Lezione 3
Apprendimento Automatico: Tecniche e Metodologia

Vito Roberto
Dipartimento di Matematica, Informatica e Fisica (DMIF)
Università di Udine
E-mail: vito.roberto@uniud.it

1.- Introduzione

Affronteremo lo studio di una procedura di apprendimento automatico in una strategia supervisionata. In particolare presenteremo uno schema iniziale detto *a correzione di errore*, e ne daremo uno schema più specifico adottando la tecnica di ottimizzazione della *discesa lungo il gradiente*, di cui daremo alcuni dettagli. Infine, illustreremo i passi di una tipica metodologia di progettazione con attenzione anche alle problematiche di tipo numerico e statistico che possono manifestarsi in corso di sperimentazione.

2.- Apprendimento supervisionato: uno schema di procedura

Una strategia supervisionata si basa su uno schema detto *a correzione di errore (error-correcting framework).* Riportiamo lo schema articolato in passi logici.

Passo 1: Inizializzare i pesi delle connessioni

Passo 2: Calcolare l'errore a partire dalla differenza tra l'output osservato e quello noto a-priori

Passo 3: Calcolare la correzione da applicare ai pesi delle connessioni

Passo 4: Aggiornare i valori dei pesi applicando le correzioni calcolate al passo 3

Passo 5: Ripetere i passi 2, 3, 4 per ognuno dei training data

Passo 6: Ripetere i passi 1, 2, 3, 4, 5 fino a che l'errore raggiunge un valore prefissato

Figura 1. Schema generico di procedura a correzione di errore.

Lo schema di procedura in Figura 1 è espresso ancora in forma generica – si osservi il Passo 3 – e richiede ulteriori specifiche.
L'idea è inquadrare l'apprendimento – affrontiamo il caso supervisionato - nell'ambito dei *problemi di ottimizzazione,* cioè della ricerca dei minimi di una funzione di più variabili. Ciò vuol dire: cercare i valori delle variabili *(parametri, coefficienti)* che rendono minima una *funzione di errore* opp. *funzione costo,* cioè una funzione della distanza tra l'output prodotto dalla procedura e il valore corretto, che è noto a priori.
Le variabili sono i pesi delle connessioni (matrice **W** nella Lezione 2, par.5): quando la rete opera in fase di apprendimento *(training mode)* l'architettura si modifica *(plasticità, auto-configurazione)* fino a raggiungere una configurazione stabile, in corrispondenza dei valori dei pesi che rendono minima la funzione costo. Viceversa, i pesi sono delle costanti quando la rete non opera in modalità di apprendimento: ad esempio, quando la fase di training è andata a buon fine, e la rete opera in modalità di test.

3.- La tecnica del Gradient Descent (GD)

3.1 Uno schema più specifico

Riscriviamo lo schema in Figura 1, specificandolo con uno schema di ottimizzazione per *Discesa lungo il Gradiente (GD).*

Passo 1: Inizializzare le variabili,
cioè i pesi delle connessioni

Passo 2: Calcolare la funzione di errore (Fcost)
in corrispondenza dei valori correnti delle variabili

Passo 3: Approssimare la derivata del costo:
delta ≈ derivative(Fcost)

Passo 4: Modificare i valori delle variabili in modo
da ridurre il valore della funzione costo:
W = W – *alpha****delta**
dove *alpha* numero reale, learning rate

Passo 5: Ripetere i passi 2, 3, 4
per ciascuno dei training data

Passo 6: Valutare il costo complessivo dopo una scansione dell'intero training set. Iterare i passi 1, 2, 3, 4, 5 fino a che il costo raggiunge il minimo entro un intervallo prefissato

Figura 2. Schema di procedura a correzione di errore di tipo GD.

3.2 - Una variante: lo Stochastic Gradient Descent (SGD)

Lo schema al paragrafo precedente richiede di calcolare un output per ciascun elemento dei training data: ciò è poco efficiente quando si ha a che fare con insiemi di dati molto grandi. In tali casi si usa una variante del GD chiamata Stochastic Gradient Descent.
In questa variante l'aggiornamento delle variabili viene effettuato dopo ogni singolo elemento dell'insieme dei dati di apprendimento *(training dataset),* e non dopo un certo numero di elementi, chiamato *batch,* par.4.1.
Il primo passo della procedura richiede che l'ordine del training dataset sia modificato in modo casuale *(randomized).* Lo si fa per rimescolare l'ordine con cui vengono aggiornati i valori delle variabili, ed evitare fluttuazioni eccessive nel calcolo della funzione costo.
Si è osservato che con l'SGD si ottiene una notevole velocizzazione della procedura per grandi insiemi di dati.

4.- Parametri e iperparametri

4.1 Terminologia

Sono *iperparametri* i parametri il cui valore è usato per controllare il processo di apprendimento; in contrasto, i valori di altri parametri – tipicamente i pesi associati ai nodi – si ottengono tramite la procedura di apprendimento.
Dati gli iperparametri la procedura è in grado di apprendere i parametri dal training set.
E' opportuno distinguere tra iperparametri *del modello* e *della procedura*. Questi ultimi influenzano la velocità e la qualità dell'apprendimento, ma non le prestazioni del modello. Esempi del primo tipo sono la topologia e le dimensioni di una rete. Esempio di iperparametro della procedura è il tasso di apprendimento *alpha* (par.3.1).

Presenteremo gli iperparametri della procedura chiamati: *elemento dei dati (sample); blocco di dati (batch); epoca (epoch).*
-*Elemento dei dati (sample).* Comprende l'input della procedura e l'output atteso, da confrontare con quello calcolato (previsto) per poi calcolare l'errore. E' anche chiamato *osservazione; vettore di input; vettore delle caratteristiche (feature vector).*
-*Blocco di dati (batch).* Il valore del batch specifica il numero di sample da elaborare prima di aggiornare i parametri del modello. Al termine di un batch si calcolano i dati previsti; si confrontano con gli output attesi; si calcola l'errore. Poi si applica la regola di apprendimento, cioè le modifiche alle variabili per migliorare il modello, muovendosi in direzione della discesa lungo il gradiente.
-*Epoca (epoch).* Iperparametro che specifica il numero di volte in cui la procedura di apprendimento scandisce l'intero training set. Una singola epoca comporta che ogni elemento del training set ha una possibilità di aggiornare i valori delle variabili. Un'epoca comprende uno o più batch.

In definitiva una procedura di apprendimento è dotata di un doppio ciclo di controllo: uno sul numero di epoche, cioè di scansioni dell'intero training set; all'interno di questo ciclo, un altro ciclo di controllo itera su ciascun batch di elementi. Generalmente il numero di epoche è dell'ordine delle centinaia o migliaia, in modo da rendere l'errore della procedura sufficientemente piccolo.
Non vi sono regole formali per fissare gli iperparametri; occorre caso per caso stabilire dei valori adeguati. Ad esempio, supponiamo di avere un dataset di 400 elementi;

scegliamo un batch di dimensione 5 e 1000 epoche. Ciò comporta che il dataset è diviso in 80 batch di 5 elementi ciascuno; i pesi delle connessioni saranno aggiornati dopo ciascun batch. In termini diversi, una singola epoca comprende 80 batch e vi sono 80 aggiornamenti dei pesi del modello. Con 1000 epoche il modello scandisce 1000 volte l'intero dataset, per un totale di 80.000 aggiornamenti dei pesi durante l'intero processo di apprendimento.

4.2 Soluzioni per l'aggiornamento dei pesi

L'aggiornamento dei pesi - applicazione della regola di apprendimento - può essere realizzato adottando una tra le alternative seguenti:

-*Metodo Stochastic Gradient Descent (SGD).* Si è visto al par.4.2 che l'errore è calcolato per ogni elemento dei dati, e i pesi vengono aggiornati immediatamente. Quindi, all'interno di un'epoca, i pesi vengono aggiornati tante volte quanti sono i training data. Inoltre i risultati risentono di effetti casuali (fluttuazioni statistiche).

-*Metodo Batch.* Gli errori sono calcolati per ognuno dei training data, ma non si aggiornano subito i pesi. In un *blocco di dati (batch)* si calcola la media aritmetica degli aggiornamenti da apportare ai pesi, ed è questa che viene usata per applicare la regola di apprendimento. Questa soluzione è appesantita proprio dal calcolo delle medie;

-*Metodo Mini-batch.* Si divide l'intero dataset in batch e si opera su un batch alla volta, calcolando l'errore delta dei singoli training data. Poi si calcola l'aggiornamento dei pesi facendo la media dei delta appena calcolati. Il mini-batch tenta di combinare la velocità d'esecuzione del metodo SDG con la stabilità del metodo batch.

5.- Una metodologia di progettazione

Il progetto, la realizzazione e il collaudo di una rete neurale seguono in generale una metodologia articolata in fasi di sviluppo, che qui presentiamo. Bisogna considerare che ogni modello di rete neurale è concettualmente diverso dagli altri; ne consegue che anche le fasi di sviluppo sono differenti e richiedono specifiche soluzioni. Dunque non esiste una metodologia unica, valida per tutti i modelli, ma è possibile solo individuare linee generali valide in modo orientativo.

Riportiamo in questo paragrafo una sintesi dei passi, che svilupperemo nei paragrafi che seguono.

Una metodologia di sviluppo tipicamente si articola in:

- Raccolta dei dati *(data loading)*
- Pre-elaborazione dei dati *(preprocessing)*
- Definizione del modello e dell'architettura
- Inizializzazioni
- Fase di apprendimento della rete
- Collaudo (test) e valutazioni di prestazioni
- Fasi di revisione

5.1 Raccolta e pre-elaborazione dei dati

Richiede in genere l'accesso a un database esterno *(database, data samples),* e la realizzazione di un *archivio locale (datastore).*

La raccolta dati può richiedere la *separazione (splitting)* del datastore in tre sottoinsiemi: quello dei *dati di apprendimento* - il *training set* in senso proprio; quello dei *dati di verifica* (anche chiamato *validation set).* Infine, il *test set* è l'insieme dei dati di collaudo delle funzioni che la rete deve svolgere – ad esempio, classificare immagini – per effettuare test dell'avvenuto apprendimento.

La pre-elaborazione dei dati *(data preprocessing)* può migliorare l'efficienza e la stabilità dell'apprendimento della rete neurale. Le elaborazioni sono di tipo statistico e geometrico. Esempi sono:
- Scalare, normalizzare (scaling, normalizing) gli input e i risultati attesi in modo che i valori rientrino in un intervallo predefinito, ad esempio l'intervallo [0, 1];
- Normalizzare la stima della media e della deviazione standard dei valori del training set. Un caso comune è ottenere media nulla e deviazione standard unitaria.
- Ridurre le dimensioni dei vettori di input applicando l'analisi nelle componenti principali (PCA);
- Effettuare un'analisi preliminare di regressione tra la risposta di rete e i corrispondenti risultati attesi.

5.2 Definizione del modello e dell'architettura
Dimensionare la rete: stabilire il numero di nodi. Individuare l'ordinamento spaziale (topologia): strati; indicizzazione degli strati; connessioni.
Si passa alle specifiche dell'architettura del modello, definendone le caratteristiche matematiche come illustrato nella Lez.2, par.5. In particolare si definiscono le funzioni di attivazione ai nodi.
Dato un obiettivo da raggiungere, non esistono criteri per stabilire la topologia ottimale e i flussi dei dati in un modello, per cui occorre applicare *criteri empirici* e affidarsi a esperimenti ripetuti.

5.3 Inizializzazioni
Si assegnano i valori iniziali ai parametri definiti dall'architettura. Ai pesi delle connessioni e ai bias sono spesso assegnati valori casuali. A tale riguardo è importante considerare che *un modello neurale è per sua natura statistico:* pertanto sono da affrontare anche problemi di stima statistica e stabilità.
I valori di attivazione dello strato di input sono inizializzati con i valori del vettore di input corrente. In una strategia supervisionata si assegna *allo strato di output* anche il corrispondente valore atteso.

5.4 Apprendimento (learning, training)
Naturalmente, si tratta della fase più importante dell'intera metodologia. Si è illustrato ai precedenti paragrafi 2,3 uno schema di procedura supervisionata impostato come problema di ottimizzazione di una funzione a più variabili, con algoritmi di discesa lungo il gradiente.
Dal punto di vista progettuale-realizzativo è una fase difficile e computazionalmente costosa. Le difficoltà sorgono dalle problematiche del calcolo numerico-statistico da affrontare: accenniamo ad alcune di esse.

- Mal-condizionamento (ill-conditioning);
- Esistenza di minimi locali dei gradienti (local minima);

- Esistenza di plateau nei domini locali delle variabili.

Per risolvere questi problemi sono state proposte varianti delle tecniche di GD (par.3), che costituiscono oramai un insieme di *tecniche di ottimizzazione specializzate:* ciascuna dà origine ad algoritmi e implementazioni specifiche.
Esempi tra le tecniche disponibili sono: la Stochastic Gradient Descent (SGD, par.3.2) con l'aggiunta di un *termine di momento;* l'ottimizzazione del tasso di apprendimento *(learning rate optimization)* chiamata *tecnica ADAM (ADAptive Moments).*
Anche per il caso di un dato problema di apprendimento non esistono criteri per individuare l'algoritmo di ottimizzazione adeguato, e occorre affidarsi alla sperimentazione.
La fase di apprendimento termina tipicamente con la valutazione dell'efficacia ed efficienza dell'apprendimento stesso. Possiamo ad esempio calcolare la curva della stima dell'errore di apprendimento rispetto alle epoche: naturalmente ci aspettiamo che l'errore diminuisca *(loss curve, learning curve).*
Riassumendo e riprendendo lo schema al par.3.1, il flusso del calcolo in una tipica fase di apprendimento supervisionato si articola nelle seguenti sottofasi:

- Inizializzazione;
- Propagazione in avanti *(feedforward step);*
- Stima dell'errore;
- Aggiornamento dei pesi per propagazione all'indietro *(backpropagation step);*
- Ciclo sul training set;
- Ciclo sulle epoche;
- Valutazioni di prestazioni: curve di apprendimento *(loss curves).*

5.5 Collaudo (test) della rete

E' anche chiamata *fase di generalizzazione.* Problemi che si presentano in questa fase sono *l'underfitting* e *l'overfitting.*
-*Underfitting.* La rete non consente di ottenere un errore sufficientemente piccolo sul training set, nonostante la sperimentazione ripetuta;
-*Overfitting.* La rete ha superato la fase di apprendimento del training set, ma non generalizza a nuovi input in modo soddisfacente. Produce un errore piccolo sul training set, ma più grande quando si presentano alla rete i dati del test set. Tecniche per migliorare la generalizzazione comprendono:
- *La regolarizzazione*. Si modifica la stima dell'errore che il processo di apprendimento deve ridurre al minimo. La regolarizzazione produce una rete che apprende meglio i dati di training, e mostra un comportamento più fluido quando vengono presentati nuovi dati;
- *L'arresto anticipato.* Utilizza due set di dati distinti: il training set in senso proprio, per aggiornare i pesi; il validation set (par.5.1), per interrompere l'addestramento quando la rete inizia ad avere problemi di overfitting dei dati.

Infine, la fase di test comprende la valutazione delle prestazioni della rete rispetto agli obiettivi finali del progetto.