Report modello tesi

Matteo Posenato

April 2023

1 Introduction

Ho selezionato il modello "$MTL - ML_2$" perchè dalle tabelle sembra essere il migliore a livello complessivo di performance. Il modello è basato sul mutual learning e usa la somiglianza del coseno come misura della performance.

2 Funzionamento

Il metodo è composto da due parti principali, una preposta alla Topic Detection e una relativa alla Sentiment Classification. L'approccio proposto in questo paper è lo scambio di informazioni tra le due parti in modo tale da aumentare le informazioni nei due modelli ed avere una risposta migliore dell'output. Per la parte di Topic Detection viene utilizzato il dataset Yelp, che contiene le recensioni di diverse attività commerciali e non è supervisionato. Per quanto riguarda invece la parte di Sentiment Classification invece viene utilizzato il dataset IMBD che riguarda recensioni di film classificate (supervisionate) come positive o negative.

2.1 Topic Detection

Questa sezione è divisa in due reti: inference network and generative network.

2.1.1 Inference Network

Definiamo due MLP (multi layers perceptron): f_{μ_θ} e $f_{\sum_\theta}$ prendono come input la frequenza delle parole all'interno di ogni documento d,

$$\mu_\theta = f_{\mu_\theta}(w_d)$$

$$\sum_\theta = diag(f_{\sum_\theta}(w_d))$$

e restituiscono la media e la varianza di una distribuzione Gaussiana. Per ogni documento w_d la sua distribuzione viariazionale è: $q(\theta) \simeq \mathcal{N}(\mu_\theta, \sum_\theta)$ da cui è possibile generare un campione $\hat{\theta} = \sigma(\mu_\theta + \sum_\theta^{1/2} \epsilon)$

2.1.2 Generation Network

prende il campione generato $\hat{\theta}$ e lo uso in due MLP per generare z_d, si può evidenziare un limite inferiore di

$$L_t(w_d) \approx \frac{1}{L}\sum_{l=1}^{L}\sum_{n=1}^{N_d} logp(w_{d,n}|\hat{\theta}^{(l)}) - KL(q(z_d|w_d)||p(z_d))$$

2.2 Reconstruct Layer

Infine abbiamo una rete con un layer singolo che ricostruisce e cattura i pesi tra ogni parola e il topic latente.

2.3 Sentiment Classification

Per la parte di text classification usiamo una recurrent neural network gerarchica per modellare un documento. Assumiamo che in un documento w_d contiene M_d frasi $w_d = \{s_1, s_2, ..., s_{M_d}\}$ e il word embedding della j-esima parola nella i-esima frase è w_i^j quindi la rappresentazione della frase s_i può essere ottenuta dai seguenti step:

$$x_i^j = W \cdot w_i^j$$
$$\overrightarrow{h_i^j} = \overrightarrow{GRU}(x_i^j)$$
$$\overleftarrow{h_i^j} = \overleftarrow{GRU}(x_i^j)$$
$$h_i^j = \overrightarrow{h_i^j} \oplus \overleftarrow{h_i^j}$$
$$u_i^j = tanh(W_w \cdot h_i^j + b_w)$$
$$\alpha_i^j = \frac{exp(u^T \cdot u_i^j)}{\sum_T exp(u^T \cdot u_i^t)}$$
$$s_i = \sum_{j=1}^{n} \alpha_i^j \cdot h_i^j$$

Infine un softmax layer messo in alto che prevede le etichette dei documenti minimizzando la funzione di perdita cross-entropy tra le etichette previste e quelle reali.

$$L_c(w_d) = -\sum p \cdot log[softmax(W_d \cdot w_d + b_d)]$$

dove l'output della funzione softmax è la distribuzione delle etichette previste e p è la distribuzione delle etichette reali.

2.3.1 Spiegazione Variabili

2.4 Mutual Learning

L'idea alla base di questo modello è quella di scambiare le informazioni tra i due task per aumentarne le capiacità complessive. Questo viene eseguito tramite l'uso della distribuzione latente del topic di ogni parola ottenuta dalla parte di Topic Detection per guidare il calcolo del world-level attention signals nella text classification, quindi essenzialmente incorporando le informazioni del topic nell'allenamento del modello Text Classification. Dall'altra parte il vettore world-level attention è usato per guidare l'apprendimento della distribuzione latente del topic nella Topic Detection, perche tale vettore contiene le informazioni sul livello di polarità delle parole.

2.4.1 Funzionamento

La distribuzione latente del topic per ogni parola può essere ottenuta usando i pesi che connettono il penultimo livello e il livello di ricostruzione nella Topic Detection. Il vettore di attenzione di ogni parola nella Recurrent Neural Network è salvato in u_i^j. La distibuzione latente del topic per la i-esima parola è rappresentata come $w_i^{'} = \{w_{i1}, w_{i2}, ..., w_{iK}\}$ dove K è il numero totale dei topic. Il vettore di attenzione $u_i^{'}$ per la i-esima parola è ottenuta al quinto passaggio definito nella sezione Text classfication. Durante l'allenamento del modello cerchiamo di massimizzare la somiglianza tra le due variabili definite sopra per la i-esima parola. il nosto modello ($MTL - ML_2$) specificatamente utilizza questa misura di somiglianza:

$$\mathcal{O}_2 = \sum_i |w_i^{'} \cdot u_i^{'}| - ||w_i^{'}||_2 - ||u_i^{'}||_2$$

L'obiettivo di ottimo ora è di minimizzare la funzione di perdita definita come:

$$argmin_{\theta, w_i^{'}, u_i^{'} \in V} \sum_d (\alpha \cdot L_t(w_d) + \beta \cdot L_c(w_d)) - \sum_V (\mathcal{O}_2(w_i^{'}, u_i^{'}))$$

2.5 Spiegazione Variabili

- f_{μ_θ} e $f_{\sum_\theta}$: sono due MLP con funzione di attivazione lineare, nel codice prima vengono prese le dimensioni del vocabolario e le dimensioni del encoder ed attraverso una funzione lineare si crea la variabile pi, successivamente si crea sempre attraverso una funzione lineare i due MLP che calcolano la media e la varianza, tenendo conto della variabile pi e del numero di topic.
- w_d: questa è la variabile che identifica i diversi documenti dove w_d è il singolo documento
- θ: parametro modificabile

- ϵ: termine di errore che ha una distribuzione normale
- L: numero di campioni indipendenti
- L_t: indica la funzione di perdita e il pedice t sta per topic
- N_d: numero di parole nel documento
- $w_{d,n}$: identifica la parola nel documento
- q: l'approssimazione a posteriori calcolata dalla rete inferenziale
- z_d: la distribuzione calcolata dalla rete inferenziale (semantica latente)
- W: parametro imparato sia dalla Topic detection (W_t) che dalla classification (W_c)
- w_i^j: è il word embedding della j-esima parola nel i-esimo frase
- x_i^j: indica il primo step dove viene moltiplicato il parametro W per w_i^j
- L_c: funzione di perdita della text classification
- b_d: termine di errore per il documento d