

STATISTICAL MACHINE LEARNING

Uno dei modi per comprendere testo non strutturato è usare il machine learning

1) Definire un **training set** (d_i, y_i)

2) Scegliere delle feature che formano una **representation** (Φ) di un documento: $d: \Phi(d) \in \mathbb{R}^m$

3) Scegliere una **hypothesis function** F , detta anche **modello**, che dipende da qualche **parametro**, che è in grado di stimare $\hat{y}: \hat{y} = F(\Phi(d) | \Theta)$ **parametri**

4) Guardare la **cost function** J per valutare il modello rispetto al training set, usando qualche tipo di **error function** e : $J(\Theta) = \sum_i e(\hat{y}_i - y_i)$

5) Trovare nuovi Θ che minimizzano $J(\Theta)$ e portano ad **ottimizzare il modello**

MODELLARE UN LINGUAGGIO

Un **language model (LM)** è una funzione che assegna delle probabilità a una sequenza di parole. Difatti è una distribuzione di probabilità che assegna una probabilità a ogni parola, anche a quelle che non hanno senso insieme o erette.

Vocabolario

Definiamo il **vocabolario** V come un set finito di simboli. Ogni simbolo $w \in V$ è detto **word**. Per semplicità definiamo $V = |V|$ ovvero la sua cardinalità.

Frase

È una lista di parole prese da V : $w_1, w_2, \dots, w_n \in V$.

V^+

Definiamo V^+ come il set di tutte le frasi possibili in V , di tutte le lunghezze finite possibili.

Definizione matematica di un LM

Un language model basato sul vocabolario V è una funzione del tipo: $P: V^+ \rightarrow \mathbb{R}$ tale per cui:

- 1) \forall frase $s \in V^+$: $P(s) > 0$
- 2) $\sum_{s \in V^+} P(s) = 1$

Corpus

Approssimeremo V^+ con \hat{V}^+ , detto corpus, e ci concentreremo nel calcolare LM approssimati con \hat{V}^+ .

In qualche caso il vocabolario \hat{V} ricavato dal corpus \hat{V}^+ non coincide con V . Se $\hat{V} \equiv V$ allora contenerebbe tutte le parole possibili, e ovviamente ciò che è definito come **closed vocabulary** \hat{V} . Quando non coincide abbiamo un **open vocabulary** \hat{V} con parole sconosciute anche dette **Out-of Vocabulary** (OOV words).

Terminologia

Da ora in avanti chiameremo:

- 1) $P(s)$ la distribuzione di probabilità
- 2) $P[\bar{s}]$ il valore della distribuzione $P(s=\bar{s})$
- 3) ugualmente per $P[w_1 \dots w_n]$ e $P["\text{the black cat}"]$
- 4) $\hat{V}^+ \approx V^+$ quindi useremo solo V^+
- 5) Useremo $P(x|\Theta) \circ P_{\Theta}(x)$ invece di $\hat{P}(x)$

EVALUATING LM

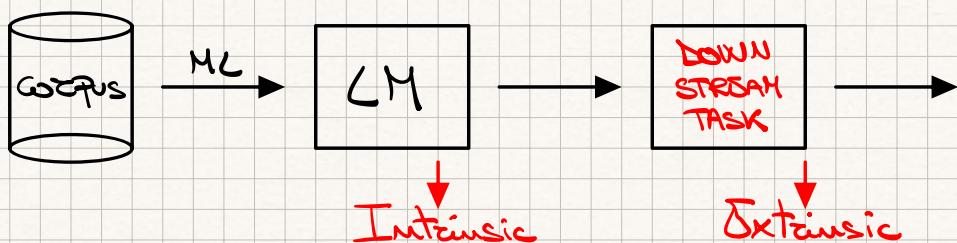
Ci serve un modo formale per valutare un LM approssimato. Ci sono due modi diversi

1) Extrinsic Evaluation

Usiamo il LM e valutiamo il suo output. Diversi task richiedono diverse valutazioni

2) Intrinsic Evaluation

Valutazione indipendente dall'applicazione del LM



PERPLEXITY (PP)

La metrica più importante per le valutazioni intrinseche è la perplexità PP.

La perplexity PP di un language model LM, calcolata sul un **corpus di valutazione** $w_1 \dots w_T$ (set diverso dal corpus di training) è definita come segue:

$$PP(w_1 \dots w_T) = P[w_1 \dots w_T]^{-\frac{1}{T}} = \sqrt[T]{\frac{1}{P[w_1 \dots w_T]}}$$

Chain Rule

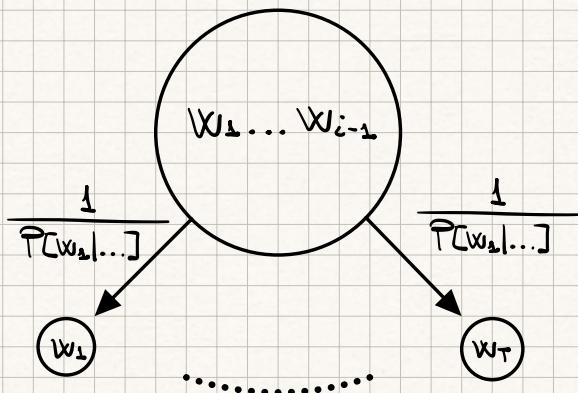
$$\begin{aligned} P[w_1 \dots w_T] &= P[w_1] \cdot P[w_2 | w_1] \cdot P[w_3 | w_1, w_2] \cdot \dots \cdot P[w_T | w_1 \dots w_{T-1}] \\ &= P[w_1] \prod_{i=2}^T P[w_i | w_1 \dots w_{i-1}] \end{aligned}$$

Usando la chain rule sulla formula della perplexity:

$$PP(w_1 \dots w_T) = \sqrt[T]{\frac{1}{P[w_1] \prod_{i=2}^T P[w_i | w_1 \dots w_{i-1}]}}$$

Che è **la media geometrica** dell'inverso di tutte le probabilità per ogni termine nel corpus di valutazione.
Vogliamo minimizzare PP (ovvero maximizzare le probabilità)

La perplexity di un LM ci dice quanto è bravo a predire la parola che segue, dato una sequenza di parole in ingresso:



Note: mediamente in inglese, la prossima parola è scelta fra 20/50 parole del vocabolario, non fra tutte.

Esempio

Calcoliamo PP di un "dump" LM, ovvero un modello che assegna la stessa probabilità a ogni parola predetta, indipendentemente da ciò che le precede:

$$\forall i=1 \dots V \quad P[w_i] = 1/V$$

Calcoliamo la perplexity, ma prima calcoliamo:

$$P[w_1 \dots w_T] = P[w_1] \prod_{i=2}^T P[w_i | w_1 \dots w_{i-1}] = \frac{1}{V} \dots \frac{1}{V} = \frac{1}{V^T}$$

Da qui PP:

$$PP(w_1 \dots w_T) = \sqrt[T]{\frac{1}{V^T}} = \frac{1}{V^{1/T}}$$

Che è quindi il valore massimo possibile, entropia massima del momento in cui non esiste una parola migliore.

START AND END TOKENS

Introduciamo due token speciali, aggiunti a V:

1) Start Token $\langle s \rangle$

2) End Token $\langle /s \rangle$

Questi token sono sempre usati in ogni frase per delinare l'inizio e la fine di essa. Ci sono sempre.

Dobbiamo fare così perché così siamo in grado di definire correttamente la probabilità di **tutta la frase** e non solo di una specifica frase di lunghezza T.

Bisogna tenere in conto anche il numero di parole dentro una frase non solo le parole stesse.

Дата una frase $w_1 \dots w_m$ consideriamo ora la nuova frase $w_0 w_1 \dots w_m w_{m+1}$ con $w_0 = \langle s \rangle$ e $w_{m+1} = \langle /s \rangle$.

Calcoliamo la nuova probabilità

$$P[w_0 w_1 \dots w_m w_{m+1}] = P[w_0] \cdot P[w_1 | w_0] \dots P[w_{m+1} | w_0 \dots w_m]$$

dove sappiamo che $P[w_0] = 1$, $P[w_1 | w_0] = P[w_1]$ quindi non è proprio zardon:

$$P[\underline{w_0 \dots w_{m+1}}] = \prod_{i=1}^{m+1} P[w_i | w_1 \dots w_{i-1}]$$

Abbiamo bisogno di ricalcolare correttamente la perplexity:

Abbiamo un evaluation corpus con t frasi:

$\langle s \rangle w_{11} w_{12} \dots w_{1T_1} \langle /s \rangle$

$\langle s \rangle w_{21} w_{22} \dots w_{2T_2} \langle /s \rangle$

\vdots

$\langle s \rangle w_{t1} w_{t2} \dots w_{tT_t} \langle /s \rangle$

Ora calcoliamo $P[w_{11} \dots w_{tT_t}]$

$$P[w_{11} \dots w_{tT_t}] = \prod_{i=1}^t P[w_{i1} \dots w_{iT_i}] =$$

$$= \prod_{i=1}^t \prod_{j=1}^{T_i+1} P[w_{ij} | w_{i1} \dots w_{i(j-1)}]$$

Stiamo moltiplicando insieme $\sum_{i=1}^t (T_i + 1)$ diverse probabilità.

Questo numero è:

$$\sum_{i=1}^t (T_i + 1) = \sum_{i=1}^t (T_i + t) = T + t$$

GENDRARS FRASI

Possiamo usare un LM per generare una semplice nuova frase che inizia con $\langle S \rangle$, data una sequenza di parole di ingresso $w_1 \dots w_n$. Il modo più semplice per generare le prossime parole, scegliendo fra $\hat{w}_1 \dots \hat{w}_i$ è:

$$\hat{w}_1 = \operatorname{argmax}_{w \in V} (P(w | w_1 \dots w_n))$$

$$\hat{w}_2 = \operatorname{argmax}_{w \in V} (P(w | w_1 \dots w_n \hat{w}_1))$$

:

$$\hat{w}_i = \operatorname{argmax}_{w \in V} (P(w | w_1 \dots w_n \hat{w}_1 \dots \hat{w}_{i-1}))$$

Ripetiamo finché non viene generato $\langle /S \rangle$

Questo approccio è detto **GREEDY SEARCH**.