Relazione sulla sperimentazione sui Kernel Method

A.A. 2023/2024

Rosso Carlo

Indice

1 Introduzione

La sperimentazione sui Kernel Methods è stata condotta utilizzando il software fornito e descritto nella pagina citata in Moschitti 2006. A causa di alcune incertezze riguardo le modalità operative, ho consultato il professore per chiarimenti. Durante l'incontro, abbiamo deciso di intraprendere le seguenti sperimentazioni:

- Sperimentazione su diversi dataset:
  - Un dataset per la regressione.
  - Un dataset per la classificazione binaria (positivi e negativi).
  - Cinque dataset, ciascuno rappresentante una classe distinta.
  - Dataset sui sottoalberi.
- Sperimentazione su diversi kernel:
  - Moschitti et. al. (2007): non sono sicuro sia il Partial Tree Kernel;
  - VISHTANAM and SMOLA 2002: Subtree Kernel;
  - COLLINS and DUFFY 2002: Subset Tree Kernel;
  - ZHANG, 2003: Subset Tree Bow Kernel;
  - MOSCHITTI, 2006: Partial Tree Kernel.
  - Something ibrid.

In particolare dei kernel sperimentati, per quanto riguarda gli ultimi due non sono stato in grado di effettuare delle predizioni perché il programma di classificazione fallisce in segmentation fault. Non sono stato in grado di sperimentare la funzione di kernel lineare per nessun modello, perché il programma di allenamento fallisce in segmentation fault. Infine, per tutti i modelli con funzione di kernel polinomiale, radiale o sigmoidale, il programma di allenamento ritorna un modello vuoto, ovvero un modello senza alcun support vector. Da questo deriva che le predizioni sono casuali perché il modello non impara. Dunque, l'unica funzione di kernel effettivamente sperimentata risulta essere quella di re-ranking oppure di combinazione di foreste.

2 Dataset

- È stata ottenuta una versione modificata del treebank, in cui le golden label sono state sostituite con i rispettivi POS-tag, utilizzando lo Stanford Parser. In particolare, il parser è stato utilizzato tramite il container Docker descritto nella repository NLPbox 2024.
- I due dataset risultanti sono stati convertiti nel formato svmlight.
- È stato generato un terzo dataset unendo gli alberi dei due dataset precedenti.
- Poiché il software fornito consente di effettuare sia regressione che classificazione binaria, i dataset sono stati ulteriormente elaborati.

- È stato creato un dataset contenente solo esempi positivi o negativi, unendo le classi "negativo" e "un po' negativo", e le classi "positivo" e "un po' positivo". La classe "neutro" è stata eliminata.

- Sono stati creati cinque dataset distinti per la classificazione multiclasse; ciascun dataset è stato progettato per addestrare un modello specializzato su una singola classe, ponendo tutte le classi tranne una a -1 e la classe target a 1.

- Le classi dei modelli sono state normalizzate trattando il valore come un float. In particolare, nel dataset di partenza le classi variavano da 0 a 4, quindi è stata applicata l'operazione $(\text{class} - 2)/2$, ottenendo valori compresi tra -1 e 1. In questo modo è ottenuto il treebank per allenare il modello di regressione.

- Sono stati generati dataset contenenti tutti i sottoalberi (togliendo le foglie) dei dataset con etichette di sentimenti sviluppati fino a questo punto.

Il codice utilizzato per generare i dataset è disponibile nella repository Rosso 2024. Non sono stato in grado di generare i sottoalberi per i dataset con etichette di sintassi, poiché gli alberi di sintassi differiscono per poco da quelli di sentimenti, per cui diventa complesso assegnare il target corretto ai sottoalberi. Per esempio, un albero di sintassi divide la parola "fully-written" in tre foglie "fully", "-", "written", mentre nell'albero di sentimenti è una sola foglia con etichetta "fully-written". Questa non è l'unica differenza che ho notato, ma anche una differenza così piccola rende complesso da abbinare i sottoalberi alla golden label corretta.

3 Modello

Per l'allenamento dei modelli, è stato sviluppato un programma che utilizza i dataset forniti. L'allenamento complessivo ha avuto una durata media di circa 1 ore per ciascun modello.
Il kernel method utilizzato è stato esclusivamente il subset tree kernel. Abbimao provato ad utilizzare anche il partial tree kernel, l'allenamento è andato a buon fine, tuttavia non è stato possibile predire i risultati sul dataset di test, perché il programma che esegue la classificazione termina con un segmentation fault. Risalendo alla causa del problema, abbiamo scoperto che il problema è dovuto alla normalizzazione dell'albero di test, durante la lettura del pattern.

Abbiamo provato a modificare la funzione di attivazione, in particolare abbiamo sperimentato:

- **linear**: l'allenamento non funziona, perché il programma va in segmentation fault;

- **polynomial**: l'allenamento non funziona, perché viene allenato un modello vuoto;

- **radial basis function**: l'allenamento non funziona, perché viene allenato un modello vuoto;

- **sigmoid**: l'allenamento non funziona, perché viene allenato un modello vuoto.

- **reranking**: l'allenamento funziona e il modello è in grado di predire i risultati sul dataset di test;

- **forest combination**: l'allenamento funziona e il modello è in grado di predire i risultati sul dataset di test.

Ancora, per quanto riguarda il reranking, come funzione di attivazione, abbiamo modificato i seguenti iperparametri:

- decay factor;

- normalizzazione;

- metodo di somma degli alberi.

In particolare, non è stato possibile ottenere l'acurattezza dei modelli sui nodi degli alberi che hanno decay factor diverso da 0. Non siamo stati in grado di capire il motivo di questo comportamento.
Per quanto riguarda la combinazione di foreste, abbiamo sperimentato con gli stessi iperparametri del reranking, ottenendo risultati simili.
Si noti che è possibile usare reranking come funzione di attivazione solo sul dataset che combina la sintassi e il sentiment, mentre la combinazione di foreste può essere utilizzata su tutti i dataset.

4 Risultati

In primo luogo è stato allenato lo stesso modello di dafault su ciascun dataset generato. In questo modo è stato individuato il dataset su cui confrontare i diversi modelli.
I test sono stati eseguiti con il medesimo dataset per quanto riguarda la classificazione di ciascun nodo dell'albero; mentre ciascun modello è viene valutato sulla radice dell'albero con un dataset che ha la medesima struttura del dataset di allenamento.

4.1 Binary

Modello	Accuracy	Precision	Recall	F1
Subtree sentiment all	**0.98**	**0.99**	**0.99**	**0.99**
Subtree sentiment root	0.94	0.94	0.94	0.94
Syntax and sentiment all	0.96	0.97	0.95	0.96
Syntax and sentiment root	0.94	0.94	0.94	0.94
Sentiment all	0.95	0.97	0.95	0.96
Sentiment root	**0.94**	**0.95**	**0.94**	**0.94**
Syntax all	0.45	0.00	0.00	0.00
Syntax root	0.74	0.72	0.78	0.75

Tabella 1: Risultati della classificazione binaria

4.2 Regression

Modello	Mean Absolute Error	Mean Squared Error	R-Squared
Subtree sentiment all	-	-	-
Subtree sentiment root	-	-	-
Syntax and sentiment all	**0.47**	**0.41**	**-0.94**
Syntax and sentiment root	0.94	1.28	-1.93
Sentiment all	0.61	0.66	-2.14
Sentiment root	1.17	1.95	-3.48
Syntax all	0.49	0.35	-0.65
Syntax root	**0.56**	**0.48**	**-0.11**

Tabella 2: Risultati della regressione

In questo caso mancano i risultati per il modello addestrato sui sottoalberi perché non ho completato l'allenamento, che richiedeva molto tempo: l'ho fermato a 40h di allenamento, ma non è stato sufficiente per il completamento.

4.3 Multiclass

Modello	Accuracy	Precision	Recall	F1
Subtree sentiment all	**0.95**	**0.92**	**0.91**	**0.92**
Subtree sentiment root	0.51	0.50	0.48	0.49
Syntax and sentiment all	0.44	0.48	0.59	0.44
Syntax and sentiment root	0.53	0.52	0.49	0.49
Sentiment all	0.40	0.45	0.56	0.40
Sentiment root	**0.54**	**0.54**	**0.50**	**0.50**
Syntax all	0.18	0.04	0.20	0.06
Syntax root	0.39	0.38	0.34	0.34

Tabella 3: Risultati della classificazione multiclasse

4.4 RNN

Modello	Accuracy	Precision	Recall	F1
RNN 50 unità nascoste all	0.79	0.65	0.53	0.57
RNN 50 unità nascoste root	0.42	0.44	0.39	0.40
Subtree sentiment all	**0.95**	**0.92**	**0.91**	**0.92**
Subtree sentiment root	0.51	0.50	0.48	0.49
Sentiment all	0.40	0.45	0.56	0.40
Sentiment root	**0.54**	**0.54**	**0.50**	**0.50**

Tabella 4: Risultati della classificazione multiclasse con RNN rispetto al kernel method

Si noti che l'RNN e il kernel method allenato sul sentiment sono stati allenati con lo stesso dataset. Tutti e tre i modelli qui presentati sono stati valutati sullo stesso dataset di test.

4.5 Confronto degli iperparametri

Per confrontare gli iperparametri sono stati utilizzati due database: uno con le etichette sia sintattiche che di sentiment, ci riferiamo ad esso come *merged*; l'altro con le etichette di sentiment, ci riferiamo ad esso come *sentiment*. L'accuratezza è stata calcolata sui test set che hanno la medesima struttura del dataset di allenamento.

Subset Tree Kernel

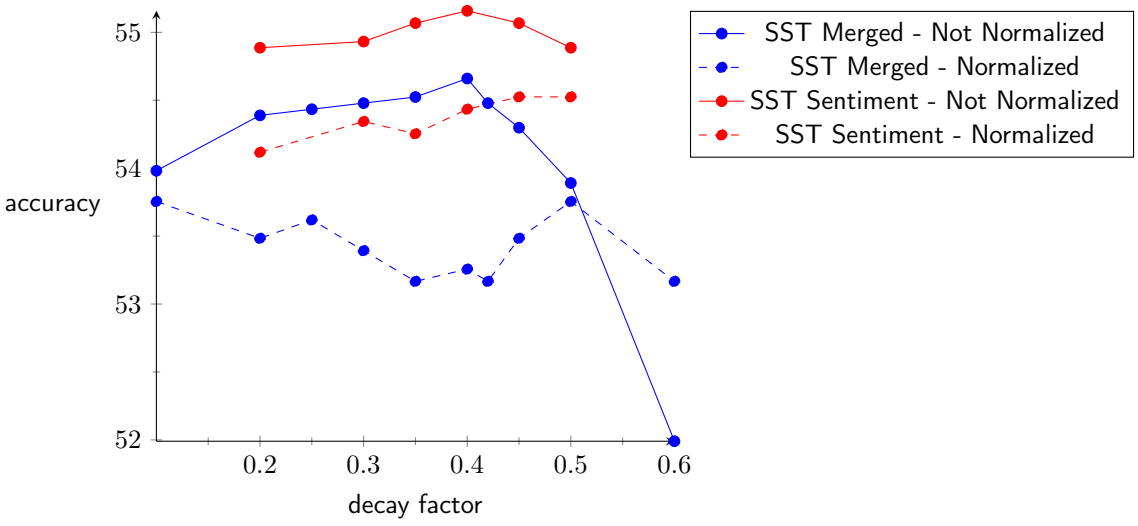

Figura 1: Confronto degli iperparametri del subset tree kernel.

Subtree Kernel

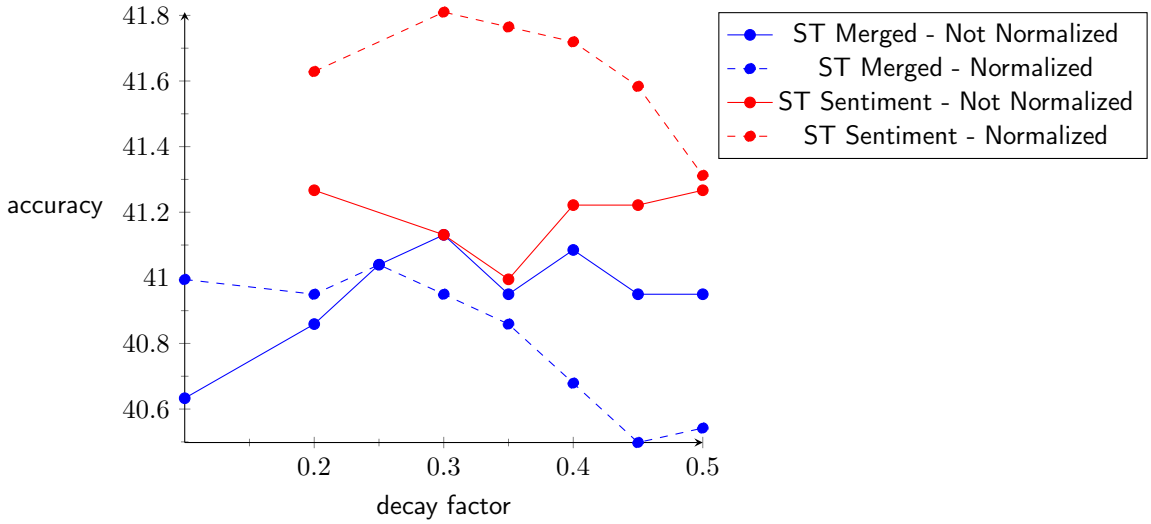

Figura 2: Confronto degli iperparametri del subtree kernel.

Subset Tree Bow Kernel

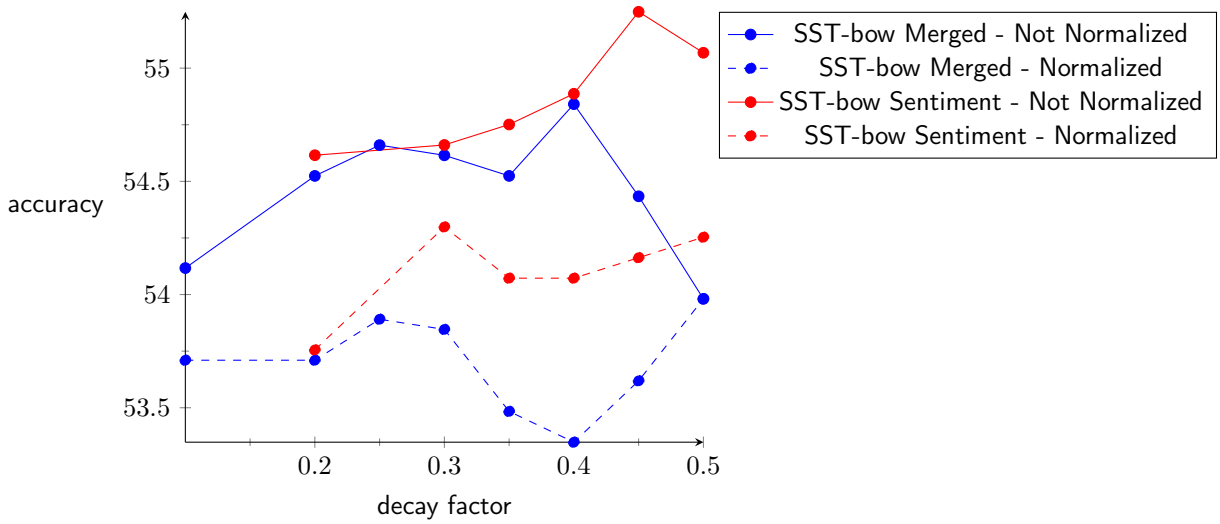

Figura 3: Confronto degli iperparametri del subset tree-bow kernel.

Partial Tree Kernel

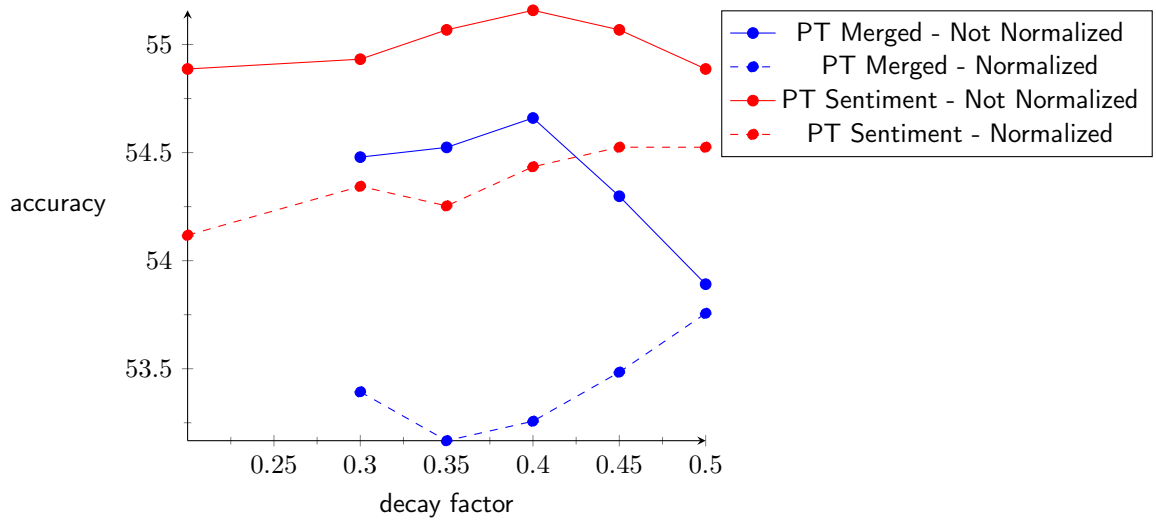

Figura 4: Confronto degli iperparametri del partial tree kernel.

Riassumendo

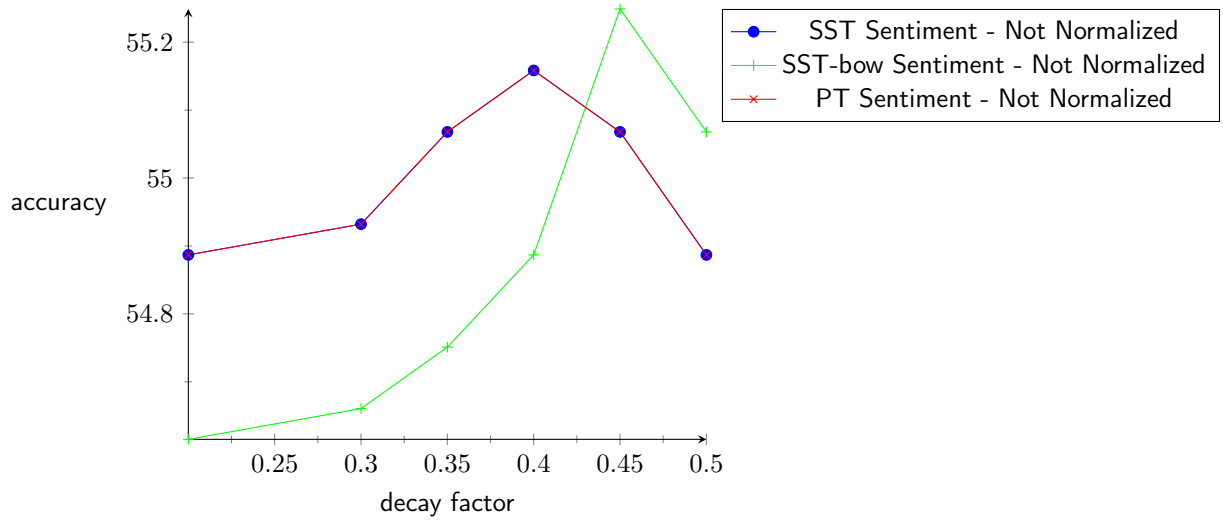

Figura 5: Confronto dei kernel method sugli iperparametri migliori, si noti che SST e PT ottengono valutazioni identiche.

5 Conclusioni

5.1 Sentiment vs Syntax vs Subtree

I modelli allenati sulla sintassi hanno mostrato prestazioni inferiori rispetto agli altri due approcci. Questa differenza è particolarmente evidente nell'analisi su tutti i nodi dell'albero. Tale risultato è prevedibile dato che il treebank di test utilizza etichette di riferimento non incluse nel treebank di allenamento dei modelli, rendendo il

confronto meno rilevante.
Nonostante ciò, anche limitando l'analisi alla radice dell'albero, i modelli basati sulla sintassi hanno registrato performance inferiori. Tale tendenza non si verifica nella regressione.
Al contrario, i modelli che integrano l'analisi del sentiment con quella sintattica tendono a superare quelli basati esclusivamente sul sentiment, anche se i risultati sono simili quando si focalizzano sulla predizione del target alla radice dell'albero.
In aggiunta, i modelli allenati sui sottoalberi hanno dimostrato un'efficacia notevolmente superiore nella classificazione dei pattern su ogni nodo, pur mantenendo risultati comparabili a quelli dei modelli basati sul sentiment per quanto riguarda la radice.

5.2 Kernel Methods vs RNN

Le tecniche basate su Kernel Methods, quando applicate ai sottoalberi, hanno evidenziato performance superiori rispetto alle RNN.
Invece, le RNN si distinguono nettamente nell'identificazione dei pattern su ciascun nodo dell'albero rispetto modelli rimanenti (modello allenato su sentiment & co.).
Si osserva, inoltre, che nei risultati relativi alla radice dell'albero, il modello che combina il sentiment con i Kernel Methods supera significativamente le RNN (0.12), evidenziando un divario maggiore di quello notato tra le RNN e il Kernel Method allenato sulla sintassi (0.3).
In ogni caso, è importante notare che il modello RNN ha dimensione fissa rispetto al numero di unità nascoste scelte per la rete, d'altro canto i kernel method aumentano di dimensione con il numero di pattern nel treebank di allenamento, per questo motivo su un dataset più grande i kernel method potrebbero risultare poco efficienti o addirittura non utilizzabili.

5.3 Iperparametri

Putroppo non è stato possibile eseguire un'analisi accurata degli iperparametri perché il software trovato fallisce nel calcolo delle performance dei modelli oppure non è in grado di classificare i pattern come richiesto.
I risultati mostrano che l'iperparametro 'W', ovvero forest sum, non è rilevante, infatti qualunque valore assuma non influisce sulle performance del modello.
La normalizzazione degli alberi tendenzialmente peggiora le performance dei modelli e ne allunga il tempo di allenamento. Infine, il decay factor migliore risulta essere di 0.4 per i modelli subset tree, partial tree, mentre il subset tree-bow kernel con decay factor di 0.45 ottiene un'accuratezza migliore.
Infine si nota che i modelli subset tree, subset tree-bow e partial tree ottengono performance comparabili: più del 55% di accuratezza nella classificazione dei pattern sulla radice dell'albero. Mentre il modello con kernel subtree raggiunge un'accuratezza poco inferiore al 42%, simile all'RNN con 50 unità nascoste.
Infine, si nota come le informazioni sintattiche aggiuntive non migliorino le performance dei modelli, ma le peggiorino. Probabilmente si ottiene questo risultato perché le frasi sono già divise in sottoalberi secondo la sintassi

e probabilmente è questa l'informazione sintattica che i modelli utilizzano per classificare i pattern, mentre le etichette di sintassi aggiuntive non sono rilevanti.

Riferimenti bibliografici

[Mos06] Alessandro Moschitti. *Tree Kernels in SVM-light*. 2006. URL: https://disi.unitn.it/moschitti/Tree-Kernel.htm (visitato il 11/07/2024).

[NLP24] NLPbox. *stanford-corenlp-docker*. 2024. URL: https://github.com/NLPbox/stanford-corenlp-docker (visitato il 11/07/2024).

[Ros24] Carlo Rosso. *bachelor-s_notes*. 2024. URL: https://github.com/danesinoo/bachelor-s_notes (visitato il 11/07/2024).