitutto, si definisce il concetto di entità:

- Un'entità *e* è un oggetto atomico identificabile che ha un'esistenza distinta e indipendente.
- Un'entità denominata è un'entità specifica per la quale è possibile utilizzare uno o più nomi propri per fare riferimento ad essa.
- Un'entità di tipo t è un insieme di classi appropriato per un'entità basata su una gerarchia di classi predefinita.
- Una menzione è un segmento di testo che fa riferimento a un'entità.
- Un tipo di relazione è un tipo di connessione tra due entità.
- Un attributo è una specifica caratteristica o proprietà di un'entità o di una relazione, con zero o più valori.
- Una Knowledge Base è un repository di entità, con informazioni relative alla loro relazione e ai loro attributi in un formato (semi-) strutturato.
- Le entità sono tipicamente rappresentate come nodi, mentre le relazioni come archi.
- Un profilo di un'entità è una descrizione testuale di una entità.

Di seguito, un'esemplificazione dei concetti prima descritti:

- Entity: city
- named entity: London
- entity type: city
- mention: London
- relation type: capitalOf (relation) (London, capitalOf, UnitedKingdom)
- attribute city:population
- knowledge base: Wikipedia
- knowledge graph: Wikidata

4.1.1. Entity Linking

L'obiettivo dell'entity linking è fornire una forma di "semantic grounding" di un testo, utilizzando entità in un Knowledge Graph e determinando quali intervalli testuali si riferiscono a quali entità specifiche. L'entity linking ha le sue radici nei domini del Natural Language Processing e nei database (dove è noto come record linkage) (Meij et al., 2013; Dietz et al., 2018).
L'entity linking è formalmente definito come segue: Dato un testo, rileva i segmenti di entità menzionate *m* all'interno del testo e collegarli ad un'entità e in un knowledge graph.

4.1.2. Riconoscimento e classificazione di un'entità

Il riconoscimento delle entità nel testo è un problema ben noto. Il task MUC-6 (Grishman e Sundheim, 1996) ha introdotto il task di estrazione di named entity e, successivamente, la Computational Natural Language Learning (CoNLL) (Tjong Kim Sang e De Meulder, 2003) e ACE (Doddington et al. , 2004) hanno spinto ulteriormente la ricerca in questo settore. Il task del riconoscimento di named entity è formalmente definito come segue: Dato un segmento di testo *s*, il task di riconoscimento dell'entità è quello di rilevare segmenti di entità menzionate *m* all'interno di *s*. Dato un sistema di classificazione del tipo T e un'entità menzionata *m* all'interno di un segmento di testo *s*, l'attività di classificazione della menzione dell'entità è decidere se *m* appartiene a un tipo $t \in T$ e, in tal caso, quale tipo. Inizialmente, il

riconoscimento di named entity si concentrava sulla classificazione delle entità in tipi di entità abbastanza generici come persona, organizzazione, posizione e così via. Successivamente, sono state proposte gerarchie di classi più dettagliate, ad esempio da Sekine e Nobata (2004), che considerano 150 tipi di entità "estesi".

4.1.3. Approcci basati su KG per l'Information Retrieval

L'ascesa e la diffusione dei KGs ha portato un sempre maggior numero di motori di ricerca commerciali ad incorporare i dati delle entità contenute nel KG per migliorare i risultati della ricerca. Ad esempio, Google sfrutta i dati estratti dal Google Knowledge Graph, mentre Facebook esegue le operazioni di ricerca sulle entità tramite il Graph Search. La capacità dei KGs di migliorare la comprensione di espressioni in linguaggio naturale supporta inoltre i sistemi di comprensioni di query e documenti. Alcune linee di ricerca emergenti si stanno concentrando sull'esplorazione del potenziale dei KG per l'information retrieval.

Le tecniche di ricerca "entity-oriented" si sono diffuse con lo sviluppo di KG su larga scala. Esistono diversi modi possibili per utilizzare la conoscenza semantica dei KG in diversi componenti, come la rappresentazione della query, la rappresentazione del documento e la classificazione di un sistema di ricerca. In particolare, la query representation può essere migliorata introducendo le entità correlate e i loro testi per espandere le query.

In generale, le entità prese da un KG possono essere sfruttate all'interno di un sistema IR per aiutare a migliorare la comprensione delle intenzioni, delle query e dei documenti di un utente, al di là di ciò che può essere ottenuto attraverso l'uso esclusivo di token di parole.

Avere a disposizione un KG ci consente anche di rispondere a esigenze di informazioni a cui potrebbe essere più chiaro e agevole per l'utente ottenere risposte dirette, invece di ottenere un elenco classificato di documenti. Allo stesso modo, i KG consentono l'esplorazione delle entità correlate menzionate in una raccolta di documenti o nella pagina dei risultati di un motore di ricerca. Infine, KG può aiutare a fornire spiegazioni di entità e relazioni in funzione del contesto, per supportare ulteriormente l'utente.
In sintesi, i KG ci consentono di migliorare l'esperienza di ricerca dell'utente attraverso una migliore comprensione delle intenzioni, delle query e dei documenti, tramite risposte dirette e tramite strumenti di esplorazione avanzati.

Document retrieval

Rispetto alla vasta letteratura sull'information retrieval in generale, e anche a task più recenti come il l'entity linking, ci sono relativamente pochi lavori che sfruttano i knowledge graph per migliorare il recupero dei documenti.

La ragione principale di ciò è che capire esattamente come sfruttare efficacemente le annotazioni e il testo delle entità insieme per migliorare il document retrieval ad hoc è ancora una questione irrisolta. Definiamo prima formalmente il document retrieval come segue: Il Document Retrieval è il processo automatico di produzione di un elenco di documenti, sulla base della loro rilevanza e appartenenti a una

collezione di documenti D, che sono rilevanti per una query q, confrontando la richiesta dell'utente con un indice prodotto automaticamente del contenuto testuale dei documenti.

Gli approcci all'information retrieval che sfruttano le informazioni entity-oriented dai KG possono essere raggruppati in modelli di fattori latenti expansion-based, language modeling, e approcci al deep learning:

- Gli approcci expansion-based incorporano esplicitamente informazioni entity based, come caratteristiche nel processo di retrieval. Al contrario, gli approcci a fattori latenti non tentano di arricchire le rappresentazioni di query o documenti direttamente da un KG, ma mirano a estrarre concetti inerenti a query e documenti.
- Gli approcci di modellazione linguistica considerano le informazioni semantiche quando vengono calcolati i punteggi di retrieval utilizzando la modellazione linguistica.
- I metodi di deep learning, in questo contesto, possono sfruttare gli embeddings basati su KG per migliorare le rappresentazioni di query e documenti, o cambiare la funzione di matching, per incorporare queste rappresentazioni vettoriali.

Entity retrieval

L'Entity retrieval ha guadagnato una sempre maggiore attenzione a partire dalla presentazione del lavoro di Craswell et al., 2005 (TREC). Da allora, ci sono state presentate varie soluzioni in contesti diversi, come INEX (de Vries et al., 2008), ed anche setting alternativi, ad esempio, classificare le entità appena inserite in raccolte di documenti, knowledge graph, od in entrambi.
In linea di principio, è possibile definire il problema dell'entity retrieval come segue: data una query q e una collezione di documenti D, entity retrieval consente di ritrovare e assegnare un rank ad entità che appaiono a ciascun documento $d \in D$ in base alla loro rilevanza per q.

Si formalizza di seguito un altro setup di entity retrieval, dove le entità candidate sono ottenute da un KG. Data una query q e un Knowledge Graph KG, l'entity retrieval consente di ritrovare e assegnare un rank ad entità in KGs in base alla loro rilevanza per q.

Le forme di entity retrieval considerate in letteratura includono il term-based entity retrieval, l'ad-hoc object retrieval, e il list retrieval:

- L'ad hoc entity retrieval consiste nel rispondere a una query in testo libero con una lista ordinata di entità;
- La term-based entity retrieval consiste di due passi:
    - Innanzitutto i documenti vengono ritrovati, successivamente vengono estratte le feature da tali documenti;
    - Rappresentando le entità come esse stesse documenti, concatenando tutti i documenti per una entità.
- La list retrieval non è di interesse per la presente sezione.

Qui ci concentriamo sui primi due moduli e consideriamo il list retrieval come una forma specifica di entity recommendation, da discutere nella sezione successiva.

Approcci all'entity retrieval

Gli approcci all'entity retrieval si raggruppano in quelli di language modelling ed approcci neurali. Questi ultimi in particolare sono quelli più strettamente collegati con entity retrieval basata sull'analisi di termini, in cui non sono fornite rappresentazioni esplicite delle entità.
Gli approcci di language modelling comprendono tipicamente metodi definiti intorno al problema dell'*expert finding*, e per lo più si basano su varie estensioni dei metodi di language modelling classici. Al contrario, gli approcci neurali puntano ad apprendere una distributed word representation di entità, le quali sono ottimizzate per la ricerca.
In generale, nel retrieval di oggetti ad-hoc, le entità sono considerate come oggetti con attributi e relazioni, quindi possono essere rappresentate come documenti multi-field.

Entity recommendation

Un altro task entity-oriented consiste nel raccomandare entità correlate in risposta ad una query di testo ed un insieme di entità. Una realizzazione di questo task può essere trovata in tutti i principali motori di ricerca web, dove vengono mostrate le entità relative a un'entità rilevante per la query.
Ci riferiamo a questo task come entity recommendation. In letteratura, l'attività viene talvolta definita anche ricerca di entità correlate (Bron et al., 2010; Foley et al., 2016; Kang et al., 2011).
Le origini di questo task possono essere ricondotte al lavoro sulla ricerca di entità correlate, introdotto al TREC 2009 (Balog et al., 2009b).
In questa versione, l'espressione del bisogno di informazioni è tipicamente accompagnata da una descrizione delle entità correlate attese e anche dal tipo di entità atteso. Nelle versioni successive, l'input è solo un'entità o un'entità più parole di contesto.
Definizione di Entity Recomendation: Da una query q (dove q può avere la forma di un'entità, o di un'entità più alcune keyword di contesto), viene conferito un punteggio a ogn entità e $\in$ KG, sulla base di quanto essa sia correlata alla query.

Approcci all'entity recommendation

Esistono tre approcci generali all'entity recommendation:

- Euristico,
- Comportamentale,
- Basato su grafo.

Dal momento che gli approcci discussi sono disegnati per domini e setting differenti, essi non sono tipicamente comparabili.
Classifichiamo gli approcci euristici come approcci che non modellano la entity recommandation direttamente come un problema di apprendimento, ma si basano su associazioni statistiche delle entità stimate da una sorgente dati.
In contrasto, gli approcci comportamentali utilizzano segnali derivati dalle interazioni degli utenti o dai feedback, per generare le raccomandazioni.

Dell'ultimo gruppo fanno parte gli approcci che si basano sulle relazioni semantiche, senza alcun esplicito feedback degli utenti, e vengono indicati come approcci basati su grafi.

Entity relationship explanation

L'Entity relationship explanation è un task emergente e relativamente nuovo. La motivazione principale è che per descrivere coppie di entità sono necessarie spiegazioni, "*percorsi*" tra entità, come anche relazioni osservate, ad esempio, in log di query.

In particolare, una Entity relationship explanation può essere definita come segue. Data una coppia di entità e ed e_0, una una Entity relationship explanation fornisce una spiegazione, ad esempio una descrizione testuale, supportata da un KG, di come la coppia di entità sia correlata.

Tra gli approcci all'Entity relationship explanation, di particolare interesse sono i seguenti:

- instance-based explanations, che ha come obiettivo quello di fornire una spiegazione di una relazione, restituendo un set di entità correlate;
- description ranking, i cui approcci forniscono descrizioni testuali candidate di una relazione e generano una classifica delle possibili spiegazioni.

5. Piattaforme, strumenti e tecnologie

Come anticipato nei capitoli precedenti, profilare accuratamente gli utenti così da fornire raccomandazioni di prodotti che incontrino i loro gusti, predirne le future ricerche, adattare la visualizzazione dei contenuti in base ai loro principali interessi (identificando micro-generi) e migliorare in questo modo la loro esperienza nella fruizione dei servizi è un task di interesse per molte grandi compagnie, specialmente se si pensa ad aziende operanti nel campo del delivery di contenuti multimediali (quali ad esempio film, musica, etc.).

Non di meno, la crescita vertiginosa della quantità e della varietà dei contenuti disponibili sul web ostacola l'identificazione di elementi potenzialmente di interesse per l'utente. Numerosi sono gli ambiti nei quali tale sovraccarico di informazione pregiudica l'efficacia della ricerca delle informazioni dell'utente. Si consideri ad esempio piattaforme di streaming di film o di musica: In tali contesti, avere a disposizione strumenti che da un lato supportino l'utente nella ricerca dei contenuti voluti e, dall'altro, suggeriscano a quest'ultimo, proattivamente, prodotti a lui potenzialmente non noti ma di suo gradimento, è di importanza decisiva per garantire servizi soddisfacenti per l'utente e massimizzare i profitti per i provider, in termini di fidelizzazione della clientela e maggiore volume di vendite.

Ad oggi, strumenti come ContentWise (engine usato per profilare utenti di servizi IPTV e suggerire contenuti d'interesse) sono già impiegati da grandi società come Sky, Mediaset, TIM, RAI, BBC, Turner Broadcasting, per migliorare la qualità dei servizi offerti ed incrementare il loro bacino di utenti.

In questo capitolo sono prese in esame piattaforme, strumenti e tecnologie che supportano in modi diversi la fruizione dei contenuti digitali, in particolare focalizzandosi sui tre task principali di nostro interesse: *(i)* sistemi di raccomandazione, *(ii)* chatbot e *(iii)* sistemi di ricerca avanzati e knowledge graph.

Di seguito si introducono piattaforme e tecnologie attualmente disponibili associate ai tre campi applicativi prima introdotti (recommendation, chat bot, information retrieval e knowledge graph).

5.1. Recommender Systems

Il problema del sovraccarico delle informazioni, e quindi della scarsa rintracciabilità delle informazioni, è stato in prima istanza affrontato per mezzo di sistemi di Information Retrieval, come i motori di ricerca, che hanno parzialmente risolto questo problema, ma l'assegnazione di priorità nella presentazione dei risultati e la personalizzazione delle informazioni (ovvero l'associazione dei contenuti disponibili agli interessi e alle preferenze dell'utente) non viene realizzata da tali sistemi. Tale situazione ha creato l'esigenza e la domanda di avere a disposizione sistemi che potessero supportare l'utente nella scelta dei contenuti, e potessero allo stesso tempo suggerirne di nuovi. Tale esigenza ha portato quindi alla progettazione e allo sviluppo di sistemi di raccomandazione che, come detto in precedenza, sono sistemi di filtering delle informazioni. L'obiettivo è quindi quello di filtrare frammenti di informazioni potenzialmente di interesse, da una grande quantità di informazioni generate dinamicamente in base alle preferenze dell'utente, e all'interesse o al comportamento osservato su uno o più elementi. In altri termini, semplificando, Il sistema di raccomandazione ha la capacità di prevedere se un particolare utente preferirebbe o meno un articolo, in base al profilo ad esso attribuito. I sistemi di raccomandazione offrono sensibili vantaggi sia per i fornitori di servizi che per gli utenti. Ad esempio, essi consentono di

ridurre i costi di transazione nella ricerca e la selezione di articoli in un ambiente di shopping online. Anche in tale esempio, esistono vantaggi anche per i fornitori di servizi: nell'ambito del commercio elettronico, infatti, i sistemi di raccomandazione aumentano i ricavi, in quanto sono mezzi efficaci per vendere più prodotti. I sistemi di raccomandazione hanno inoltre dimostrato di migliorare il processo. Altro contesto di esempio è quello delle biblioteche scientifiche, nell'ambito delle quali i sistemi di raccomandazione supportano gli utenti consentendo loro di andare oltre le ricerche "classiche" nel catalogo. L'importanza strategica di avere a disposizione sistemi di raccomandazione efficaci ha come effetto quello di aumentare la richiesta di tecniche di raccomandazione efficienti e accurate, portando allo studio e l'ideazione di approcci innovativi, di cui si darà conto più avanti.

Di seguito viene proposta una panoramica su alcuni tra i più interessanti sistemi di raccomandazione disponibili, si in forma di prodotto industriale, sia nella forma di prototipo di ricerca. Sono citate alcune tecniche (collaborative filtering, content-based filtering, etc.) già descritte nel capitolo 2. I sistemi qui presi in considerazione sono dunque i seguenti:

- Il primo sistema di raccomandazione che viene presentato è **Ringo**, un sistema di raccomandazione di content filtering user based, che mette a disposizione raccomandazioni relative ad album musicali e artisti. In Ringo, quando un utente accede inizialmente al sistema, gli viene fornito una lista di 125 artisti, da valutare in funzione delle preferenze nell'ascolto dei brani. Tale lista è costituita da due differenti sezioni. La prima è contiene gli artisti più spesso valutati, e questo consente all'utente di valutare artisti in maniera similare a quanto hanno fatto altri utenti. I livelli di similarità nelle preferenze con gli altri utenti consentono di fornire importanti informazioni per la profilazione. La seconda sessione viene generata successivamente sulla base di una selezione random di item appartenenti alla matrice user-item, in modo tale che tutti gli artisti e gli album ricadano nel tempo nelle valutazioni iniziali degli utenti.

- **MovieLens** è un sistema di raccomandazione e una comunità virtuale che raccomanda film per i suoi utenti, basate sulle loro preferenze, usando la tecnica del collaborative filtering applicata alle valutazioni e alla recensioni dei film da parte degli utenti. E' stato creato nel 1997 da GroupLens Research, un laboratorio di ricerca del dipartimento di Computer Science and Engineering dell'Università del Minnesota. In particolare, MovieLens basa le sue raccomandazioni sugli input messi a disposizione dagli utenti di un sito web, tra le quali rivestono grande importanza (ma non esclusiva) le valutazioni dei film. MovieLens utilizza diversi algoritmi di raccomandazione, inclusi algoritmi di collaborative filtering, come quelli di tipo item-item e quelli di tipo user-user. MovieLens, inoltre, al fine di affrontare il problema Gold Start, per i suoi nuovi utenti, USA metodi di digitazione delle preferenze che supportino la scelta dell'utente. Inoltre, il sistema chiede ai nuovi utenti di valutare quanto gradiscano vedere di vari generi di film, per esempio film con humor Dark oppure commedie romantiche. Le preferenze memorizzate per mezzo di queste scelte consentono sistema di fare raccomandazioni iniziali, anche prima che l'utente abbia valutato un ampio numero di film sul sito web. Sulla base dei dati iniziali prima raccolti, e per ogni utente, MovieLens predice come essi valuteranno ogni film presente nei propri database. Sulla base di queste valutazioni predette il sistema raccomanda film che sulla base delle predizioni prima fatte dovrebbero essere meglio valutati dall'utente. MovieLens incoraggia gli utenti a fare un maggior numero possibile di

valutazione, in modo tale che le raccomandazioni siano il accurate possibile, e questo perché tali raccomandazioni si baserebbero su un set migliore di campioni relativi ai gusti degli utenti. Tuttavia, il meccanismo di rating che caratterizza movielens non è sempre particolarmente efficace, dal momento che più del 20% dei film nelle liste generate dal sistema hanno un numero così basso di rating che gli algoritmi raccomandazione non possono fare predizioni accurate. MovieLens, per le sue raccomandazioni, oltre ai rating degli utenti, utilizza anche i metadati dei film, per generare similitudini tra film, e sfruttare tali similitudini per migliorare le raccomandazioni.

- **Amazon.com** è un esempio di engine di raccomandazioni di e-commerce che usa tecniche scalabili di collaborative filtering item-to-item, per raccomandare i propri prodotti (tra cui anche film) a diversi utenti. L'algoritmo utilizzato scala indipendentemente dal numero di utenti e dal numero di item contenuti nel database. Amazon.com usa una tecnica di raccolta delle informazioni esplicita per ottenere informazioni dagli utenti. Il sistema predice gli interessi degli utenti sulla base degli item che ha valutato. Il sistema successivamente confronta i pattern di navigazione degli utenti sul sistema stesso, e decide l'oggetto da raccomandare.

- **Netflix** ha avuto un notevole impatto nel campo dello sviluppo dei sistemi di raccomandazione. Si tratta di una piattaforma per la fruizione di contenuti multimediali d'intrattenimento che offre legalmente la visione di prodotti come film, serie TV, show, documentari in streaming su Internet, in modalità on demand, che promosse il noto *Netflix Prize*, una competizione che mirava allo sviluppo del miglior modello di raccomandazione di film basato su algoritmi di collaborative filtering. Ad oggi il sistema di raccomandazione utilizzato dalla piattaforma è molto complesso ed impiega una varietà di modelli: *(i)* Personalised Video Ranking (PVR), un algoritmo general purpose che filtra il catalogo tramite un criterio specificato e che combina side-information dell'utente; *(ii)* Top-N Video Ranker, il quale lavora in maniera simile a PVR eccetto il fatto che preferisce la "testa" dell'intero catalogo; *(iii)* Trending Now Ranker, un algoritmo in grado di catturare trend temporali e *(iv)* Continue Watching Ranker, il quale analizza i contenuti che l'utente ha iniziato a visionare ma che non ha completato.

- **Disney+** è una piattaforma emergente per la fruizione di contenuti multimediali. La piattaforma è in grado di raccogliere diverse informazioni comportamentali trami *ESPN +* un servizio di abbonamento di streaming video over-the-top disponibile negli Stati Uniti. La piattaforma impiega strumenti di machine learning per personalizzare i consigli sui contenuti dei suoi servizi di streaming in base all'ora del giorno e a ciò che gli utenti stanno facendo in un determinato momento. Il sistema di raccomandazione apprende cosa le persone amano guardare in diversi momenti della giornata sfruttando le informazioni raccolte da ESPN+, e le prossime evoluzioni consentiranno di utilizzare altri segnali, ad esempio se l'utente sta visionando il contenuto in streaming da TV o da smartphone, per suggerire la raccomandazione.

- **Spotify,** è un servizio di streaming on demand di una selezione di brani di varie case discografiche ed etichette indipendenti, incluse Sony, EMI, Warner Music Group e Universal. L'home page del servizio è governata da un sistema di AI chiamato *Bandits for Recommendations as Treatments* o semplicemente noto come BaRT. BaRT è la principale ragione perché gli utenti spotify non cercano delle playlist appropriate ai loro gusti.

Quando si impiega spotify, esso tipicamente inizia consigliando brani in base alle precedenti attività di ascolto. Ma BaRT introduce anche musica nuova che verosimilmente potrà piacere agli utenti così da estrarli dallo solito loop di ascolto. Due concetti principali entrano in gioco con BaRT: exploiting ed exploring. La combinazione di "exploit" ed "explore" è la chiave della raccomandazione di Spotify. Durante l' exploiting, Spotify utilizza ogni attività prodotta da un utente. L'exploit di solito fa uso della cronologia di ascolto dell'utente, dei brani saltati, delle playlist che l'utente ha creato, dell'attività sui social media sulle piattaforme e persino della posizione di uno per consigliare la musica. Durante l'esplorazione, Spotify studia il resto del mondo. Inizia a cercare playlist e artisti simili ai gusti di ascolto dell'utente ed osserva anche la popolarità degli artisti di cui l'utente potrebbe essere ignaro o altre opere correlate. In ultimo, una regola importante adottata dal meccanismo di raccomandazione è quella dei "30 secondi", se l'ascolto è durato più di 30 secondi allora la piattaforma considera l'item per successive raccomandazioni.

5.2. Chatbots

L'interazione tra esseri umani e sistemi informatici si sta sempre più spostando verso interfacce basate sul linguaggio naturale. Come visto in precedenza, negli ultimi anni il linguaggio naturale è stato visto come una tecnologia abilitante per la personalizzazione, la quale consente a ciascun utente di interagire con il sistema con le proprie parole piuttosto che utilizzare un limitato numero di meccanismi di interazione predefiniti. Una conversazione basata su testo o voce viene solitamente avviata dall'utente che pone una domanda utilizzando il linguaggio naturale ed il chatbot risponde anch'esso alla domanda utilizzando il linguaggio naturale. Questi strumenti sono progettati principalmente per condurre una conversazione con gli esseri umani utilizzando dialoghi in linguaggio naturale tramite tecniche di AI basate su parlato o testo, in modo tale che gli esseri umani possano comunicare facilmente con essi. Molti progressi sono stati compiuti dalle prime piattaforme prototipali che erano pensate per lo più per l'intrattenimento. Sebbene i chatbot moderni inglobino strumenti di AI per rispondere a domande complesse formulate dagli utenti, gli attuali sistemi stato dell'arte sono ancora molto lontani dall'essere in grado di avere conversazioni coerenti, contestuali e naturali con gli esseri umani.

I chatbot generalmente sono progettati sia per svolgere alcuni compiti specifici, chiamati task oriented chatbot, sia per "*chiacchierare*" in questo caso chiamati non-task oriented chatbot, usati per divertimento e intrattenimento.

I chatbot sono utilizzati dalle aziende per aumentare le vendite e migliorare l'assistenza clienti. Di seguito sono stati presi in esame i principali strumenti sviluppati ed attualmente impiegati in quest'ambito considerando tre importanti aspetti: *tipo*, *piattaforma* e *licenza di utilizzo*.

Il *tipo* definisce il modo in cui l'utente interagisce con il sistema di chatbot AI. Tutti i sistemi dispongono di chatbot conversazionali con intelligenza artificiale e/o logica e possono operare in maniera multiforme. Di seguiti i principali meccanismi di interazione:

- *Testo* = si clicca sul pulsante e si digita l'input
- *Live Chat* = sono in grado di gestire una live chat con un essere umano

- *Voce* = utilizzano la voce, spesso tramite un telefono cellulare. La maggior parte di questi si connette a sistemi di AI vocale di Google Assistant o Amazon Alexa.

La *piattaforma* definisce su che tipo di piattaforma il chatbot è impiegato. L'elenco considera alcune delle principali ma non ha la pretesa di essere esaustivo:

- *Sito web* = si interfaccia tramite un sito web desktop e mobile
- *Facebook* = utilizza Facebook Messenger, l'utente deve avere un account Facebook con Messenger attivo
- *SMS Text* = utilizza SMS mobile texting
- *Slack, Kik, Twitter, Discord ed altre reti* = Il sistema funziona su altri social e reti di comunicazione
- *Piattaforme di e-commerce* = funzionano o si integrano con una piattaforma di e-commerce

La licenza indica semplicemente se il chatbot è rilasciato a pagamento oppure sotto licenza free e/o open source.

Di seguito in tabella sono presentati alcuni dei principali sistemi attualmente impiegati commercialmente in vari ambiti applicativi, in particolare evidenziando gli aspetti di sopra riportati.

Sistema	Tipo	Piattaforma	Licenza
Ochatbot	Testo, Live Chat	Siti Web, Facebook ed E-Commerce (ad Es., Shopify,	Free, Commercial
IBM Watson	Testo, Voce	Siti Web, Facebook, SMS, Slack, Alexa	Free, Commercial
Intercom	Testo, Live Chat	Siti Web, Facebook, Slack	priced based on interactions)
Smooch	Testo, Live Chat	Siti Web, Facebook	Commercial
Pullstring	Voce	Alexa, Google	Requires contact
AIVO	Testo, Voce	Siti Web, Facebook, SMS	Commercial
Chatfuel	Testo, Live Chat	Facebook	Free, Commercial
Engati	Testo, Live Chat	Siti Web, Facebook, ECommerce	Free, Commercial
Engagr	Testo	Siti Web	Commercial
Drift	Testo, Live Chat	Siti Web	Free, Commercial
Botsify	Testo, Live Chat	WordPress, Alexa, Shopify	Free, Commercial
ManyChat	Testo	Facebook	Free, Commercial
Dexter	Testo, Live Chat	Slack	Free, Commercial (Enterprise pricing
LivePerson	Testo, Live Chat	Siti Web, SMS, Facebook	Requires contact
BotCore	Testo, Voce	SMS	Requires contact
AI Assist	Testo	Email	Requires contact
Amazon Lex	Testo, Voce	SMS	Priced per request
Autochat	Testo, Live Chat	Siti Web	Requires contact
Avaamo	Testo, Voce	Siti Web, Facebook, Skype	Requires contact
Dialogflow	Testo, Voce	Siti Web, Facebook, Alexa, Google Assistant	Free, Price per connection
Clustaar	Testo, Live Chat	Siti Web, Facebook, Slack	Commercial
FlowXO	Testo	Facebook, Slack	Commercial
Helloumi	Testo, Live Chat	Siti Web, Facebook	Requires contact
Botanic Technologies	Test, Voce	Siti Web, Facebook, Slack	Requires contact
Chatpath	Testo, Voce, Live Chat	Siti Web, SMS	Requires contact
Conversy	Testo, Live Chat	Siti Web, Facebook, Slack	Free, Commercial
Digital Genius	Testo, Live Chat	Salesforce, Zendesk	Commercial
Fandango	Testo	Facebook, Alexa	Free, Commercial
Flamingo	Testo, Voce	SMS	Requires contact
Help Shift	Testo	Siti Web, In-app	Commercial
Hu:toma	Testo, Voce	Siti Web, Facebook, Slack, Alexa, Google Assistant	Open source, paid options
BotsCrew	Testo, Voce, Live Chat	Siti Web, Slack, Facebook	Requires contact
ItsAlive	Testo	Facebook	Free, Commercial (Enterprise pricing
Kore.ai	Testo, Voce, Live Chat	Siti Web, Facebook, SMS, Slack	Requires contact
Landbot	Testo	Siti Web	Free, Commercial
Mobile Monkey	Testo, Live Chat	Facebook	Free, Commercial
MSG.ai	Testo, Voce, Live Chat	Siti Web, Facebook, SMS, E-Commerce	Requires contact
NextIT	Testo	Siti Web, Facebook	Requires contact
Octane AI	Testo	Facebook, ecommerce	Commercial
Pandorabots	Testo, Voce	Siti Web, Facebook, SMS, Slack, ecommerce	Free, requires contact
Pypestream	Testo	Siti Web, Facebook, SMS	Requires contact
Quriobot	Testo	Siti Web, Facebook, SMS, Slack	Free, Commercial
Reply.ai	Testo, Voce, Live Chat	Siti Web, Facebook, Alexa	Requires contact
Salesmachine	Testo	Slack	Commercial
Selekt	Testo	Siti Web, ecommerce	Requires contact
Semantic Machines	Voce	Alexa, Google, Cortana, E-Commerce	Requires contact
Spotify's bot	Testo	Facebook	Commercial
Snatchbot	Testo, Live Chat	Siti Web, Facebook	Free, Commercial
Twyla	Testo, Voce, Live Chat	Siti Web	Requires contact
Web Spiders	Testo, Live Chat	Siti Web, Facebook, Slack	Requires contact
Wit.ai	Voce, Testo	Facebook, Alexa	Free
Wizeline	Testo, Voce, Live Chat	Siti Web, Facebook, SMS Slack, Alexa, Google	Requires contact
Zyratalk	Testo, Voce	Siti Web, Facebook, SMS, Slack, Alexa	Free

Tabella 1: Analisi comparativa dei più diffusi sistemi chatbot

In dettaglio, di particolare interesse per il nostro caso applicativo sono i seguenti sistemi:

- *Fandango*. Fandango è un bot di Facebook Messenger di Fandango e consente di visionare trailer di film, trovare teatri locali e vedere le tendenze della settimana. Semplicemente inserendo tua città o codice postale e il chatbot mostrerà cosa sta suonando nelle vicinanze, quando e indirizzerà a una pagina dove sia possibile acquistare i biglietti. Questo strumento è particolarmente apprezzato per la sua interfaccia intuitiva e facile da usare, per la sua integrazione diretta con Facebook Messenger e per il modo in cui evidenzia film, trailer e i cinema in "*tendenza*".

- *Amazon Lex*. Amazon Lex è un servizio per la creazione di interfacce conversazionali in qualsiasi applicazione utilizzando voce e testo. Integra un sistema di riconoscimento vocale automatico che converte il parlato in testo, e di comprensione del linguaggio naturale per riconoscere il significato del testo. Lex integra al suo interno la stessa tecnologia basata su deep learning di Amazon Alexa e ciò consente di sviluppare di creare applicazioni con esperienze utente altamente coinvolgenti e interazioni conversazionali realistiche.

- *Kore.AI* . Kore.ai Conversational Platform è una piattaforma end-to-end per creare, addestrare, analizzare e gestire bot per le aziende. Fornisce strumenti, processi e metodi per sviluppare chatbot o assistenti virtuali pronti per il business. La piattaforma fornisce alle aziende tutti i componenti necessari per progettare, costruire, testare e distribuire chatbot basati su AI. La piattaforma fornisce componenti tecniche e aziendali per sviluppare bot intelligenti e gestirli durante tutto il loro ciclo di vita. Il core della piattaforma è un motore NLP che combina più metodi di intelligenza per comprendere le espressioni e gli intenti umani, inclusi sentimenti ed emozioni. I bot sviluppati sulla piattaforma Kore.ai funzionano su più canali di comunicazione.

- *Spotify Facebook Messenger Bot*. Il bot Facebook Messenger di Spotify consente all'utilizzatore di cercare, ascoltare e condividere musica. Una volta loggato, l'utente riceverà suggerimenti sulle playlist in base al suo umore, a cosa sts facendo o al genere di musica che desidera. Il bot include inoltre funzioni per la ricerca, suggerimento, e condivisione di clip (30 secondi) di un brano all'interno di Messenger o l'avvio di Spotify per ascoltare l'intera canzone.

- *IBM Watson.* IBM Watson può essere definito come un sistema di question answering capace di rispondere a domande poste in linguaggio naturale. Questo strumento combina metodi di intelligenza artificiale ed un sofisticato software analitico per rispondere efficacemente ed efficientemente alla domande proposte dall'utilizzatore. L'Assistente può essere utilizzato per una varietà di canali, inclusi dispositivi mobili, piattaforme di messaggistica e robot.

- *Engati*. La piattaforma engati offre sia un chatbot che una chat dal vivo. Ha la capacità di "*sospendere*" la risposta di un bot in modo che un essere umano possa prendere il controllo della conversazione. Funziona su piattaforme social e siti Web e offre supporto multilingue.

- *Engagr*. Engagr fornisce diversi template di chat per siti Web che si concentrano sulla generazione di lead e sulle conversioni. La caratteristica principale della piattaforma è che il testo della chat può essere letto ai visitatori del sito web utilizzando una voce preregistrata o automatica.

- *Botcore*. BotCore è un acceleratore che consente di creare bot conversazionali personalizzati basati sull'intelligenza artificiale. Può sfruttare qualsiasi servizio di intelligenza artificiale disponibile oggi e scalerà anche per i servizi futuri. È fully deployable in Microsoft Azure e sfrutta molte delle funzionalità disponibili in esso. La piattaforma è attualmente utilizzata da diverse aziende, PMI e startup per creare, distribuire e gestire chatbot all'interno dell'organizzazione. Le principali caratteristiche della piattaforma sono un'ampia Knowledge Base, un sistema conversazionale avanzato, integrazione di sistemi di sicurezza e per l'amministrazione del modulo.

- *Dialogflow*. Può essere personalizzata per intraprendere conversazioni su diversi argomenti tra cui film e musica. Dialogflow è una piattaforma di comprensione del linguaggio naturale utilizzata per progettare ed integrare un'interfaccia utente conversazionale in app mobili, applicazioni web, dispositivi, bot, sistemi di risposta vocale interattivi e usi correlati. Si basa sul Compute Engine di google.

- *Botsify*. Botsify è un'altra opzione popolare sulla scena dei chatbot ed è progettato per aiutare a creare chatbot intelligenti per siti Web, Slack e Facebook Messenger. Se il chatbot non può rispondere alla domanda di un utente, la conversazione viene trasferita a un essere umano tramite e-mail. Si tratta di uno strumento versatile e personalizzabile che si integra altresì con shopify, per cui molto utile anche per l'e-commerce.

- *Semantic Machines*. Semantic Machines è un'interfaccia AI conversazionale per Siri, Google e Cortana per l'e-commerce e il business. Si tratta di uno strumento molto interessante recentemente acquisito da Microsoft. Lo strumento combina tecniche di AI con metodi di NLP avanzati per fornire un'esperienza estremamente efficace.

- *Wit.AI*. Wit è NL interface per applicazioni in grado di trasformare frasi in dati strutturati. Wit.ai consente di creare bot basati su voce e testo con cui gli utenti possono chattare sulla loro piattaforma di messaggistica preferita. Con Wit.ai è possibili sviluppare applicazioni con cui è possibile sia parlare che scambiare messaggi.

5.3. Sistemi di ricerca avanzati e knowledge graph

Negli ultimi anni, i Knowledge Graph (KG) sono diventati la base di molti sistemi informativi che richiedono l'accesso a conoscenza strutturata. Il concetto di Web semantico può essere fatto risalire alla ricerca di BernersLee [Ber01] nel 2001. Nel suo lavoro, Berners-Lee hanno suggerito che gli standard tecnici come Uniform Resource Identifier (URI), Resource Description Framework (RDF) e Web Ontology Language (OWL) dovrebbero essere promossi e sviluppati. Alcune ricerche hanno contribuito a promuovere la rappresentazione della conoscenza basata su grafi utilizzando lo standard RDF. I nodi in tali grafi rappresentano entità e sono collegati da archi che rappresentano relazioni. Gli insiemi di relazioni possono essere organizzati in uno schema o in un'ontologia che definisce la loro correlatività e le restrizioni del loro utilizzo. Il concetto di Linked Data [Biz09] è stato definito nel 2009. Esso propone di collegare diversi set di dati tra loro nel Semantic Web così che possano essere gestiti come un grande knowledge graph globale. Fino al 2014, circa 1.000 dataset sono collegati tra loro nel cloud Linked Open Data. Nel 2012 Google ha proposto una nuova tecnologia, chiamata Knowledge Graph, per utilizzare la conoscenza semantica nella ricerca sul web. Il grafo di conoscenza di Google viene utilizzato per identificare e disambiguare le entità nel testo, per arricchire i risultati della ricerca con riepiloghi strutturati semanticamente e per fornire collegamenti a entità correlate nella ricerca esplorativa, allo scopo di migliorare la capacità del motore di ricerca e migliorare l'esperienza di ricerca di utenti. Successivamente, molte altre aziende hanno iniziato a sviluppare i propri grafici della conoscenza. Ad esempio, Bing, un motore di ricerca sviluppato da Microsoft, ha integrato con Satori, un knowledge graph simile. Al giorno d'oggi, "Knowledge Graph" si riferisce anche a basi di conoscenza del web semantico come DBpedia, YAGO, Wikidata o Freebase.

Con l'ascesa a la diffusione dei KG, un numero sempre maggiore di motori di ricerca commerciali basati sul Web oggi sta incorporando i dati delle entità dei KG per migliorare i propri risultati di ricerca. Ad esempio, Google incorpora i dati di Google Plus e Google Knowledge Graph, mentre Facebook esegue le attività di ricerca sulle entità con Graph Search. La proprietà di KG di contenere la conoscenza umana sulle entità di parole reali aiuta i sistemi di ricerca a migliorare la loro capacità di comprendere query e documenti. Alcuni ricercatori si stanno concentrando sull'esplorazione del potenziale dei KG per il recupero delle informazioni. La ricerca orientata all'entità si sviluppa con lo sviluppo di KG su larga scala. Ci sono molti modi possibili per utilizzare la semantica di KG in diversi componenti come la rappresentazione della query, la rappresentazione di documento ed i sistemi di ranking dei sistemi di ricerca. La rappresentazione della query può essere migliorata introducendo entità correlate e relativi testi per espandere la query. La rappresentazione del documento può essere arricchita aggiungendo le entità annotate nel modello di spazio vettoriale del documento. Un altro modo è creare le connessioni aggiuntive dalla query ai documenti tramite entità correlate per migliorare il modello di classificazione. La ricerca orientata all'entità che incorpora la conoscenza umana dei KG sta mostrando risultati promettenti nel campo dell'Information Retrieval e le tecniche di deep learning rendono possibile apprendere modelli di classificazione più complessi a partire da training set arricchiti di grandi dimensioni.

Essendo la tecnologia del KG relativamente nuova, la diffusione di piattaforme che la impieghino è ancora in parte limitata. Benché esistono come detto alcuni casi di successo interessanti per la nostra trattazione, quali:

- *Google knowledge graph*. Il Knowledge Graph è una funzione di ricerca che è stata introdotta da Google il 16 maggio 2012 sul motore "google.com". Il Knowledge Graph è il

primo passo verso una ricerca semantica: grazie a questa funzione, il motore di ricerca di Google associa alle parole cercate un oggetto e metterà in relazioni oggetti in modo da avere una ricerca più veloce e accurata. Knowledge Graph condivide la stessa tecnologia che è alla base di Google Now, un'applicazione per Android sviluppata da Google Inc.. Google definisce il Knowledge Graph come “una gigantesca enciclopedia virtuale di fatti” in quanto contiene una raccolta di informazioni da più fonti e comprende più di 500 miliardi di fatti e circa 5 miliardi di entità e relazioni tra loro. Google definisce il Knowledge Graph come “una gigantesca enciclopedia virtuale di fatti” in quanto contiene una raccolta di informazioni da più fonti e comprende più di 500 miliardi di fatti e circa 5 miliardi di entità e relazioni tra loro.

- *Amazon Product Graph*. Il Product Graph rappresenta l’implementazione del KG di Amazon. Il product graph di Amazon è in grado di descrivere ogni articolo utilizzando concetti relativi al prodotto e non solo e consente di creare collegamenti tra diverse entità. Inoltre, il grafo aiuta i clienti a utilizzare una maggiore variazione nei termini di ricerca durante la ricerca di articoli. Il Product Graph utilizza una varietà di tecniche di machine learning per ottenere informazioni relative ai prodotti dalle pagine dei dettagli di Amazon e da Internet in generale.
- *Facebook Graph Search*. Si basa su Open Graph protocol e consente di rendere ogni pagina un oggetto ricco in un social graph. Open Graph è una tecnologia introdotta da Facebook nel 2010 che consente l'integrazione tra Facebook, i dati dell'utente e il sito web. Facebook fa lo scraping dell'URL che l’utente sta condividendo e ottiene le informazioni visualizzate dai meta tag all'interno della sezione <head> della pagina. Utilizzando tale tecnologia, Facebook riesce a costruire un grafo di relazioni per suggerire frammenti di informazioni sul web e integrarli sulla propria piattaforma social.

6. Conclusioni

Il presente documento ha avuto come oggetto lo studio e l'analisi critica dei principali metodi, strumenti e metodologie impiegate nel campo della fruizione "*intelligente*" di contenuti digitali. In particolare sono stati presi in esame algoritmi ed architetture utilizzate nella soluzione di tre task specifici e molto frequenti in questo ambito ossia: (*i*) La raccomandazione di contenuti che incontrino i gusti degli utenti e quindi implicitamente la loro profilazione. In particolare, sono stati analizzati e messi a confronto approcci tradizionali e metodi basati sull'impiego di architetture neurali evidenziandone i principali pregi, (*ii*) Utilizzo di strumenti di question/answering e di chatbot capaci di assistere l'utente nella ricerca di contenuti di suo interesse. Si è approfondito come metodi e strumenti di Text Mining e Natural Language Processing siano fondamentali nello sviluppo di questi sistemi e si sono analizzate le principali soluzioni basate sull'impiego di modelli di deep learning (ad es., BERT) ed (*iii*) impiego di strumenti di information retrieval avanzati capaci di suggerire all'utente concetti/entità e relazioni "inaspettati" partendo da item di suo interesse, focalizzandosi specificatamente sull'impiego della tecnologia emergente dei Knowledge Graph come valido ed efficace strumento per risolvere questo task.

In ultimo è stata presentata una rassegna delle più note piattaforme e tecnologie attualmente utilizzate nell'ambito dei task introdotti sopra.

7. BIBLIOGRAFIA E SITOGRAFIA

7.1. Bibliografia

1. [Apt94] C. Apté, F. Damerau, and S. Weiss. Automated learning of decision rules for text categorization. ACM Transactions on Information Systems , 12(3):233–251, 1994
2. [Ble03] Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
3. [Dee90] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic indexing. Journal of the American Society for Information Science, 41(6):391–407, 1990.
4. [Gri00] L. Grippo (2000): Convergent on-line algorithms for supervised learning in neural networks. IEEE Transactions on Neural Networks, Vol.11, n.6, 2000, pp.1284-1299.
5. [Itt95] D. J. Ittner, D. D. Lewis, and D. D. Ahn. Text categorization of low quality images. In Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval , pages 301–315, Las Vegas, US, 1995.
6. [Li98] Y. H. Li and A. K. Jain. Classification of text documents. The Computer Journal , 41(8):537–546, 1998.
7. [Sal75] G. Salton, A. Wong, and C. Yang. A vector space model for automatic indexing. Communications of the ACM , 18(11):613–620, 1975. Also reprinted in [45], pp. 273–280.
8. [Seb02] F. Sebastiani, Machine learning in automated text categorization, ACM Comput. Surv. 34 (2002) 1--47.
9. [Sch98] H. Schütze. Automatic word sense discrimination. Computational Linguistics, 24(1):97–124, 1998.
10. [Wie95] E. Wiener, J. O. Pedersen, and A. S. Weigend. A neural network approache to topic spotting.In Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval , pages 317–332, Las Vegas, US, 1995.
11. [Yan97] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In Proceedings of ICML-97, 14th International Conference on Machine Learning, pages 412–420, Nashville, US, 1997
12. [Bar13] Albert-László Barabási (2013). "Discussion: Network science". Philosophical Transactions of the Royal Society 371 (1987)
13. [Bal09] Balog R, Jørgensen B, Wells J, Lægsgaard E, Hofmann P, Besenbacher F & Hornekær L 2009 Journal of the American Chemical Society 131(25), 8744–8745.
14. [Cra05] Nick Craswell, Stephen E. Robertson, Hugo Zaragoza, Michael J. Taylor: Relevance weighting for query independent evidence. SIGIR 2005: 416-423
15. [Deng14] Li Deng and Dong Yu (2014), "Deep Learning: Methods and Applications", Foundations and Trends® in Signal Processing: Vol. 7: No. 3–4, pp 197-387.
16. [Bas17] Basiliyos Tilahun Betru, Charles Awono Onana, and Bernabe Batchakui. 2017. Deep

Learning Methods on Recommender System:
17. [Bet17]. Basiliyos Tilahun Betru, Charles Awono Onana, Bernabe Batchaku. A Survey of State-of-the-art. International Journal of Computer Applications 162, 10 (Mar 2017), 17–22. https://doi.org/10.5120/ijca2017913361
18. [Cao17] S. Cao, N. Yang, and Z. Liu. 2017. Online news recommender based on stacked auto-encoder. In 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS). 721–726. hŠps://doi.org/10.1109/ICIS.2017.7960088
19. [Che17] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. AŠentive Collaborative Filtering: Multimedia Recommendation with Item- and Component-Level AŠention. Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval (2017)
20. [Tze16] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. ACM, 7–10.
21. [Col08] Ronan Collobert and Jason Weston. 2008. A unied architecture for natural language processing: Deep neural networks with multitask learning. In Proceedings of the 25th international conference on Machine learning. ACM, 160–167.
22. [Dai16] Hanjun Dai, Yichen Wang, Rakshit Trivedi, and Le Song. 2016. Deep coevolutionary network: Embedding user and item features for recommendation. arXiv preprint. arXiv preprint arXiv:1609.03675 (2016).
23. [Den14] Li Deng, Dong Yu, et al. 2014. Deep learning: methods and applications. Foundations and Trends® in Signal Processing 7, 3–4 (2014), 97–387.
24. [Dev16] Robin Devooght and Hugues Bersini. 2016. Collaborative ltering with recurrent neural networks. arXiv preprint arXiv:1608.07400 (2016). ACM.
25. [Ber01] Berners-Lee, T., Hendler, J., Lassila, O., et al.: The semantic web. Scientific american 284(5), 28–37 (2001)
26. [Biz11] Bizer, C., Heath, T., Berners-Lee, T.: Linked data: The story so far. In: Semantic services, interoperability and web applications: emerging concepts, pp. 205–227. IGI Global (2011).
27. [Oar98] Oard DW, Kim J. Implicit feedback for recommender systems. In: Proceedings of 5th DELOS workshop on filtering and collaborative filtering; 1998. p. 31–6
28. [Bur02] R. Burke. Hybrid recommender systems: survey and experiments User Model User-adapted Interact, 12 (4) (2002), pp. 331-370
29. [Bob13] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez. Recommender systems survey Knowl-Based Syst, 46 (2013), pp. 109-132
30. Deng e Yu 2014 : Deng L, Yu D (2014) Deep learning: methods and applications. Found Trends Signal Process 7(3–4):197–387. https://doi.org/10.1561/2000000039
31. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Comput 18(7):1527–1554. https://doi.org/10.1162/neco.2006.18.7.1527
32. [Ben09] Bengio Y (2009) Learning deep architectures for ai. Found Trends® Mach Learn 2(1):1–127. https://doi.org/10.1561/2200000006
33. [Che17] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive Collaborative Filtering: Multimedia Recommendation with Item- and Component-Level Attention. Proceedings of the 40th International ACM SIGIR conference on

Research and Development in Information Retrieval (2017).
34. [Gon16]Yuyun Gong and Qi Zhang. 2016. Hashtag Recommendation Using Attention-Based Convolutional Neural Network.. In IJCAI. 2782–2788.
35. [Seo17] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Representation Learning of Users and Items for Review Rating Prediction Using Attention-based Convolutional Neural Network. In 3rd International Workshop on Machine Learning Methods for Recommender Systems (MLRec)(SDM'17).
36. [Ouy14] Ouyang et al. 2014: Yuanxin Ouyang,Wenqi Liu,Wenge Rong, and Zhang Xiong. 2014. Autoencoder-based collaborative filtering. In International Conference on Neural Information Processing. Springer, 284–291.
37. [Sed15] Sedhain et al. 2015: Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In Proceedings of the 24th International Conference on World Wide Web. ACM, 111–112.
38. Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item Silk Road: Recommending Items from Information Domains to Social Users. (2017).
39. [Den17] Deng et al. 2017: Shuiguang Deng, Longtao Huang, Guandong Xu, Xindong Wu, and Zhaohui Wu. 2017. On deep learning for trust-aware recommendations in social networks. IEEE transactions on neural networks and learning systems 28, 5 (2017), 1164–1177.
40. [Zuo16] Zuo et al. 2016 Yi Zuo, Jiulin Zeng, Maoguo Gong, and Licheng Jiao. 2016. Tag-aware recommender systems based on deep neural networks. Neurocomputing 204 (2016), 51–60.
41. [Ung16] Unger et al. 2016: Moshe Unger, Ariel Bar, Bracha Shapira, and Lior Rokach. 2016. Towards latent context-aware recommendation systems. Knowledge-Based Systems 104 (2016), 165–178.
42. [Geo13] Georgiev e Nakov 2013: Kostadin Georgiev and Preslav Nakov. 2013. A non-iid framework for collaborative filtering with restricted boltzmann machines. In International Conference on Machine Learning. 1148–1156.
43. [Hue] Hu et al. 2014: Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM '08). IEEE Computer Society, Washington, DC, USA, 263–272. https://doi.org/10.1109/ICDM.2008.22
44. [Tru09] Truyen et al. (2009): Truyen TT, Phung DQ, Venkatesh S (2009) Ordinal Boltzmann machines for collaborative filtering. In: Proceedings of the 25th conference on uncertainty in artificial intelligence, pp 548–556
45. [Gun08] Gunawardana e Meek (2008): Gunawardana A, Meek C (2008) Tied Boltzmann machines for cold start recommendations. In: Proceedings of the 2nd ACM conference on recommender systems, Lausanne, Switzerland, pp 19–26
46. [Wan14] Wang e Wang 2014: Wang X, Wang Y (2014) Improving content-based and hybrid music recommendation using deep learning. In: Proceedings of the 22nd ACM international conference on multimedia, Orlando, Florida, USA, pp 627–636
47. [Wu16] Caihua Wu, Junwei Wang, Juntao Liu, and Wenyu Liu. 2016. Recurrent neural network based recommendation for time heterogeneous feedback. Knowledge-Based Systems 109 (2016), 90–103.
48. [Str15] Strub and Mary 2016. Hybrid Recommender System based on Autoencoders. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. ACM, 11–16.

49. [Wan15a] Hao Wang, Xingjian Shi, and Dit-Yan Yeung. 2015. Relational Stacked Denoising Autoencoder for Tag Recommendation. In AAAI. 3052–3058.
50. [Str16] Strub et al. 2016. Hybrid Recommender System based on Autoencoders. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. ACM, 11–16.
51. [Sed15] Suvash Sedhain, Aditya Krishna Menon, Sco Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In Proceedings of the 24th International Conference on World Wide Web. ACM, 111–112.
52. [Li15] Sheng Li, Jaya Kawale, and Yun Fu. 2015. Deep collaborative ltering via marginalized denoising auto-encoder. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. ACM, 811–820.
53. [Kyo14] Kyo-Joong O, Won-Jo L, Chae-Gyun L, Choi HJ (2014) Personalized news recommendation using classified keywords to capture user preference. In: Proceedings of the 16th international conference on advanced communication technology, Phoenix Park, PyeongChang Korea(south), pp 1283–1287
54. [Zha15] Zhao Y, Wang J, Wang F (2015) Word embedding based retrieval model for similar cases recommendation. In: Proceedings of 2015 Chinese automation congress, Wuhan, China, pp 2268–2272
55. [Hu14] Hu et al. 2014: Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM '08). IEEE Computer Society, Washington, DC, USA, 263–272. https://doi.org/10.1109/ICDM.2008.22
56. [Wu16a] Wu S, Ren W, Yu C, Chen G, Zhang D, Zhu J (2016a) Personal recommendation using deep recurrent neural networks in netease. In: Proceedings of the IEEE 32nd international conference on data engineering, Helsinki, Finland, pp 1218–1229
57. [Ko16] Ko YJ,Maystre L, GrossglauserM (2016) Collaborative recurrent neural networks for dynamic recommender systems. In: Proceedings of the 8th Asian conference on machine learning, Hamilton, New Zeland, vol 63, pp 366–381
58. [Dev17] Devooght R, Bersini H (2017) Long and short-term recommendations with recurrent neural networks. In: Proceedings of the 25th conference on user modeling, adaptation and personalization, Bratislava, Slovakia, pp 13–21
59. [Oor13] Oord Avd, Dieleman S, Schrauwen B (2013) Deep content-based music recommendation. In: Proceedings of the 26th international conference on neural information processing systems, Lake Tahoe, NV, USA, pp 2643–2651
60. [She16] Shen X,Yi B, Zhang Z, Shu J, LiuH(2016) Automatic recommendation technology for learning resources with convolutional neural network. In: Proceedings of the international symposium on educational technology, Beijing, China, pp 30–34
61. [Zho16] Zhou J, Albatal R, Gurrin C (2016) Applying visual user interest profiles for recommendation and personalisation. In: Proceedings of the 22nd international conference on multimedia modeling, Miami, FL, USA, pp 361–366
62. [Lei16] Lei C, Liu D, Li W, Zha ZJ, Li H (2016) Comparative deep learning of hybrid representations for image recommendations. In: Proceedings of the 29th IEEE conference on computer vision and pattern recognition, Las Vegas, NV, USA, pp 2545–2553
63. [Wu17b] Wu H, Zhang Z, Yue K, Zhang B, Zhu R (2017b) Content embedding regularized matrix factorization for recommender systems. In: Proceedings of the 2017 IEEE international

congress on big data, Boston, MA, USA, pp 209–215