Università degli studi di
Milano-Bicocca

Advanced Machine Learning

Final Project

Previsione dell'Interazione fra Molecole e Vie Metaboliche con ANNs

Authors:
Alessia Petescia - 839141- a.petescia@campus.unimib.com
Louis Fabrice Tshimanga - 847529 -
l.tshimanga@campus.unimib.com

July 10th, 2020

Abstract

In questo progetto l'interazione di un set di molecole con specifiche vie metaboliche viene predetto utilizzando due tipi di Artificial Neural Networks (ANNs): Convolutional Neural Network 1-Dimensionali o Long Short Term Memory Units, che ricevono in input un'elaborazione della notazione SMILES che identifica ciascuna molecola. Tale notazione è resa univoca per ciascuna molecola e rileva informazioni strutturali e chimico-fisiche. I filtri convoluzionali, così come le unità LSTM, sono capaci di riconoscere pattern testuali, che a loro volta rappresentano precise sottostrutture responsabili delle interazioni fra molecole del dataset e reagenti nelle vie metaboliche target.

1 Introduzione

1.1 Chimica: Teoria e Pratica

La chimica si sviluppa come scienza prettamente empirica, basata sulla raccolta di osservazioni, analisi e tentativi pratici. Le forze responsabili delle reazioni chimiche sono tutte di natura elettromagnetica e sono rilevanti gli effetti quantistici. Pertanto la natura degli oggetti di studio è piuttosto differente dall'esperienza quotidiana, la teoria e le notazioni matematiche sottostanti sono molto recenti rispetto allo studio dei fenomeni chimici, nonché complesse e per molti aspetti predittive solo in senso statistico. In definitiva, reazioni e applicazioni difficilmente si possono derivare con un approccio top-down: dalle scienze dei materiali all'industria farmaceutica, molte scoperte arrivano per "trial and error", o in maniera poco efficiente. Ciononostante, ci sono molte leggi e regole sperimentali o pattern che riassumono l'esperienza chimica, e che si riescono a collegare alla conoscenza teoretica.

1.2 Vie Metaboliche

Nel lavoro qui presentato si è analizzata l'interferenza tra molecole di interesse tossicologico e vie metaboliche di due tipi: Nuclear Receptor (NR) signaling, Stress Response (SR) pathways. Nel primo caso si tratta di proteine, cioè macromolecole costituite da catene di amminoacidi (20 molecole specifiche), situate nella cellula, atte a recepire la presenza di specifici ormoni e modificare di conseguenza l'attività e lo sviluppo cellulare. Nel secondo caso si tratta di ampie varietà di risposte adattative della cellula per mantenere il

proprio stato fisiologico nonostante stress esterni (meccanici, fisici, chimici, ambientali, et alia). In entrambi, l'interazione fra molecole, quando nota, è riconducibile alla qualità, quantità e organizzazione degli elementi chimici nelle strutture molecolari della cellula e dei reagenti; eppure non tale interazione non è direttamente e universalmente prevedibile.

1.3 Ipotesi di Lavoro

L'ipotesi di base per questo e altri studi nel campo della chimica computazionale è che l'informazione rilevante sull'interazione tra molecole sia almeno in parte trasmessa nelle notazioni che descrivono le strutture delle molecole (il tipo di atomi e di legami che si instaurano tra di essi, la carica ionica, la configurazione elettronica, la geometria e la simmetria delle strutture etc.), e da esse ricavabile e utilizzabile da sistemi di Machine Learning.

2 Datasets

2.1 Tox21 10k

Tox21 è un'iniziativa del governo statunitense, volta allo sviluppo della tossicologia nel 21mo secolo, grazie alle nuove tecnologie sia sperimentali sia analitiche. Gli esseri umani sono esposti ad enormi quantità e varietà di agenti chimici di tossicità poco nota, contemporaneamente circa il 30% dei farmaci sperimentali promettenti nei test su animali si rivelano tossici per l'essere umano. Ciò evidenzia la necessità di analizzare velocemente ed efficientemente grandi quantità di dati diversi per origine e formato (esattamente le 3 V dei Big Data, Velocity, Variety, Volume). Nel 2014 è stata dunque lanciata una Tox21 Data Challenge, un insieme di sotto-problemi il cui obiettivo generico fosse prevedere esattamente l'esito di 12 saggiature per circa 10 000 molecole, 6 saggiature di tipo NR e 6 di tipo SR, il cui esito eventualmente determinasse se ciascuna molecola interferisse o meno con le 12 vie metaboliche. I 12 dataset originali, suddivisi in train, validation e score set, si differenziano in quanto la stessa molecola può essere o meno attiva indipendentemente sulle diverse vie, e perché non tutte le saggiature per coppia molecola-via metabolica danno risultati univoci o dirimenti. Sono resi disponibili pubblicamente in formato SMILES [`https://tripod.nih.gov/tox21/challenge/data.jsp`].

2.2 Notazione SMILES

Il "simplified molecular-input line-entry system (SMILES) of notation", è appunto un sistema di notazione che permette di associare anche in maniera biunivoca stringhe di caratteri e strutture molecolari. Il sistema è stato sviluppato negli anni '80, insieme all'incremento di dati sperimentali generati dalle nuove tecnologie elettroniche, che a loro volta richiedevano l'uso di elaboratori automatici, facilitati da notazioni codificate. L'alfabeto simbolico è costituito dalle abbreviazioni standard degli elementi ([H] per Idrogeno, [He] per Elio), insieme a segni numerici o di interpunzione ASCII per segnalare tipi di legame atomico, strutture anulari, aromaticità (proprietà elettronica di alcune strutture anulari), diramazioni e simmetrie.

2.3 Da Stringhe SMILES ad Input per ANNs

Seguendo l'approccio e l'algoritmo presentato in (Maya et al. 2018), le stringhe SMILES sono state trasformate in Feature Matrices di 42 colonne. 21 Colonne necessarie al one-hot-encoding dei simboli nativi SMILES, altre 21 colonne per raccogliere features degli atomi, come il tipo di elemento (categorie H, C, O, N, o Altro), la carica e la valenza elettronica, il numero di legami con atomi di Idrogeno, la chiralità (destra, sinistra, o altra), la saturazione e l'ibridazione degli orbitali.[1] Essendo la molecola più lunga del dataset identificata da 400 caratteri SMILES, ogni matrice è stata sottoposta a zero-padding per parificare le dimensioni. Le Feature Matrices così costruite sono state utilizzate come input delle 1-Dimensiona Convolutional Neural Networks (1-D CNNs), mentre per le Long Short Term Memory units (LSTMs) si è preferito evitare di approcciare il problema come multivariato (42 features possibilmente scorrelate), riducendo le matrici a one-hot-encodings delle stringhe SMILES: in particolare, sono state rimosse le colonne con le caratteristiche atomiche aggiuntive oltre al tipo di elemento, ciascuna riga è stata sostituita con un "indicatore" del proprio indice con elemento non-nullo al fine di apprendere un embedding efficace contemporaneamente al training dell'LSTM (vedasi sezione successiva).

3 Metodi

3.1 Linee di Lavoro

Per questo progetto sono state seguite due linee di lavoro principali per lo sviluppo di architetture e modelli, ricongiunte dal testing sui 4 score dataset. Questi sono stati separati tra NR e SR, ed in ciascun sottoinsieme un dataset è stato scelto come "sandbox". Le architetture dei modelli migliori sono sempre state applicate tentativamente a tutti i 4 dataset.

3.2 SR: Modelli per Euristica, Cross-Validation

3.2.1 CNNs

Nella linea di lavoro SR, il dataset SR-ARE è stato selezionato come sandbox per CNNs. L'architettura di base della letteratura nei prevedeva 2 layer convoluzionali seguiti ciascuno da MaxPooling o AveragePooling, un layer di GlobalAverage o GlobalMax e infine un layer denso prima dell'output. La dimensione del pooling è stata fatta variare attorno a 3, il numero di filtri per layer attorno a 100 e la dimensione dei kernel attorno a 10. In generale è emersa una certa equivalenza tra modelli, quanto a risultati sul training e validation set (splitting 0.8:0.2), con una tendenza all'overfitting entro 20 epoche e risultati sistematicamente minori tra training+validation e test set. Infine si è scelto come modello di benchmark una rete a due layer convoluzionali di 120 filtri ciascuna, kernel di 15 e Pooling in ordine Max, Max e GlobalAverage, sezione densa anticipata da dropout al 50% e con 64 unità prima dell'output. Eccetto questo, ogni unità con funzione di attivazione ReLU. Tra i principali fattori in grado di migliorare la performance del modello di base, l'oversampling, le classi pesate in maniera disuniforme ed il padding per la convoluzione. I primi due fattori sono alternativi e la performace sul test non ha rilevato vantaggi significativi. A livello metodologico, però, l'oversampling della classe minoritaria richiede accorgimenti per la Cross-Validation (CV), sicché si sono scelti i pesi delle classi, inversamente proporzionali alla frequenza della classe e dimezzati per diminuirne i valori assoluti. Il training è avvenuto minimizzando la binary cross-entropy con le impostazioni di default dell'ottimizzatore Adam di keras (in particolare, learning rate=0.001). Variazioni dei parametri o dell'algoritmo di ottimizzazione non hanno mai restituito risultati preferibili.
Configurata l'architettura di base, si sono implementate altre 3 varianti da

cross-validare. Una rete "deep" con un ulteriore layer convoluzionale uguale ai primi 2, due reti "wide" per il numero di filtri aumentato a 300, in un caso con kernel size ridotta a 8 e nell'altro aumentata a 30. In questo modo si sono potute fare dei confronti a coppie su specifiche dimensioni architetturali. Dalla 5-fold CV sono emersi valori di accuracy centrati o limitati superiormente all'85%, mentre per la Receiver Operating Characteristic Area Under the Curve (ROC-AUC, semplicemente AUC) si sono rilevati due modelli significativamente migliori, il modello "deep" e quello "wide" con kernel ampio, rispettivamente con AUC stimate in 0.92 e 0.94 ($\pm$ 0.01 in entrambi i casi), laddove il massimo per gli altri modelli si è stimato in 0.83. Queste due architetture sono state quindi riapplicate al training set complessivo e valutate sullo score dataset, complessivamente 10 volte ciascuna per raccogliere statistiche data la variabilità non eliminabile fissando i random seed.

3.2.2 Modelli LSTM

Data la natura sequenziale del problema, sia per la struttura dei dati, sia per la natura tipica degli oggetti fisici in questione, si è voluto tentare l'utilizzo di LSTM. Il problema è stato ridotto alla classificazione della stringa SMILES come sequenza di caratteri, con alfabeto espresso per one-hot-encoding (con codifica unica per gli atomi della categoria Altro) e successivo embedding. L'architettura fondamentale scelta è stata di due layer LSTM di 400 e 200 unità, seguiti da dropout al 50% verso un layer denso di 64 unità e infine l'unica unità di output. L'ottimizzazione è stata effettuata come in precedenza minimizzando la binary cross-entropy tramite Adam, con pesi sulle classi proporzionali all'inverso della frequenza. La rete così implementata è stata allenata con splitting tra training e validation set (0.8, 0.2) per 200 epoche, mostrando due comportamenti salienti e diversi rispetto alle CNNs. Di prima evidenza il tempo medio per epoca, maggiore di un fattore tra 10 e 30 (fino a 1min/epoca); in seguito si è notato un maggior accoppiamento tra le performance su training, validation e soprattutto score set, quantomeno in termini di accuracy (mentre la validation loss ha mostrato una fase di risalita e possibile plateau, al diminuire della training loss). Un modello di architettura analoga, ma basato su GRUs invece che LSTM, è stato affiancato per testare le differenze fra tipo di unità (non rilevanti in questa applicazione) e gli effetti di diverse dimensioni di embedding, in particolare scegliendo tra le formule:

$$embedding\_dim = 1.6 \times num\_categories^{0.56} \tag{1}$$

$$embedding\_dim = num\_categories^{0.25} \quad (2)$$

In quest'applicazione un embedding più contenuto (pensato per l'ambito dei word embeddings, con vocabolari assai più ampi dell'alfabeto SMILES) non si è rivelato né più veloce né più efficiente in termini di accuracy e statistiche affini. Per cui invece che 2 o 3 embedding dimensions, ne sono state utilizzate 10. Il modello risultante è stato riapplicato al training set e valutato sullo score set 10 volte, ma solamente per il dataset SR-ARE, senza raggiungere la fase di valutazione su ogni dataset.

3.3 NR: Modelli ad Iperparparametri ottimizzati

3.3.1 Riduzione preliminare del Search-Space

Nella linea di lavoro NR, il dataset selezionato come sandbox per le CNNs e la successiva ottimizzazione degli iperparametri è stato il dataset NR-AhR. Inizialmente, sono state svolte una serie di prove empiriche per restringere il campo delle possibili combinazioni applicabili. Da un'analisi iniziale, è emerso che le migliori performance erano raggiunte per un numero di filtri oscillante intorno a 100, dimensione del kernel compresa tra 10 e 30, ed un numero di layer convoluzionali non superiore a due, con un layer denso prima dell'output ed il padding completo in ogni layer. In particolare, per numeri inferiori a quanto indicato, i modelli non risultavano in grado di apprendere sufficientemente, mentre nel caso di valori maggiori, questi tendevano all'overfitting molto velocemente. Inoltre, è stato testato l'effetto di diverse funzioni di attivazione nei layers interni, dove la scelta finale è ricaduta sulla Relu. A partire da queste considerazioni, sono state analizzate architetture di sei tipi, tra cui due architetture "deep" con due convolutional layer e 4 con un solo convolutional layer, testando l'impatto che la modifica di diversi parametri. Ciascun modello è stato valutato utilizzando una 3-fold CV. L'architettura migliore è risultata essere la seguente:

- Dropout layer
- Conv1D layer
- Global Average Pooling Layer
- Dense layer
- Dropout layer

- Output layer

L'architettura scelta, ha prodotto i seguenti risultati: Tabella performances sullo score I parametri che più hanno influenzato le performances tra le diverse architetture utilizzate sono risultati essere l'inserimento di un primo layer di dropout e l'utilizzo di un Global Average Pooling. Come è stato possibile notare nella sezione precedente, le performances relative alla classe più rappresentata sono risultate sensibilmente migliori rispetto a quelle raggiunte sulla classe minoritaria. Si è voluto quindi indagare per verificare se tale risultato potesse dipendere dallo sbilanciamento del dataset. Sono state utilizzate due diverse tecniche di oversampling: random oversampling e Synthetic Minority Over-sampling Technique (SMOTE).[2] L'architettura individuata precedentemente è stata quindi addestrata nuovamente sui dataset così bilanciati. L'oversampling non ha dato benefici sullo score set, mentre nel caso della tecnica SMOTE (crossvalidata senza bisogno di modifiche), poiché vengono generati dei dati sintetici "simili" ma non identici a quelli target, si è notato un lieve miglioramento in termini di recall della classe target, con un trade-off rispetto alla precision.
Per il dataset NR-ER si sono mostrate analoghe le valutazioni sulle 6 architetture di prova, così come sugli effetti dell'oversampling classico e sintetico per bilanciare il dataset.

3.3.2 Ottimizzazione Bayesiana

Si è deciso di applicare delle tecniche di ottimizzazione per verificare se queste fossero in grado di migliorare ulteriormente le performance dei modelli precedentemente descritti. Inizialmente, si è tentato un approccio di ampio spettro, provando ad ottimizzare contemporaneamente sia la struttura dell'architettura (numero di layers convoluzionali e di pooling), sia i vari iperparametri interni, utilizzando il pacchetto Python GpyOpt per sfruttare l'ottimizzazione Bayesiana utilizzando processi gaussiani come modello. Tuttavia, date le ridotte risorse computazionali, si è successivamente deciso di restringere lo spazio da analizzare. E' stata quindi presa l'architettura ritenuta migliore tra quelle già sviluppate, e di questa ne sono stati ottimizzati gli iperparametri, ossia numero di filtri, dimensione del kernel, numero di neuroni del layer denso, nonché learning rate. Inoltre, per verificare l'effetto degli iperparametri, sono state utilizzate tre diverse funzioni obiettivo in fase di ottimizzazione, ossia: loss, f2Score e f1Score, tutte e tre riferite ai valori calcolati sul validation set. Dato l'effetto dello SMOTE rilevato in precedenza,

si è deciso infine di allenare nuovamente il modello ad iparametri ottimizzati, prima sul dataset originale, poi sul dataset sottoposto a SMOTE, per poter effettuare un confronto di performance sullo score test.

Table 1: Tabella degli iperparametri ottimizzati

Hyperparameters	**Domain**
Learning rate	$(0.1, 0.01, 0.001, 0.0001)$
Neurons	$(16, 32, 64, 128, 256, 512)$
Filters	$(1, 500)$
Kernel	$(1, 51)$

4 Risultati

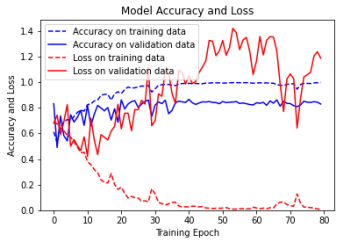

Figure 1: Storico del training della CNN di base

Table 2: Risultati su SR-ARE

Modelli	f1	precision	recall	accuracy	AUC
DeepSR	0.45 ± 0.02	0.40 ± 0.10	0.54 ± 0.10	0.78 ± 0.04	0.69 ± 0.02
WideSR	0.43 ± 0.02	0.35 ± 0.03	0.59 ± 0.10	0.74 ± 0.03	0.68 ± 0.02

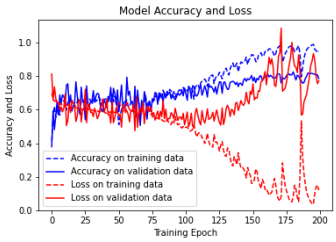

Figure 2: Storico di 200 epoche di training per LSTM di base

Table 3: Risultati su SR-MMP

Modelli	f1	precision	recall	accuracy	AUC
DeepSR	0.55 ± 0.03	0.43 ± 0.05	0.77 ± 0.06	0.86 ± 0.02	0.82 ± 0.02
WideSR	0.52 ± 0.03	0.40 ± 0.06	0.81 ± 0.10	0.84 ± 0.03	0.83 ± 0.03
(LSTM)	0.38 ± 0.02	0.29 ± 0.02	0.57 ± 0.03	0.70 ± 0.01	0.65 ± 0.02

5 Discussione

I risultati ottenuti dalle CNNs sono in linea con i metodi automatici originariamente impiegati nella Tox 21 Data Challenge 2014 e a tratti vicini allo stato dell'arte, per come tutti questi sono riportati in (Maya et al. 2018). Nonostante uno sbilanciamento nei dataset quasi costante (rapporto 1:5 tra target e classe secondaria), sembra che la variabilità fra i risultati dipenda dal target metabolico specifico. All'interno del singolo dataset, una grande varietà di architetture dà performance simili e l'ottimizzazione ha un ritorno di valutazione non immediata: tendenzialmente si migliora leggermente l'f1score sbilanciando precision e recall, piuttosto instabili, aumentando considerevolmente i costi computazionali e temporali che sono invece un punto a favore delle CNNs. Si nota anche che le LSTM potrebbero essere valide in termini di risultati, anche per la vicinanza già menzionata tra risultati di training, validation e test, ma sono particolarmente svantaggiose in termini computazionali ed il ritorno marginale si riduce molto.

Table 4: Risultati su NR-AhR

Modelli	f1	precision	recall	accuracy	AUC
Opt F2 (dataset og)	0.56	0.45	0.75	0.87	0.87
Opt F2 (dataset SMOTE)	0.37	0.23	0.96	0.61	0.87

Table 5: Risultati su NR-ER

Modelli	f1	precision	recall	accuracy	AUC
Opt F2 (dataset og)	0.34	0.24	0.62	0.76	0.73
Opt F2 (dataset SMOTE)	0.23	0.14	0.80	0.49	0.71

6 Conclusioni

Nel campo della drug discovery, la componente empirica ricopre un ruolo fondamentale. Tuttavia, spesso questa porta con sé lunghi tempi di sperimentazione e notevoli costi. In tale contesto, le tecniche di machine learning e deep learning stanno acquisendo sempre più rilevanza all'interno del settore. In questo elaborato, attraverso la rappresentazione degli SMILES come matrici sparse di features, si è voluto testare l'efficacia di due strumenti tipici del deep learning:le CNN e le LSTM. I risultati ottenuti hanno dimostrato che questi modelli possono essere degli strumenti utili per riconoscere l'eventuale interazione fra diverse molecole, anche in presenza di un forte sbilanciamento, essendo più efficienti di un approccio casuale. Tuttavia, tali modelli sono da considerarsi dei punti di partenza, migliorabili da diversi punti di vista, dove un possibile sviluppo futuro potrà essere l'estensione dell'ottimizzazione ad un numero maggiore di iperparametri.

References

[1] "Convolutional neural network based on SMILES representation of compounds for detecting chemical motif", Maya Hirohara et al, BMC Bioinformatics, 2018.

"Toxicity Prediction Method Based on Multi-Channel Convolutional Neural Network", Qing Yuan et al, PMC, 2019