NLP - LEZIONE 20

DEL 16/12/2019

RAPPR. DISTRIBUITA II

Continuiamo la discussione sulle rappresentazioni su SPAZI METRICI...

RAPPR. DISCRETA VS RAPPR. DISTRIBUITA

Come già detto con una rappr. DISCRETA possiamo SEMPRE recuperare i simboli iniziali. La rappr. DISTRIBUITA invece si avvicina al modo in cui funziona il nostro CERVELLO per rappresentare la CONOSCENZA. Ci dobbiamo quindi chiedere: le due rappr. sono la stessa cosa oppure no?

Se vogliamo capire quanto vicina è una rappr. DISTRIBUITA o DISTRIBUZIONALE ad una rappr. DISCRETA, dobbiamo analizzare i seguenti livelli:

1) SYMBOLIC LEVEL

   ↳ SIMBOLI SONO RICONOSCIBILI?

2) COMPOSITIONAL LEVEL

   Il tipo di COMPOSIZIONALITÀ utilizzata è tale da mantenere INALTERATI i simboli all'interno della composizione?

In altre parole, è possibile TIRARE FUORI un dato simbolo dopo averlo convertito?

SYMBOLIC LEVEL ~ (11:40 min)

Ricordiamo il ruolo di TRANSLATION dalle due tipologie di rappr:

$$\underset{\text{(SIMBOLO)}}{w} \longrightarrow \underset{\text{(ONE-HOT VECTOR)}}{\vec{w}_e} \longrightarrow \underset{\text{(VETTORE)}}{\vec{w} = W \cdot \vec{w}_e}$$

Dato il vettore $\vec{w}$, per capire il SIMBOLO DISCRETO codificato in $\vec{w}$, NECESSITIAMO che la matrice W sia INVERTIBILE. Infatti, se esiste W^{-1}, allora

$$\begin{aligned} W^{-1} \cdot \vec{w} &= W^{-1} \cdot (W \vec{w}_e) \\ &= (W^{-1} \cdot W) \cdot \vec{w}_e \\ &= I \cdot \vec{w}_e \\ &= \vec{w}_e \end{aligned}$$

e da $\vec{w}_e$ riusciamo immediatamente ottenere il simbolo discreto w.

Notiamo poi che, tipicamente, W è una matrice $d \times n$ con $d \ll n$. Questo ci permette di RIDURRE la DIMENSIONE della matrice. Un modo per ottenere la matrice W è offerto dal modello della SEMANTICA DISTRIBUZIONALE trattato nella lezione precedente.

Dati due vettori ONE-HOT $\vec{a}_e, \vec{b}_e$ abbiamo che

$$\vec{a}_e \cdot \vec{b}_e = \begin{cases} 1 & , \ \vec{a}_e = \vec{b}_e \\ 0 & , \ \text{altrimenti} \end{cases}$$

$\cdot$ ↑ PRODOTTO SCALARE

(!) CAPIRE 22:00

Se invece ho due vettori $\vec{a}$ e $\vec{b}$, troviamo che

$$\vec{a} \cdot \vec{b} = (W\vec{a}_e) \cdot (W\vec{b}_e)$$

$$= (\vec{a}_e W^T) \cdot (W \vec{b}_e)$$

$$= \vec{a}_e (W^T \cdot W) \cdot \vec{b}_e$$

Per mantenere la DISTINGUIBILITÀ dei simboli, ovvero per fare in che $\vec{a} \cdot \vec{b} = \vec{a}_e \cdot \vec{b}_e$, NECESSITIAMO che $W^T = W^{-1}$. Se non rinuncio a fare questo allora le rappr. DISTRIBUITE non hanno avere una composizionalità CONCATENATIVA. Per avere una buona APPROSSIMAZIONE del fatto che $W^T = W^{-1}$ vogliamo

$$W^T \cdot W = I + \Sigma$$

con Σ "piccolo". Abbiamo questa situazione quando W ha la seguente forma: i) valori SIMILI a 1 sulla DIAGONALE e ii) valori SIMILI a 0 FUORI dalla DIAGONALE.

OSS: Un altro modo di costruire la matrice W è tramite il WORD EMBEDDING nel contesto delle RETI NEURALI.

Consideriamo adesso il modello della SEMANTICA DISTRIBUZIONALE, per vedere se la matrice W costruita utilizzando i CONTESTI rispetta la CONDIZIONE trovata prima.

Sia $W = \begin{pmatrix} | & & | \\ w_1 & \cdots & w_n \\ | & & | \end{pmatrix}$, e consideriamo il prodotto $W^T \cdot W$. Abbiamo che,

$$W^T \cdot W = (a_{i,j})_{i,j} = (w_i^T \cdot w_j)_{i,j}$$

con

$$a_{i,j} = \begin{cases} 1, & i = j \\ > 0, & i \neq j \end{cases}$$

Notiamo che $a_{i,j} > 0$ quando $i \neq j$ in quanto la SIMILARITÀ tra due parole in questo modello dipende da quanto le parole appaiono negli stessi contesti.

Da questo segue che la SEMANTICA DISTRIBUZIONALE ci allontana dalla DISTINGUIBILITÀ dei SIMBOLI e quindi da una rappr. DISCRETA.

GESTIRE SEQUENZE DI SIMBOLI ~ (40:00 min)

La SEMANTICA DISTRIBUZIONALE, che poi è diventata il WORD EMBEDDING, ci permette di rappresentare la CONOSCENZA SEMANTICA di un # molto LIMITATO di parole.

Siamo adesso interessati a come gestire la TRANSIZIONE da SINGOLI simboli a SEQUENZE di simboli. In questo contesto siamo soltanto interessati a come MANTENERE le RELAZIONI tra le varie parole: parole simili devono rimanere tali e anche le parole diverse devono rimanere tali.

Consideriamo le seguenti due sequenze di parole:

SUSSUNZIONE
- MELA — ROSSA
- FRUTTO — COLORATO

Per UNIRE le due parole di ogni sequenza in una rappr. DISTRIBUITA procediamo come segue:

$$\vec{x} := \overrightarrow{\text{MELA}} + \overrightarrow{\text{ROSSA}} = \vec{m} + \vec{r}$$
$$\vec{y} := \overrightarrow{\text{FRUTTO}} + \overrightarrow{\text{COLORATO}} = \vec{f} + \vec{c}$$

Per vedere se le due parole sono SIMILI procediamo calcolando il PRODOTTO SCALARE:

$$\vec{x}^T \cdot \vec{y} = (\vec{m} + \vec{n})^T \cdot (\vec{p} + \vec{c})$$

$$= \vec{m}^T \cdot \vec{p} + \vec{m}^T \cdot \vec{c} + \vec{n}^T \cdot \vec{p} + \vec{n}^T \cdot \vec{c}$$

Notiamo però che così facendo stiamo comparando gli elementi a COPPIE, e questo non è IDEALE, in quanto comparare MELA con COLORATO è poco utile come cosa.

Anche introducendo delle MATRICI che idealmente codificano in modo diverso parti del discorso diverse, non riusciamo a risolvere la problematica di prima

$$\vec{x} := A \cdot \overrightarrow{\text{MELA}} + B \cdot \overrightarrow{\text{ROSSA}}$$

$$\vec{y} := A \cdot \overrightarrow{\text{FRUTTA}} + B \cdot \overrightarrow{\text{COLORATA}}$$

MATRICE CHE CODIFICA I NOMI

MATRICE CHE CODIFICA GLI AGGETTIVI.

$\vec{x} \cdot \vec{y}$ = Formula complicata con

$A^T \cdot A$, $A^T \cdot B$, $B^T \cdot A$, $B^T \cdot B$

Se volessimo confrontare SOLO le parole nella stessa posizione (MELA vs FRUTTA & ROSSA vs COLORATA) allora NECESSITIAMO che

i) $A^T \cdot A \approx I$

ii) $A^T \cdot B \approx 0$

Q1: Esistono delle TRASFORMAZIONI LINEARI W t.c. $W^T = W^{-1}$?

R: Tipicamente no, in quanto le matrici con cui lavoriamo non sono MATRICI QUADRATE $n \times n$ ma sono matrici $d \times n$, con $d << n$ per RIDURRE la dim. dello SPAZIO.

Esistono però l'algoritmo di PSEUDO-INVERSIONE che trova la matrice W della PSEUDO-INVERTIBILE, che è la GENERALIZZAZIONE del concetto di MATRICE INVERSA per matrici NON QUADRATE.

Q2: Come facciamo a ottenere matrici W $d \times n$ t.c.

i) $d << n$

ii) W^T è la PSEUDO-INVERSA di W

R: Vedere SEZIONE successive.

GAUSSIANA MULTIVARIATA ~ (1:03:00 min)

Le GAUSSIANA MULTIVARIATA è formata da VETTORI i cui componenti sono GAUSSIANE STANDARD.

Formalmente, dati $x_1, \dots, x_n$ i.i.d. con $x_i \sim N(\mu_i, \sigma_i^2)$ abbiamo che $\vec{x} = (x_1, x_2, \dots, x_n) \sim N_n(\vec{\mu}, \Sigma)$, con

i) $\vec{\mu}$ vettore delle medie

$$\vec{\mu} = (\mu_1, \dots, \mu_n)$$

ii) Σ matrice di covarianza

$$\Sigma = \begin{pmatrix} \sigma_{11}^2 & \cdots & \sigma_{1n}^2 \\ \vdots & \ddots & \\ \sigma_{n1}^2 & \cdots & \sigma_{nn}^2 \end{pmatrix} = \left(\sigma_{i,j}^2\right)_{i,j}$$

con

$$\sigma_{i,j}^2 = \mathrm{Cov}[x_i, x_j]$$

Notiamo che, nelle RETI NEURALI, l'inizializzazione viene effettuata tramite dei vettori GENERATI da una GAUSSIANA MULTI-VARIATA. Questa scelta deriva dal fatto che se abbiamo una GAUSSIANA MULTI-VARIATA con $\vec{\mu} = \vec{0}$ e $\Sigma = \frac{1}{n} \cdot I$, allora i vettori generati hanno NORMA ≈ 1 e PRODOTTO SCALARE ≈ 0.

LEMMA DI JOHNSON-LINDENSTRAUSS

Risultato che dice che è possibile passare da uno SPAZIO ad un altro SPAZIO riuscendo comunque a MANTENERE le DISTANZE tra due punti.

In particolare il risultato dice che ESISTE un ALGORITMO che, in un TEMPO FINITO, trova i VETTORI che permettono di ottenere tale TRASFORMAZIONE.

Formalmente si trova una TRASFORMAZIONE W t.c.

$$-\varepsilon|x-y| < |W\cdot x - W\cdot y| < \varepsilon|x-y|$$

con x e y vettori dello spazio di PARTENZA

ESERCIZIO: ~ (1:15:00 min)

Utilizzare PYTHON per generare dei vettori da una NORMALE (GAUSSIANA) MULTIVARIATA $\bar{x} \sim N_m\left(0, \frac{1}{\sqrt{d}} \cdot I\right)$ e verificare che prendendo coppie di vettori $\bar{u}$ e $\bar{v}$ generati in questo modo si ha che:

1) $-\varepsilon < \bar{u}^T \cdot v < \varepsilon$

2) $1-\varepsilon < u^T \cdot u < 1+\varepsilon$

con $\varepsilon = p(d)$ e $d :=$ DIMENSIONE dello spazio di ARRIVO.

Ricapitolando, tramite l'utilizzo di questi vettori siamo in grado di RICONOSCERE, più o meno, il simbolo associato ad un vettore utilizzando il PRODOTTO SCALARE.

Adesso però dobbiamo capire come possiamo mettere le parole ASSIEME cercando nel frattempo di mantenere l'ORDINE e l'IDENTIFICABILITÀ dei SIMBOLI.

OSS: L'utilizzo delle rappresentazioni DISTRIBUITE è NECESSARIO se vogliamo utilizzare degli algoritmi di APPRENDIMENTO che lavorano su SPAZI METRICI.

Prima del successo delle RETI NEURALI si utilizzavano le SVM (SUPPORT VECTOR MACHINES). In ogni caso, da un po' di anni si sente parlare solo di reti neurali.

Anche se le RETI NEURALI funzionano bene, il loro COMPORTAMENTO e il modo in cui prendono le decisioni rimangono cose DIFFICILI da descrivere. Per fare questo bisogna quindi studiare le PROPRIETÀ dei due tipi di rappr., per vedere se sono fondamentalmente DIVERSE oppure no. I RICERCATORI moderni però non sembrano essere interessati a queste tematiche.

CONVOLUZIONE CIRCOLARE ~ (1:40:00 min)

Nel 1996 PLATE ha lavorato alle HOLOGRAPHIC REDUCED REPRESENTATIONS (HRR), che utilizzano i VETTORE discussi prima connessi con la CONVOLUZIONE CIRCOLARE.

La CONVOLUZIONE CIRCOLARE è una tecnica che viene utilizzata per RECUPERARE i segnali dopo averli combinati.

Per capire meglio l'effetto di questa operazione introduciamo il concetto di MATRICE CIRCOLANTE:

Dato un vettore $\vec{a} = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_d \end{pmatrix}$,

la MATRICE CIRCOLANTE di $\vec{a}$ è definita come segue:

$$A_c := \begin{pmatrix} a_0 & a_d & a_{d-1} & \cdots & a_1 \\ a_1 & a_0 & a_d & & a_2 \\ \vdots & a_1 & a_0 & \ddots & \vdots \\ & \vdots & a_1 & & a_d \\ a_d & a_{d-1} & a_{d-2} & & a_0 \end{pmatrix}$$

La CONVULAZIONE CIRCOLARE è quindi definita come segue:

$$a \circledast b := \underbrace{A_c}_{\text{MATRICE CIRCOLANTE di } a} \cdot b$$

$\circledast$ ↑ CONVOLUZIONE CIRCOLARE

Notiamo che, se lavoriamo con due vettori a, b generati come discusso prima, utilizzando la CONVOLUZIONE CIRCOLARE, non recuperiamo i SIMBOLI che non siano stati composti tra loro. Questo segue dal fatto che $A_c^T \cdot A_c \approx I$, e quindi $A_c^T \cdot A_c \cdot b \approx b$.

Dato che comunque $\otimes$ è una operazione COMMUTATIVA

$$a \otimes b = A_c \cdot b = B_c \cdot a = b \otimes a$$

non possiamo ottenere le SEQUENZE effettive dei simboli utilizzati.

SHUFFLED CIRCULAR CONVOLUTION ~ (2:03:00 min)

Vogliamo cercare di non PERDERE l'ordine dei SIMBOLI una volta COMPOSTI.

Per fare questo inizialmente si è cercato di implementare delle MEMORIE utilizzando delle MATRICI. (DISTRIBUTED MEMORIES)

Un altro approccio utilizzato è stato quello del RANDOM INDEXING, inizialmente ideato per COMPATTARE la mole di rappr. dei CONTESTI delle parole.
Successivamente, SAHLGREN ET. AL. hanno pensato di utilizzare una MATRICE DI SHUFFLING per rappresentare l'ordine.

Una MATRICE DI SHUFFLING è una matrice il cui effetto è quello di mostare gli elementi del vettore con cui viene moltiplicata.

ESEMPIO:

Date $\underset{3\times3}{\Phi} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ e $\underset{3\times1}{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix}$,

abbiamo che

$$\Phi \cdot a = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

$$= (a_2 \ a_1 \ a_3)$$

Notiamo inoltre che $\Phi^T \cdot \Phi = I$.

La SHUFFLE MATRIX Φ è utilizzata in una nuova operazione chiamata SHUFFLED CIRCULAR CONVOLUTION e definita come segue

$$a \circledast b := (\Phi a) \otimes b = a \otimes (\Phi b)$$

($\otimes$ è COMMUTATIVA)

Per codificare BENE la nozione devo avere che

$$\nexists n : \Phi^n = I$$

Notiamo che l'introduzione della SHUFFLE MATRIX Φ ha il seguente EFFETTO:

$$a \circledast (b \circledast c) = A_c \; \Phi (B_c \; \Phi \; c)$$

$$(a \circledast b) \circledast c = (A_c \; \Phi \; b)_c \; \Phi \; c$$

Φ li rende DIVERSI

Portando a DIVERSI risultati l'operazione non è COMMUTATIVA, ed è quindi possibile ottenere l'ORDINE corretto dei simboli combinati.

OSS: È possibile definire sia l'elemento NULLO che quello NEUTRO per l'operazione $\circledast$.