



SAPIENZA
UNIVERSITÀ DI ROMA

Analisi della Sample Complexity in Reti Neurali Feedforward

Facoltà di Scienze Matematiche, Fisiche e Naturali
Laurea Triennale in Scienze Matematiche per l'Intelligenza Artificiale

Rosamaria Graziosi
Matricola 2083375

Relatore
Prof. Luca Becchetti

Correlatore
Prof. Fabrizio Silvestri
Dott.ssa Maria Sofia Bucarelli

Anno Accademico 2024/2025

Analisi della Sample Complexity in Reti Neurali Feedforward
Tesi di Laurea Triennale. Sapienza Università di Roma

© 2025 Rosamaria Graziosi. Tutti i diritti riservati

Questa tesi è stata composta con L^AT_EX e la classe Sapthesis.

Sito web: https://github.com/WrongMedal/Th_Sample_Complexity
Email dell'autore: graziosi.rosamaria@gmail.com

Abstract

Questa tesi investiga empiricamente la sample complexity delle reti neurali feedforward (FNN). Il lavoro si articola in tre parti principali.

Nella prima parte viene presentato il framework teorico del PAC Learning e della VC-dimension, con particolare attenzione al bound $O(W \log W)$ proposto da Bartlett et al. (2019).

Nella seconda parte vengono descritti tre esperimenti comparativi. I risultati mostrano discrepanze significative rispetto alle predizioni teoriche: il bound appare eccessivamente conservativo, con reti che raggiungono buone performance con quantità di dati molto inferiori a quelle suggerite dalla teoria. Emerge inoltre un ruolo cruciale del tempo di addestramento e del criterio di early stopping.

Nella terza parte vengono discusse le limitazioni del framework basato su VC-dimension e presentate misure alternative di complessità, tra cui la Natarajan dimension per problemi multiclass, la Rademacher complexity, la fat-shattering dimension, e approcci basati su stabilità algoritmica e teoria dell'informazione.

Il lavoro evidenzia come la teoria classica, pur fornendo garanzie di generalizzazione rigorose, non catturi adeguatamente le dinamiche di apprendimento, suggerendo la necessità di framework che incorporino aspetti algoritmici.

Ringraziamenti

Un grazie alla mia famiglia, devo loro molto per avermi permesso di intraprendere questo percorso. Un grazie a zia Luciana e zio Michele per tutto l'aiuto e per i momenti trascorsi insieme. Un grazie ai miei nonni per credere così tanto in me. Un grazie a Clarissa, Gaia, Natalia, Eliseo, Eleonora, Aurora e a tutti i miei amici, vecchi e nuovi, vicini e lontani, che hanno reso questi 3 anni meno solitari. Un grazie a Luigi e Luca per avermi sopportato in appartamento con loro durante la stesura di questo lavoro.

Vi sono molto riconoscente.

Grazie.

Indice

1 Introduzione	1
2 Cenni sull'apprendimento statistico	3
2.1 Notazioni e Convenzioni adottate	3
2.2 Paradigma PAC Learning	4
2.3 VC-dimension	5
2.4 Notazione Asintotica	7
2.5 Classe delle Reti Neurali Feedforward	8
3 Riscontro Empirico	11
3.1 Concezione degli esperimenti e struttura delle reti	11
3.2 Esperimenti	14
3.2.1 <i>Exp3</i> - Analisi con epoche limitate	14
3.2.2 <i>Exp4</i> - Estensione a reti più ampie e verifica dell'ipotesi sui parametri addestrabili	18
3.2.3 <i>Exp5</i> - Ruolo del tempo di addestramento: early stopping, instabilità e fenomeni non monotoni	20
3.3 Studio del rango	26
4 Criticità del framework e alternative alla VC-Dimension	29
4.1 Misure con convergenza uniforme	29
4.1.1 Natarajan dimension	29
4.1.2 Rademacher complexity	30
4.1.3 Fat-Shattering Dimension	31
4.2 Altre tipologie di misure	31
4.2.1 PAC-Bayesian Theory	32
4.2.2 Stabilità algoritmica	32
4.2.3 Information-Theoretic Generalization Bounds	33
5 Riepilogo e riflessioni finali	35
Bibliografia	37

Capitolo 1

Introduzione

Nell'ambito della Computer Science vi è interesse nel misurare le "necessità" degli algoritmi, cioè quantificare le risorse da essi utilizzate per svolgere le proprie operazioni con successo. Per tale ragione per ciascuna tipologia di risorsa si definisce una cosiddetta misura di complessità corrispondente.

Se ci concentriamo sugli algoritmi di apprendimento, una delle principali risorse utilizzate sono i dati e il tema di questa lavorazione riguarda proprio tale oggetto: parliamo di sample complexity, cioè del quantitativo di dati sufficiente a raggiungere un buon livello di performance. Più precisamente l'obiettivo è indagare le discrepanze tra la letteratura e la pratica circa la sample complexity di Reti Neurali Feedforward tramite uno studio comparativo di 3 reti.

Stimare questo fabbisogno dei modelli non permette soltanto di fare una loro classificazione. Infatti non è inconsueto avere accesso a pochi dati etichettati. Un esempio in ambito medico è discusso in Khadka et al. (2022) ma non è l'unico lavoro, né l'unico ambito, in cui si cerca di ovviare alla scarsità di dati. In questi contesti la scelta di un'architettura può esser dettata da vincoli di disponibilità del dataset, da qui anche l'utilità di conoscerne o poterne stimare la sample complexity.

Contesto

Il Machine Learning è un insieme di metodi che permettono a un sistema di apprendere modelli e relazioni a partire dai dati. Invece di programmare esplicitamente le regole per svolgere un compito, si fornisce all'algoritmo un insieme di esempi da cui ricavare una funzione che generalizzi a nuovi casi. I problemi più comuni includono classificazione, regressione, clustering e generazione di dati. Le tecniche spaziano da modelli lineari a reti neurali profonde, ognuno con diversi compromessi in termini di capacità, interpretabilità e costi computazionali. L'obiettivo finale è costruire modelli che generalizzino bene, evitando sia l'underfitting sia l'overfitting, e garantendo prestazioni affidabili su dati mai visti.

Il campo del Machine Learning si suddivide, approssimativamente, in branche in base al contesto in cui sono immersi gli algoritmi considerati. Quello adottato in questo testo è il seguente:

- Supervised Learning

I dataset sono provvisti di annotazioni corrette, senza rumore.

- Offline Batch Learning

Gli algoritmi di apprendimento hanno accesso a tutto (in batch) il dataset di training durante l'addestramento e non viene modificato in corso d'opera.

È importante chiarire questi assunti preliminari poiché è dimostrato che, in assenza di conoscenza a priori sulla struttura del problema, nessun algoritmo è intrinsecamente migliore di un altro. È possibile definire problemi, situazioni e input per cui un determinato algoritmo fallisce. Il risultato in questione si chiama No Free Lunch Theorem. Esso è, più precisamente, una famiglia di risultati formali per cui un guadagno di prestazioni su un insieme di compiti è necessariamente compensato da una perdita su un altro insieme di compiti. Alcuni degli enunciati più famosi di questa famiglia sono quelli in Wolpert and Macready (1997), Wolpert (2001) per i metodi Bayesiani, Cesa-Bianchi and Lugosi (2006) per contesti con avversari. Questi teoremi rendono esplicito che senza assunzioni sul problema – cioè senza introdurre bias induktivi, vincoli, modelli o ipotesi strutturali – non si può ottenere generalizzazione in senso non banale. Definire il contesto operativo è il punto di partenza per ottenere delle garanzie teoriche.

Capitolo 2

Cenni sull'apprendimento statistico

Questo capitolo introduce rapidamente il paradigma PAC Learning, la definizione di sample complexity e il suo legame con la VC-dimension. Tutto ciò è poi declinato al caso specifico delle Reti Neurali Feedforward.

2.1 Notazioni e Convenzioni adottate

L'algoritmo di apprendimento A , dato un training set S , produce un modello di predizione h . Il training set S è una sequenza finita di coppie i.i.d. di datapoint x_i e etichette y_i . Queste coppie vengono estratte da $\mathcal{X} \times \mathcal{Y}$ tramite la distribuzione congiunta \mathcal{D} , sconosciuta. Vi è una distribuzione di probabilità sulle etichette perché non sappiamo se sono assegnate deterministicamente e per ammettere il caso in cui ci sia rumore su di esse. L'obiettivo dell'algoritmo A è individuare h tale per cui la probabilità di

$$\mathbb{P}_{(x,y) \sim \mathcal{D}}[h(x) \neq y]$$

sia bassa. Possiamo osservare che, nel caso binario per cui $\mathcal{Y} = \{0, 1\}$ questa probabilità è equivalente a

$$\mathbb{E}_{(x,y) \sim \mathcal{D}}[\mathbf{1}_{h(x) \neq y}]$$

Questa prima nozione di errore si può estendere sostituendo alla funzione indicatrice un'altra funzione, la loss function $\ell(h, (x, y))$, che può misurare varie forme di errore rispetto al risultato desiderato, anche quando y non è binaria. La loss precedente, definita tramite funzione indicatrice, è anche chiamata 0-1 loss. Poiché in generale la loss dipende da (x, y) essa è una variabile aleatoria; di conseguenza il suo valore atteso è ben definito.

Sostituendo nella notazione \mathcal{Z} a $\mathcal{X} \times \mathcal{Y}$, possiamo allora definire:

$$L_{\mathcal{D}}(h) = \mathbb{E}_{z \sim \mathcal{D}}[\ell(h, z)] = \int_{\mathcal{Z}} \ell(h, z) P_{\mathcal{D}}(z) dz$$

dove $P_{\mathcal{D}}$ è la misura di probabilità associata a \mathcal{D} . Questa funzione rappresenta il rischio atteso (*expected risk*) o errore di generalizzazione (*generalization error*) del

modello h . Nella letteratura, è anche chiamata *true risk* (Shalev-Shwartz and Ben-David, 2014), *population risk* (Boucheron et al., 2005), oppure *expected prediction error* (Hastie et al., 2009), a seconda del contesto

Sottolineiamo che, affinché esista $L_{\mathcal{D}}(h)$, è necessario che la loss, dato h , sia misurabile, cioè che esista $(\mathcal{Z}, \mathcal{F}, P_{\mathcal{D}})$ spazio di probabilità, dove $\mathcal{F} \subseteq \mathcal{P}(\mathcal{Z})$ è una σ -algebra su \mathcal{Z} , tale che:

$$\forall a \in \mathbb{R}^+, \quad \{z \in \mathcal{Z} : \ell(h, z) \leq a\} \in \mathcal{F}$$

dove $\mathcal{P}(\mathcal{Z})$ è l'insieme delle parti di \mathcal{Z} .

Questa richiesta è sufficiente poiché ℓ , dato h , ha per dominio \mathbb{R}^p per un certo p e codominio i reali positivi (v. Proposizione 2.3 in Folland (1999) caso σ -algebra di Borel su $B(\mathbb{R}^+)$). Oltretutto chiediamo che ℓ sia integrabile per poter valutare significativamente il rischio all'interno dell'algoritmo A.

Purtroppo $L_{\mathcal{D}}(h)$ è sconosciuta, poiché \mathcal{D} non è nota. Si può allora utilizzare il suo surrogato empirico ottenuto tramite S :

$$L_S(h) := \frac{1}{m} \sum_{i=1}^m \ell(h, z_i)$$

dove m è la cardinalità di S .

Un popolare algoritmo d'apprendimento, introdotto da Vapnik (1999), seleziona un modello basandosi sulla minimizzazione del rischio empirico (Empirical Risk Minimization, ERM). L'idea alla base di ERM è di scegliere l'ipotesi nella classe \mathcal{H} tale che:

$$h_{\text{ERM}} \in \arg \min_{h \in \mathcal{H}} L_S(h).$$

La "bontà" di questa scelta dipende dalla capacità del rischio empirico di approssimare il rischio atteso, tema al centro della teoria dell'apprendimento statistico. Sotto l'ipotesi di campioni i.i.d., in letteratura sono state studiate le condizioni sotto cui il modello prodotto da ERM generalizza bene, stabilendo limiti per il divario tra rischio empirico e rischio atteso (ad esempio in Anthony and Bartlett (1999)). Tipicamente le FNN sfruttano questa logica all'interno dell'ottimizzatore dei pesi.

2.2 Paradigma PAC Learning

"PAC" sta per "Probably Approximately Correct" ed è un paradigma proposto da Valiant. Esso si prefigge questo obiettivo: con alta probabilità - Probably - l'errore del modello selezionato deve essere piccolo - Approximately Correct.

In questo paradigma la sample complexity $m_{\mathcal{H}}$ è una funzione che individua il numero di samples necessari a garantire queste due richieste di attendibilità e di correttezza, parametrizzate rispettivamente con $1 - \delta$ ed ϵ .

Definizione 2.1 (Agnostic PAC Learnable).

Una classe di ipotesi \mathcal{H} è detta Agnostic PAC Learnable rispetto a un insieme \mathcal{Z} e a una loss function $\ell : \mathcal{H} \times \mathcal{Z} \rightarrow \mathbb{R}^+$ se $\exists m_{\mathcal{H}}$ è un algoritmo di apprendimento t.c $\forall \epsilon, \delta \in (0, 1)$ e $\forall \mathcal{D}$ distribuzione su \mathcal{Z} , addestrando l'algoritmo con $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ samples i.i.d. generati da \mathcal{D} , l'algoritmo restituisce $h \in \mathcal{H}$ con probabilità almeno $1 - \delta$ sulla scelta degli esempi t.c.

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon$$

dove $m_{\mathcal{H}} : (0, 1)^2 \rightarrow \mathbb{N}$ è detta sample complexity function ed è minimale.

Siamo nel caso agnostico perché a priori non sappiamo se $\min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') = 0$, ovvero non sappiamo se esiste il modello corretto nella classe \mathcal{H} selezionata e neppure se esiste una certa \mathcal{H} che soddisfi questa richiesta.

La probabilità $1 - \delta$ codifica l'impatto della scelta degli esempi: potremmo pescare dei samples non rappresentativi. Essi, a loro volta, influiscono sul rischio empirico, per questo nel paradigma PAC si introduce una definizione che cerca di controllarne gli effetti, ovvero quella di insieme ϵ -rappresentativo.

Definizione 2.2 (Insieme ϵ -rappresentativo).

Il training set S si dice ϵ -rappresentativo rispetto al dominio \mathcal{Z} , alla classe di ipotesi \mathcal{H} , alla loss ℓ e alla distribuzione \mathcal{D} se

$$\forall h \in \mathcal{H} \quad |L_S(h) - L_{\mathcal{D}}(h)| \leq \epsilon.$$

Lemma 2.1.

Sia S un training set $\frac{\epsilon}{2}$ -rappresentativo rispetto al dominio \mathcal{Z} , alla classe di ipotesi \mathcal{H} , alla loss ℓ e alla distribuzione \mathcal{D} . Allora, ogni output h_ di un algoritmo ERM addestrato su S , soddisfa*

$$L_{\mathcal{D}}(h_*) \leq \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon$$

Queste due definizioni introdotte possono essere intese come attributi della classe \mathcal{H} : ci teniamo a descriverla poiché la sua struttura può influire molto su $m_{\mathcal{H}}$ e, ricordiamo, l'obiettivo finale è selezionare la famiglia di modelli corretta, in base ai requisiti richiesti (ϵ e δ) e disponibilità di campioni.

Ad esempio, è dimostrabile che tutte le classi finite sono Agnostic PAC Learnable (Thm. 2.2 in Mohri et al. (2018)). Purtroppo la classe di nostro interesse, quella delle FNN, non è finita: bisogna introdurre delle nozioni ulteriori per incapsulare la sua sample complexity.

2.3 VC-dimension

La Vapnik–Chervonenkis dimension è stata introdotta all'interno del lavoro di Vapnik and Chervonenkis (1971) anche se non con il nome con cui è nota oggi. Questo e l'integrazione col paradigma PAC Learning è invece dovuta a Blumer et al. (1989), come viene evidenziato in (Vovk et al., 2015).

La natura della VC-dimension è combinatoria ed è legata a task di classificazione binaria. Adottarla comporta quindi delle restrizioni: i task di regressione devono essere intesi come multiclass (interpretazione permessa se si considera come i computer memorizzano i numeri) e a loro volta i task multiclass devono essere presi nella loro versione One VS All o All pairs. Ciò avrà delle conseguenze, analizzate nel capitolo 4, ma la VC-dimension funge da trampolino per le successivamente generalizzazioni. Nella notazione la principale restrizione introdotta è questa: $\mathcal{Y} = \{0, 1\}$.

Definizione 2.3 (Dicotomia).

Data una classe di ipotesi \mathcal{H} , una dicotomia di un insieme C è una delle possibili modalità di etichettare i punti di C utilizzando un'ipotesi $h \in \mathcal{H}$.

Definizione 2.4 (Restrizione o Restriction).

La restrizione di $\mathcal{H} = \{h : \mathcal{X} \rightarrow \{0, 1\}\}$ a un insieme $C = \{c_1, \dots, c_n\} \subseteq \mathcal{X}$ è un sottoinsieme di $\{0, 1\}^{|C|}$, definito come:

$$\mathcal{H}_C = \{(h(c_1), \dots, h(c_n)) : h \in \mathcal{H}\}$$

ovvero l'insieme di tutte dicotomie di C realizzabili tramite \mathcal{H} .

Definizione 2.5 (Frammentazione o Shattering)).

Una classe di ipotesi \mathcal{H} frammenta (o shatters) un insieme finito $C \subseteq \mathcal{X}$ se la restrizione di \mathcal{H} a C coincide con l'insieme di tutte le funzioni da C a $\{0, 1\}$. In altre parole se

$$|\mathcal{H}_C| = 2^{|C|}.$$

Definizione 2.6 (VC-dimension).

La VC-dimension di una classe di ipotesi \mathcal{H} , indicata con $\text{VCdim}(\mathcal{H})$, è la massima cardinalità di un insieme $C \subseteq \mathcal{X}$ che può essere frammentato da \mathcal{H} . Se \mathcal{H} è in grado di frammentare insiemi di dimensione arbitrariamente grande, si dice che \mathcal{H} ha VCdim infinita.

Teorema 2.1.

Sia \mathcal{H} una classe di ipotesi con dimensione VC infinita. Allora, \mathcal{H} non è PAC learnable.

Teorema 2.2 (Teorema Fondamentale dell'Apprendimento Statistico - Restrizione ad Agnostic PAC Learnability – Vers. Quantitativa).

Sia $\mathcal{H} = \{h : \mathcal{X} \rightarrow \{0, 1\}\}$ e sia ℓ la 0–1 loss. Supponiamo $\text{VCdim}(\mathcal{H}) = d < \infty$. Allora $\exists C_1, C_2 > 0$ t.c. \mathcal{H} è Agnostic PAC Learnable con sample complexity

$$C_1 \frac{d + \log(1/\delta)}{\epsilon^2} \leq m_{\mathcal{H}}(\epsilon, \delta) \leq C_2 \frac{d + \log(1/\delta)}{\epsilon^2};$$

La dimostrazione si può trovare nel libro di Shalev-Shwartz and Ben-David (2014), al capitolo 28.

In ambito statistico la VC-dimension può essere annoverata tra gli strumenti per lo studio della *consistency* degli stimatori con condizione di convergenza uniforme e distribution-free. Più precisamente:

Definizione 2.7 (Algoritmo consistente).

Siano \mathcal{D} distribuzione, S_m training set con cardinalità m estratto su \mathcal{Z} , ℓ loss function, \mathcal{A} un algoritmo di apprendimento e $h_m = A(S_m)$ successione di modelli al crescere di m . \mathcal{A} è detto consistente se

$$L_{\mathcal{D}}(h_m) \xrightarrow[m \rightarrow \infty]{\mathbb{P}} \inf_{h \in \mathcal{H}} L_{\mathcal{D}}(h),$$

dove \mathbb{P} sta per convergenza in probabilità.

Cioè un algoritmo è consistente se il rischio del modello prodotto converge quasi sicuramente al rischio minimo possibile all'interno della classe \mathcal{H} quando la dimensione del campione tende all'infinito. Possiamo dedurre la consistenza tramite la VCdim in quanto equivalente alla richiesta di convergenza uniforme.

Definizione 2.8 (Convergenza uniforme).

Dati \mathcal{H}, S un training set di m esempi i.i.d. secondo una distribuzione \mathcal{D} su \mathcal{Z} . Si dice che \mathcal{H} gode di convergenza uniforme sul rischio empirico se

$$\sup_{h \in \mathcal{H}} |L_S(h) - L_{\mathcal{D}}(h)| \xrightarrow[m \rightarrow \infty]{\mathbb{P}} 0.$$

Invece la richiesta di distribution-free è intrinseca nel Paradigma PAC: ricordiamo che la definizione 2.1 richiede validità $\forall \mathcal{D}$.

2.4 Notazione Asintotica

Nella teoria della complessità degli algoritmi, la notazione O-grande viene utilizzata per descrivere il comportamento asintotico di una funzione rispetto alla dimensione del suo input. In altre parole, essa consente di caratterizzare come cresce il tempo di esecuzione o l'uso di memoria o, nel nostro caso la sample complexity, di un algoritmo al crescere dell'input.

Definizione 2.9 (Notazione O-grande).

Sia $f : \mathbb{N} \rightarrow \mathbb{R}^+$ una funzione che rappresenta, ad esempio, il numero di operazioni di un algoritmo in funzione della dimensione dell'input n . Si dice che

$$f(n) = O(g(n))$$

se esistono costanti positive c e n_0 tali che

$$f(n) \leq c g(n) \quad \forall n \geq n_0,$$

dove $g(n)$ è una funzione di riferimento che definisce il limite superiore asintotico della crescita di $f(n)$.

La notazione O-grande permette quindi di confrontare algoritmi indipendentemente dai dettagli implementativi e dai fattori costanti, concentrandosi sul comportamento predominante per grandi valori di n . Ad esempio, un algoritmo il cui tempo di esecuzione cresce quadraticamente con la dimensione dell'input può essere descritto come $O(n^2)$, mentre un algoritmo lineare come $O(n)$. Questa rappresentazione risulta particolarmente utile nell'analisi della scalabilità di modelli di apprendimento automatico, reti neurali e metodi numerici, dove il numero di parametri o il volume dei dati può diventare molto grande.

In combinazione con il Teorema Fondamentale dell'Apprendimento Statistico (versione quantitativa) questa notazione permette di esprimere in maniera compatta come la quantità minima di campioni necessari cresca rispetto ad un oggetto più comodo che descrive il modello, ad esempio per gli MLP, come discusso nella prossima sezione, sfrutteremo il numero di parametri. In altre parole, possiamo confrontare modelli di dimensioni diverse senza dover determinare esattamente costanti o termini di ordine inferiore.

2.5 Classe delle Reti Neurali Feedforward

Una Rete Neurale Feedforward (FNN) è un modello computazionale costruito su un grafo aciclico diretto $G = (V, E)$ i cui nodi sono organizzati in strati (layer) e gli archi, pesati, partono esclusivamente da un layer l per arrivare al layer $l + 1$. In questo lavoro, in particolare, si prende in considerazione il caso MLP, "completamente connesso", ovvero ogni nodo del layer l è connesso a tutti i nodi del layer $l + 1$. Unica eccezione sono i neuroni che rappresentano i bias, i quali non hanno archi entranti.

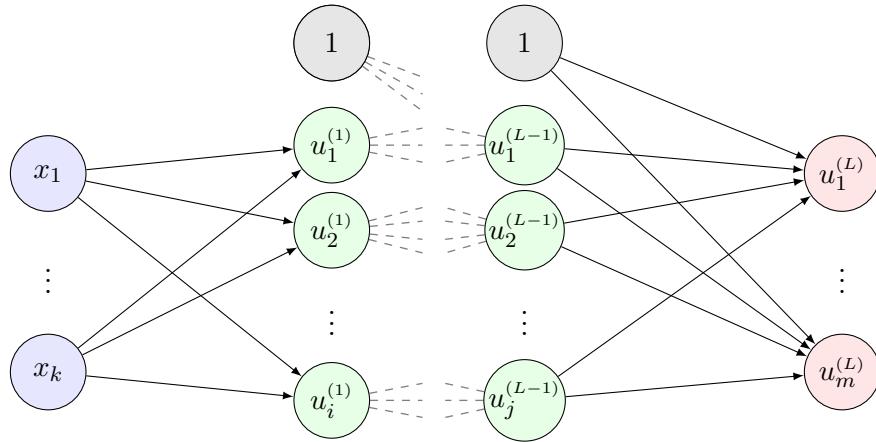


Figura 2.1. Struttura di un MLP

La computazione rappresentata da questo grafo può essere espressa come composizione di funzioni, dove ogni funzione codifica le trasformazioni applicate da un layer:

$$f(\mathbf{x}) = f^{(L)} \circ f^{(L-1)} \circ \dots \circ f^{(2)} \circ f^{(1)}(\mathbf{x})$$

dove $\mathbf{x} \in \mathbb{R}^k$ è l'input e $f^{(l)}$ è la trasformazione dovuta al layer l . Ogni layer infatti applica al proprio input una trasformazione affine e successivamente una funzione non lineare chiamata attivazione:

$$f^{(l)}(\mathbf{u}^{(l-1)}) = \sigma^{(l)}(\mathcal{W}^{(l)}\mathbf{u}^{(l-1)} + \mathbf{b}^{(l)})$$

dove, con d_l dimensione del layer l :

- $\mathbf{u}^{(l-1)} \in \mathbb{R}^{d_{l-1}}$ è l'output del layer precedente ($\mathbf{u}^{(0)} = \mathbf{x}$)
- $\mathcal{W}^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$ è la matrice dei pesi associata
- $\mathbf{b}^{(l)} \in \mathbb{R}^{d_l}$ vettore di bias
- $\sigma^{(l)} : \mathbb{R}^{d_l} \rightarrow \mathbb{R}^{d_l}$ funzione d'attivazione, applicata componente per componente

Otteniamo così una descrizione della classe di ipotesi $\mathcal{H}_{V,E,\sigma}$ in cui si collocano le FNN. Come già accennato precedentemente questa classe non è finita poiché i pesi e bias appartengono ai reali. La ricerca al suo interno di uno specifico modello h è chiamata addestramento/training e avviene tramite un algoritmo. Come accennato

nella sezione 2.1, questo algoritmo è ERM, esempi ne sono SGD (Stochastic Gradient Descent) oppure Adam (Adaptive Moment). Il processo di apprendimento stabilisce i valori di $\mathcal{W}^{(l)}$ e $b^{(l)}$ invece σ è scelta a priori. La funzione di attivazione influisce sulla $\text{VCdim}(\mathcal{H}_{V,E,\sigma})$, anche drasticamente.

Esempio 2.1.

Sia, con $c > 0$,

$$s(x) = \frac{1}{1 + e^{-x}} + cx^3 e^{-x^2} \sin a$$

Data una rete neurale con 3 layer dove l'input layer ha un nodo, l'hidden layer ha 2 nodi e l'output layer ha un nodo, costruiamo la classe a cui appartiene \mathcal{H}^* in modo tale che ci siano funzioni di questa forma

$$x \mapsto \text{sgn}(w_0 + w_1 s(x - b_1) + w_2 s(x - b_2)),$$

ovvero con $\text{sgn}()$ e $s()$ usate come attivazioni e con $w_0, w_1, w_2, b_1, b_2 \in \mathbb{R}$ pesi e bias.
Allora

$$\text{VCdim}(\mathcal{H}^*) = \infty$$

Questo è un esempio preso da Anthony and Bartlett (1999) paragrafo 7.2, in cui si trova anche la dimostrazione del risultato ottenuto. Ciò vuol dire che nonostante la funzione di attivazione sia abbastanza regolare, limitata e monotonamente crescente, con la derivata di s monotonamente crescente a sinistra dello zero e decrescente a destra:

$$\frac{d^2}{dx^2} s(x) = \begin{cases} < 0, & \text{se } x > 0, \\ > 0, & \text{se } x < 0. \end{cases}$$

non abbiamo assicurazione che $\text{VCdim} < \infty$.

Fortunatamente vi sono dei risultati per cui, date delle ipotesi sulla struttura delle funzioni d'attivazione, riusciamo a limitare dall'alto la VCdim . In particolare per funzioni lineari a tratti e polinomiali a tratti sussistono i bound dimostrati in Bartlett et al. (2019). Nell'articolo si sostiene anche che, usando ReLU come attivazione:

$$\text{VCdim}(\mathcal{H}_{V,E,\text{ReLU}}) = O(W \log W)$$

dove W è il numero di parametri, pesi e bias, da non confondere con la matrice dei pesi dei vari layers $\mathcal{W}^{(l)}$.

Dominio (datapoints)	$\mathcal{X} \subseteq \mathbb{R}^n$
Dominio (etichette)	$\mathcal{Y} \subseteq \mathbb{R}^k$
Dominio completo	$\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$
Training set	$S = ((x_1, y_1), \dots, (x_m, y_m))$ con $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y} \quad 1 \leq i \leq m$
Distribuzione congiunta	\mathcal{D}
Misura di probabilità associata a \mathcal{D}	$P_{\mathcal{D}}$
Classe dei modelli	\mathcal{H}
Modello	$h : \mathcal{X} \rightarrow \mathcal{Y}, \quad h \in \mathcal{H}$
Algoritmo di apprendimento	$A : S \rightarrow \mathcal{H}$
Algoritmo ERM	$A(S) = h_{\text{ERM}}$ t.c. $h_{\text{ERM}} \in \arg \min_{h \in \mathcal{H}} L_S(h)$
Loss function	$\ell : \mathcal{H} \times \mathcal{Z} \rightarrow \mathbb{R}^+$
True Risk	$L_{\mathcal{D}}(h)$
Empirical Risk	$L_S(h)$
Sample complexity della classe	$m_{\mathcal{H}}(\epsilon, \delta)$
Margine di errore accettabile	ϵ
Probabilità di fallimento	δ
Dimensione di Vapnik–Chervonenkis	$\text{VCdim}(\mathcal{H})$
Numero totale di parametri in un MLP	W
Classe di ipotesi delle FNN con V nodi, E archi e attivazione ReLU	$\mathcal{H}_{V,E,\text{ReLU}}$

Tabella 2.1. Riepilogo notazione

Capitolo 3

Riscontro Empirico

Il limite superiore proposto da Bartlett et al. (2019) costituisce un contributo rilevante nel tentativo di comprendere i requisiti di generalizzazione delle reti neurali. Questo bound suggerisce che la quantità di dati necessari per garantire una buona performance cresca in modo quasi lineare con la dimensione della rete. Tuttavia, una domanda fondamentale rimane aperta: quanto questo limite teorico riflette il comportamento reale delle reti neurali? L'obiettivo principale di questo capitolo è indagare empiricamente se il bound $O(W \log W)$ catturi effettivamente le dinamiche di apprendimento osservabili nella pratica. Per fare ciò, ci concentreremo su un'analisi comparativa di tre architetture di complessità crescente.

È importante sottolineare fin da subito una questione metodologica. La VC-dimension e con essa il paradigma teorico che sottende il bound $O(W \log W)$ sono stati originariamente sviluppati per problemi di classificazione binaria. Ciononostante, gli esperimenti presentati in questo capitolo coinvolgono task di classificazione multiclasse. Questa scelta risponde a un duplice obiettivo: da un lato, riflette la realtà applicativa delle reti neurali, che sono utilizzate più in contesti multiclasse che binari; dall'altro, permette di verificare se il bound teorico possa catturare, almeno qualitativamente, anche questa casistica più generale. L'analisi di questa discrepanza ci condurrà a esplorare alternative e generalizzazioni della VC-dimension.

3.1 Concezione degli esperimenti e struttura delle reti

Il primo esperimento nasce chiedendosi che succede al bound se parte dei parametri non è "addestrabile". Concorrono questi alla sample complexity?

È stata costruita una rete (Rete B) con il primo layer non addestrabile cioè con i pesi non aggiornati dall'optimizzatore. Per comparare il suo comportamento sono state allenate altre due reti (A e C) con funzione di benchmark. Sottolineamo che i pesi sono stati inizializzati stocasticamente. Infatti un'inizializzazione deterministica in questo contesto può essere paragonata ad un preprocessing. Il preprocessing tradizionale agisce sugli input applicando una trasformazione come normalizzazione, standardizzazione, ecc. Se i pesi del primo layer, non aggiornabili, fossero anche inizializzati deterministicamente vuol dire che ad ogni passo iterativo dell'addestramento stiamo usando sempre la stessa trasformazione lineare sugli input: nient'altro che un preprocessing computazionalmente inefficiente. In questo lavoro non è di inte-

resse centrale ma vi sono già articoli che mostrano come il preprocessing contribuisca al livello di generalizzazione di un modello (v. Kotsiantis et al. (2006)).

Prima di passare alla descrizione più dettagliata delle tre reti ricordiamo che grazie al teorema fondamentale dell'apprendimento statistico (Teorema 2.2) possiamo usare la VC-dimension per limitare la sample complexity, dati ϵ e $1 - \delta$. Ignoriamo il contributo di ϵ e $1 - \delta$ nell'ottica della notazione O-grande introdotta.

Rete A

- Tutti i parametri addestrabili
- Un input layer di ampiezza 28×28 (grandezza dell'input)
- 3 hidden layers di ampiezza n , variabile all'interno degli esperimenti, con ReLU funzione d'attivazione
- Un output layer di 10 logits, convertiti in probabilità con la funzione softmax.

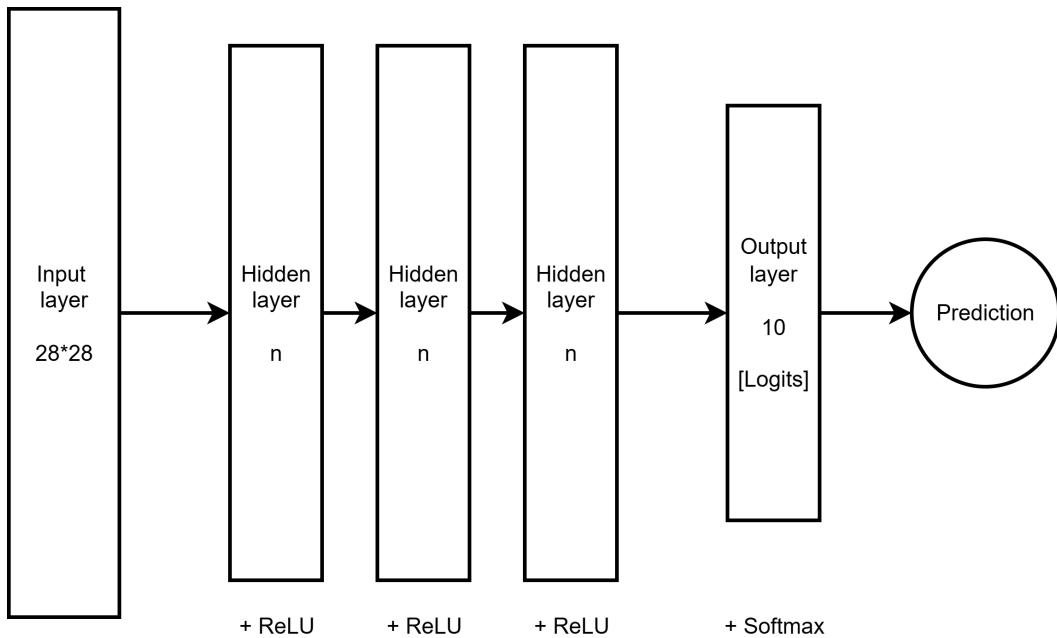


Figura 3.1. Architettura della Rete A

Rete B

- Un input layer di ampiezza 28×28 (grandezza dell'input)
- 4 hidden layers di ampiezza n , variabile all'interno degli esperimenti, ReLU funzione d'attivazione
- I pesi del primo hidden layer **NON SONO ADDESTRABILI**: restano fissi dopo l'inizializzazione
- Un output layer di 10 logits, convertiti in probabilità con la funzione softmax.

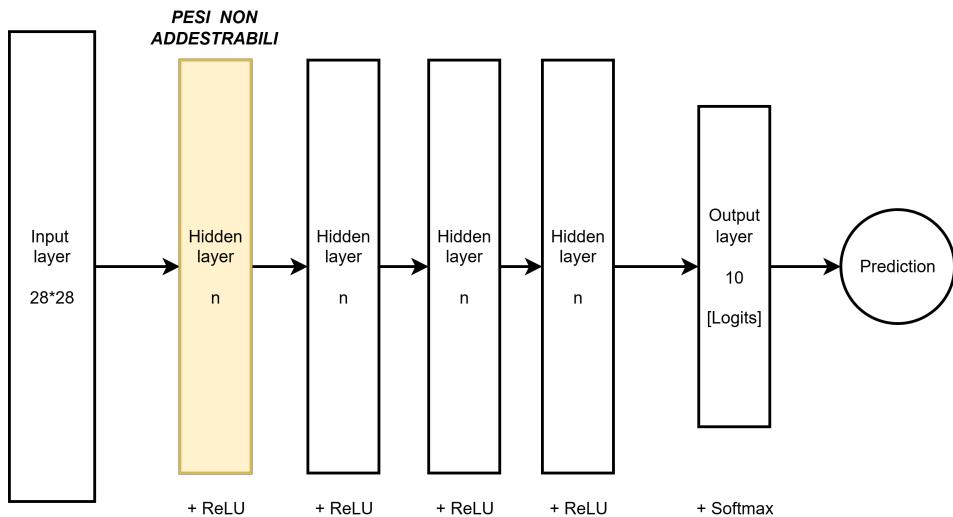


Figura 3.2. Architettura della Rete B

Rete C

- Tutti i parametri addestrabili
- Un input layer di ampiezza 28*28 (grandezza dell'input)
- 4 hidden layers di ampiezza n, variabile all'interno degli esperimenti, con ReLU funzione d'attivazione
- Un output layer di 10 logits, convertiti in probabilità con la funzione softmax.

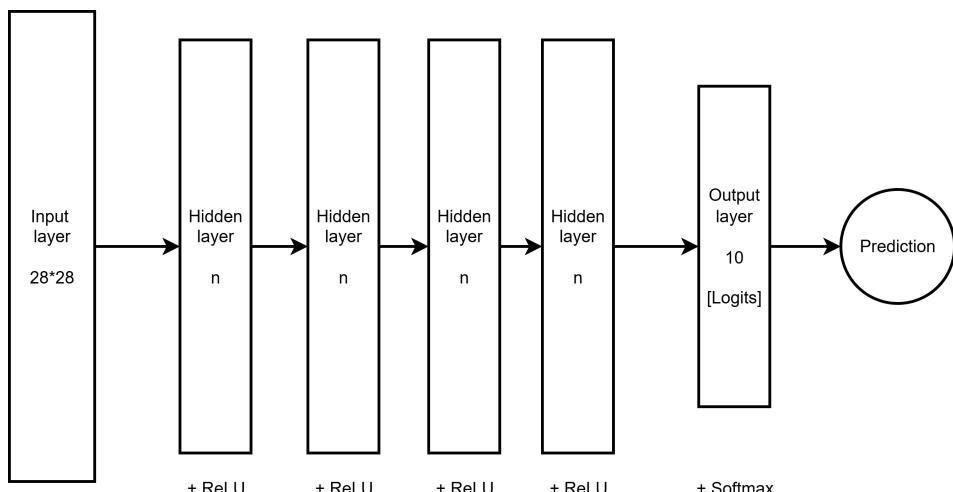


Figura 3.3. Architettura della Rete C

Inizializzazione e Ottimizzatore

L'inizializzazione delle Reti A e C è quella di default di PyTorch Lightning:

$$\text{param} \sim U\left(-\sqrt{\frac{1}{|\text{in_features}|}}, \sqrt{\frac{1}{|\text{in_features}|}}\right)$$

dove U indica la distribuzione uniforme. Per la Rete B, invece, i pesi non addestrabili sono inizializzati secondo lo schema di **Xavier/Glorot** che è una nota euristica (v. Goodfellow et al. (2016)):

$$\text{param} \sim U\left(-\sqrt{\frac{6}{|\text{in_features}| + |\text{out_features}|}}, \sqrt{\frac{6}{|\text{in_features}| + |\text{out_features}|}}\right)$$

I restanti parametri della rete B (quelli addestrabili) usano l'inizializzazione di default di PyTorch Lightning. L'ottimizzatore utilizzato è SGD con learning rate pari a 0.001, la loss è la cross-entropy:

$$L_S(h) = -\frac{1}{m} \sum_{i=1}^m \sum_{k=1}^{10} y_{i,k} \log(h(x_{i,k}))$$

3.2 Esperimenti

Il dataset scelto è MNIST (Modified National Institute of Standards and Technology): un insieme di immagini in bianco e nero di cifre scritte a mano, 55.000 di queste sono state riservate al training set, 5000 al validation set e 10.000 al test set. Questo dataset è comunemente impiegato per addestrare algoritmi di Machine Learning e per fare benchmark. Esso è una rielaborazione di dataset già esistenti proposta in LeCun et al. (1998).

Su questo dataset sono stati condotti 3 esperimenti, nominati *Exp3*, *Exp4*, *Exp5* (la numerazione segue quella della repository Github associata a questo testo). Ciascuno prevede l'addestramento delle reti su 3 seed diversi (0, 1, 42), su un training set S di cardinalità crescente; l'insieme delle cardinalità considerate è indicato in equazione (1).

$$|S| \in \{8000, 12.000, 25.000, 32.000, 40.000, 47.000, 55.000\} \quad (1)$$

Per ogni rete e per ogni seed, in funzione di $|S|$, sono stati misurati il rischio empirico e l'accuracy sul test set. Questi valori sono stati poi mediati sui 3 seed. Nelle prossime figure sono presentate queste due medie.

3.2.1 *Exp3* - Analisi con epoche limitate

Nell'esperimento *Exp3*, il numero di neuroni per hidden layer, ovvero n , è stato scelto in {8, 16, 32, 64, 128, 256, 512}. Inoltre l'addestramento è stato condotto per esattamente 20 epoche. Questa scelta è motivata principalmente da due ragioni: da un lato, consentire un primo addestramento rapido dei modelli; dall'altro, eliminare possibili variazioni nelle performance dovute a differenze nel numero di epoche, in modo da garantire un confronto diretto e controllato tra le architetture.

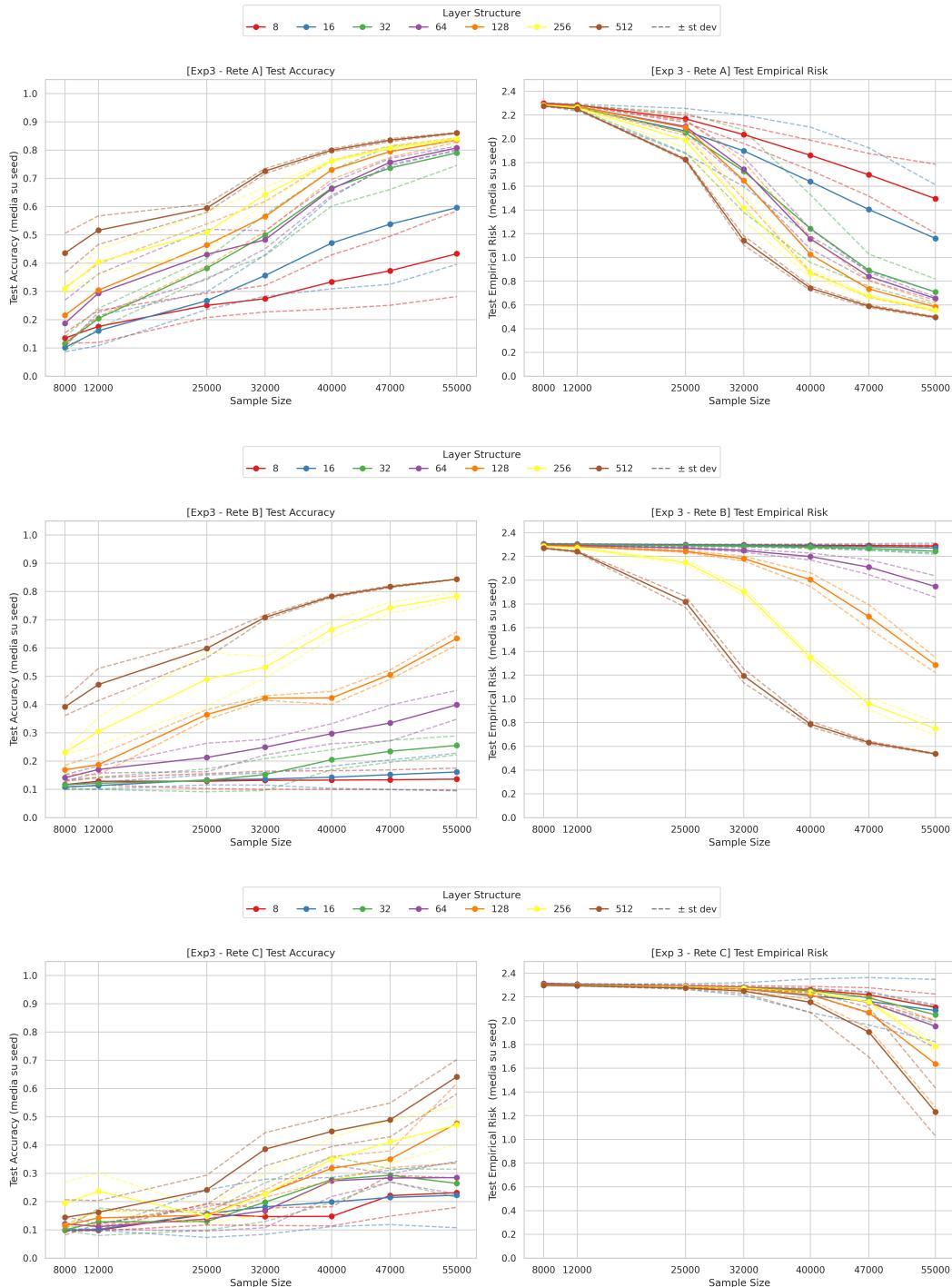


Figura 3.4. *Exp3 - Reti A, B e C*
Accuracy su Sample size & Empirical Risk su Sample size

Alla luce del bound sulla sample complexity, ci si aspetta che, al crescere di W — e quindi passando da Rete A, a Rete B, fino a Rete C — sia necessario un numero progressivamente maggiore di campioni per osservare un incremento significativo dell'accuracy. Più precisamente, per osservare un aumento significativo dell'accuracy

ci aspettiamo serva un numero maggiore di samples. Se immaginiamo di fare regressione lineare quello che vogliamo succeda è che la slope diminuisca leggendo i tre grafici dell'accuracy dall'alto verso il basso. A colpo d'occhio pare che sia questo il caso.

Seguire questo percorso $A \rightarrow B \rightarrow C$ non è l'unico modo per osservare gli effetti dell'aumentare dei parametri. Infatti W aumenta anche con l'aumentare di n . Se piuttosto analizziamo una rete alla volta pare che in architetture con n più grande aumentare la sample size ha un'influenza maggiore sull'accuracy, il che contraddice la precedente osservazione. Il fatto che questi incrementi avvengano più rapidamente può essere legato a $|S| << W$. Infatti già con $n = 64$ nella rete A ci sono quasi 60mila parametri (Tab. 3.1) ovvero più parametri che dati in tutto MNIST. Sebbene questo possa spiegare l'andamento dell'accuracy in funzione di n purtroppo indica anche che il bound $O(W \log W)$ è lontano dall'essere tight. Un altro esempio più estremo: in Rete A, con meno di 60mila dati riusciamo a raggiungere un'accuracy maggiore dell'80% con $n = 256$, ovvero avendo quasi 135 mila parametri.

Inoltre si può osservare, sempre qualitativamente, che la Rete B ha un comportamento ibrido: con n basso si comporta come la Rete C, con n alto come la Rete A. Sembra anomalo stando alle osservazioni precedenti. Una possibile spiegazione è la seguente: la sample complexity di Rete B è legata al numero di parametri addestrabili e non al totale.

Sia $W_{\text{add}}^{(B)}$ il numero di parametri addestrabili di Rete B e $W^{(A)}$ il numero di parametri di Rete A. Chiaramente:

$$\begin{aligned} W_{\text{add}}^{(B)} &= n + 3(n^2 + n) + (10n + 10) \\ W^{(A)} &= (28^2 n + n) + 2(n^2 + n) + (10n + 10) \end{aligned}$$

dove ogni parentesi rappresenta il numero di parametri tra due layers, leggendo l'architettura da sinistra a destra. Il termine additivo all'interno delle parentesi è dovuto alla presenza dei *bias*.

$$\begin{aligned} W_{\text{add}}^{(B)} - W^{(A)} &= n^2 - (28^2 - 1)n \\ W_{\text{add}}^{(B)} &= W^{(A)} + n^2 - (28^2 - 1)n \end{aligned}$$

Allora:

$$n^2 - (28^2 - 1)n \geq 0 \Rightarrow W_{\text{add}}^{(B)} \geq W^{(A)}$$

cioè se:

$$n \geq 784$$

Ovviamente escludiamo il caso $n \leq 0$ perché non ha senso rispetto alla costruzione delle reti: $n > 0$ è un requisito. Poiché in questo esperimento $n \leq 512 \leq 784$ quello che potremmo star osservando non è un comportamento a metà tra Rete A e Rete C, piuttosto una convergenza di B ad A.

Tabella 3.1. Parametri in reti A, B e C dell'esperimento *Exp3*

(a) Rete A

Size Hidden Layer	Parametri addestrabili	Parametri NON addestrabili
8	6514 — 6.5K	0
16	13274 — 13.3K	0
32	27562 — 27.5K	0
64	59210 — 59K	0
128	134794 — 135K	0
256	335114 — 335K	0
512	932362 — 932K	0

(b) Rete B

Size Hidden Layer	Parametri addestrabili	Parametri NON addestrabili
8	314 — 0.3K	6272 — 6.3K
16	1002 — 1K	12544 — 12.5K
32	3530 — 3.5K	25088 — 25K
64	13194 — 13K	50176 — 50K
128	50954 — 51K	100352 — 100K
256	200202 — 200K	200704 — 201K
512	793610 — 794K	401408 — 401K

(c) Rete C

Size Hidden Layer	Parametri addestrabili	Parametri NON addestrabili
8	6586 — 6.6K	0
16	13546 — 13.5K	0
32	28618 — 29K	0
64	63370 — 63K	0
128	151306 — 151K	0
256	400906 — 401K	0
512	1195018 — 1.2M	0

Tabella 3.2. Confronto sintetico tra il numero di parametri di Rete A e Rete B

Layers	Percentuale di parametri addestrabili su totale (Rete B)	Differenza di parametri addestrabili (Rete A - Rete B)
8	4.77%	6200
16	7.40%	12272
32	12.33%	24032
64	20.83%	46016
128	33.67%	83840
256	49.95%	134912
512	66.44%	138752
1024	79.74%	-246784

L'ultima riga della Tabella 3.1 è solamente teorica in quanto non è stato addestrato alcun modello con hidden layers di grandezza 1024 in *Exp3*. Risulta interessante poichè in questa progressione di n in potenze di 2 è la prima volta in cui Rete B può avere più parametri addestrabili di A.

3.2.2 *Exp4* - Estensione a reti più ampie e verifica dell'ipotesi sui parametri addestrabili

L'esperimento *Exp4* è un'espansione del precedente che mira a incorporare anche reti con hidden layers da 1024 neuroni. Il resto della parametrizzazione resta identica a *Exp3*.

Le precedenti figure (Fig 3.5) sono le corrispettive a quelle presentate per l'esperimento con 20 epoche, ovvero hanno solo una linea aggiuntiva per indicare le reti con hidden layers da 1024 neuroni. In questa sezione presentiamo altri grafici (Fig. 3.6), più atti a sottolineare le differenze tra le tre reti in termini di Accuracy.

Rete A e Rete B esibiscono un comportamento simile per $n \geq 512$, man mano che la sample size aumenta, il che incoraggia l'ipotesi avanzata precedentemente: non è il numero totale di parametri quello contemplato in $O(W \log W)$ ma il numero di parametri addestrabili $W_{\text{add}}^{(B)}$.

Invece Rete B e Rete C, nonostante presentino entrambi per n piccolo livelli di accuracy bassi, non seguono andamenti simili. Vi sono 2 possibilità, non mutualmente esclusive:

- L'esiguo numero di epoche causa questo comportamento;
- Rete C presenta una bassa accuracy a causa del numero ridotto di sample disponibili rispetto alla sua sample complexity e delle poche epoche.

L'interesse nell'aumentare il numero di epoche è anche legato alla necessità, stando alla teoria, di un algoritmo ERM: all'algoritmo di addestramento 20 epoche potrebbero non essere sufficienti per minimizzare o almeno diminuire sufficientemente il rischio empirico.

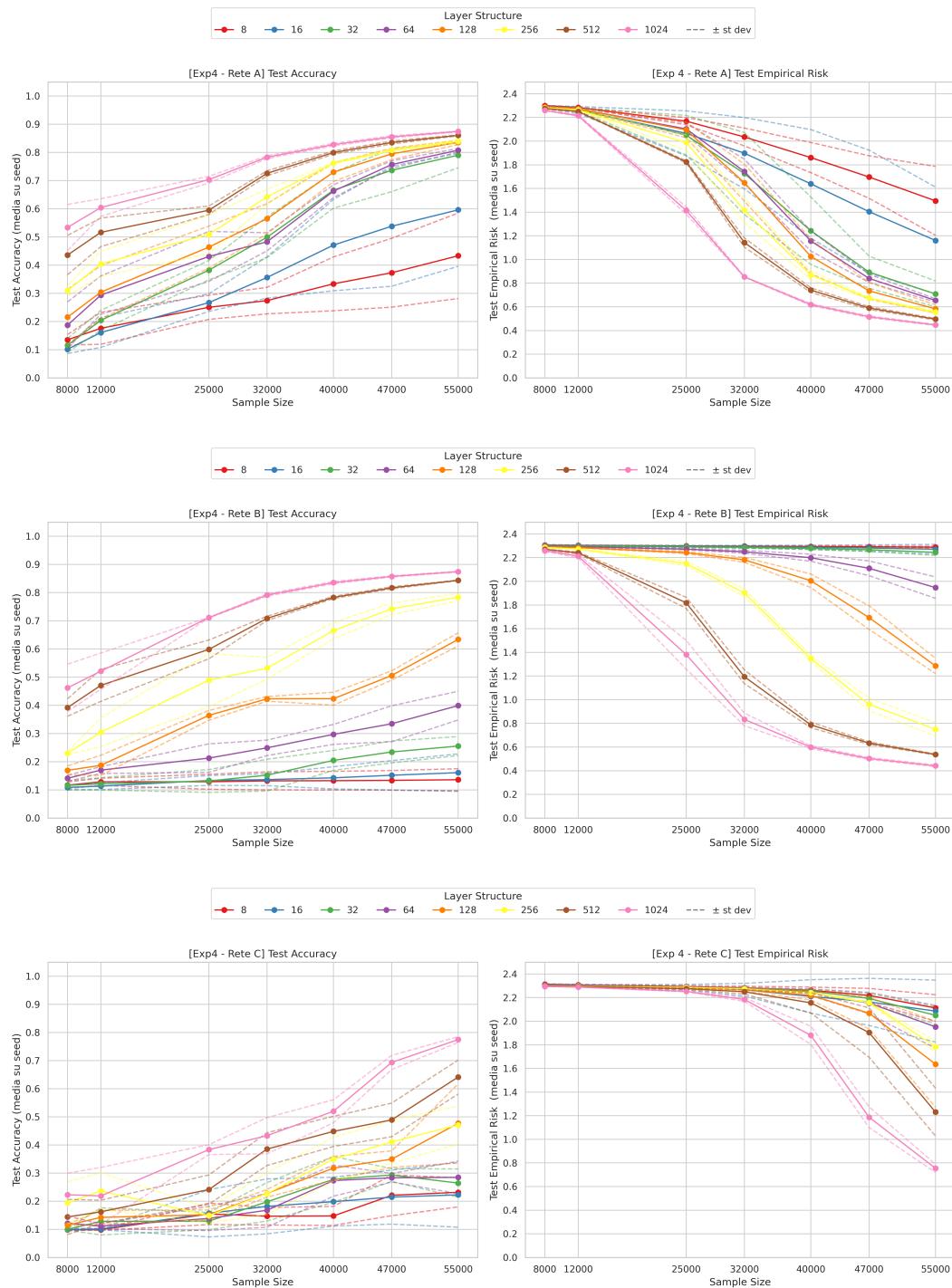


Figura 3.5. *Exp4 - Reti A, B e C*
Accuracy su Sample size & Empirical Risk su Sample size

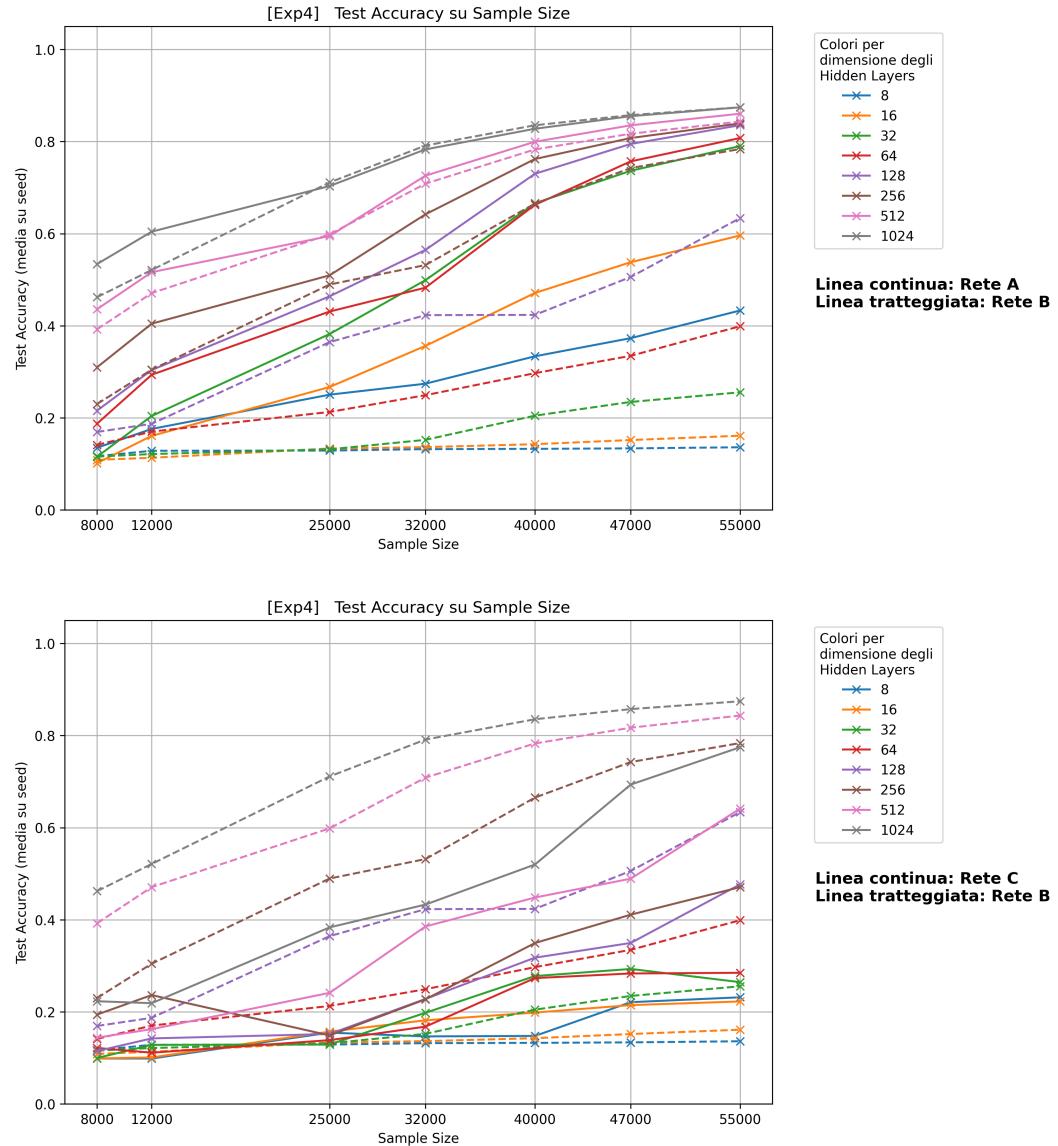


Figura 3.6. *Exp4* - Accuracy su Sample size comparata tra Rete A, B, C

3.2.3 *Exp5* - Ruolo del tempo di addestramento: early stopping, instabilità e fenomeni non monotoni

In questo esperimento è stato adottato lo stesso setup utilizzato in *Exp4*, con l'introduzione di un criterio di early stopping basato sulla validation accuracy, considerando una variazione minima di 0.001 e una patience pari a 5. Come failsafe è stato impostato il numero massimo di epoche a 1000, anche se non è mai raggiunto. Nonostante questa configurazione, la Rete C non riesce sempre a superare le 20 epoche; per affrontare questa limitazione, è stata sviluppata una variante, denominata C[LowerDelta], in cui l'early stopping è sensibile a variazioni nella validation accuracy molto più piccole, pari a 0.0001, al fine di permettere un addestramento prolungato.

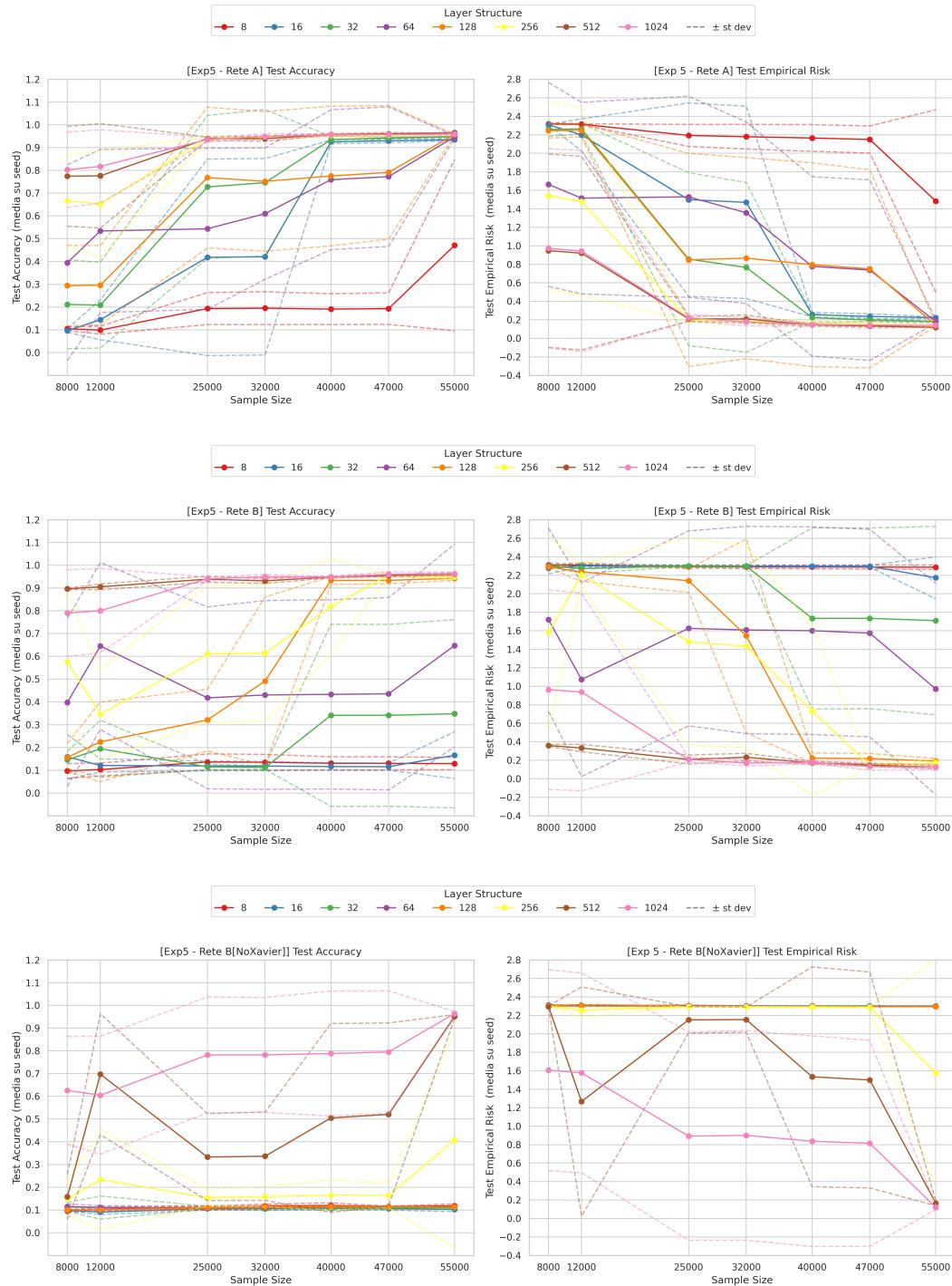


Figura 3.7. Exp5 - Reti A, B e B[NoXavier]
Accuracy su Sample size & Empirical Risk su Sample size

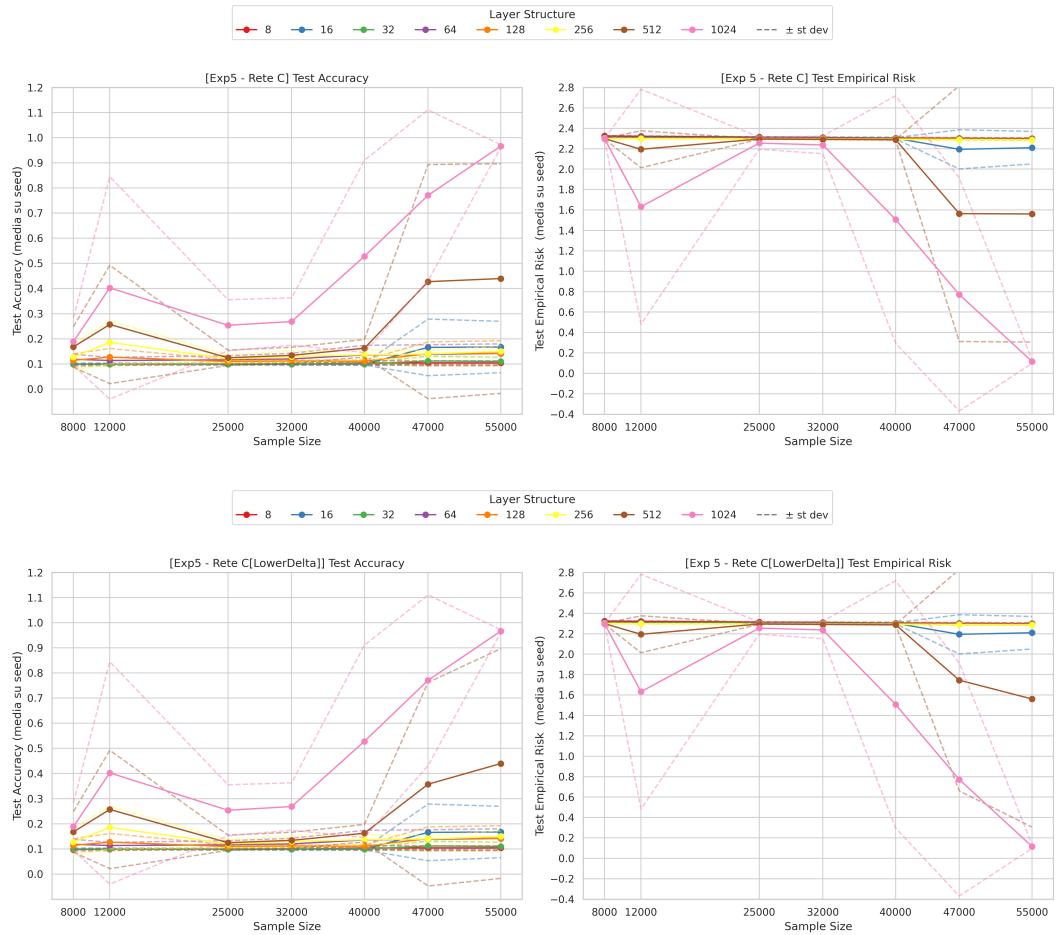


Figura 3.8. *Exp5 - Reti C e C[LowerDelta]*
Accuracy su Sample size & Empirical Risk su Sample size

Effetti dell'inizializzazione

Parallelamente, è stato di interesse verificare se diverse inizializzazioni dei pesi non addestrabili possano contribuire alle prestazioni del modello o se, come suggeriscono le osservazioni precedenti, esercitino un effetto piuttosto marginale. A tale scopo, è stata addestrata una seconda versione della Rete B, denominata B[NoXavier], nella quale l'inizializzazione è quella predefinita di PyTorch. I valori osservati mostrano una deviazione standard piuttosto elevata, rendendo i grafici più difficili da leggere e interpretare. Per cercare di comprendere meglio questa variabilità, è possibile analizzare i dati senza mediare sui diversi seed, in modo da verificare se la dispersione rimane significativa anche considerando ciascun seed singolarmente. Tutti i grafici generabili con questo approccio sono disponibili su GitHub; in questa sezione, per brevità, vengono presentati solamente i grafici relativi all'Accuracy della rete C[LowerDelta] (Fig. 3.9).

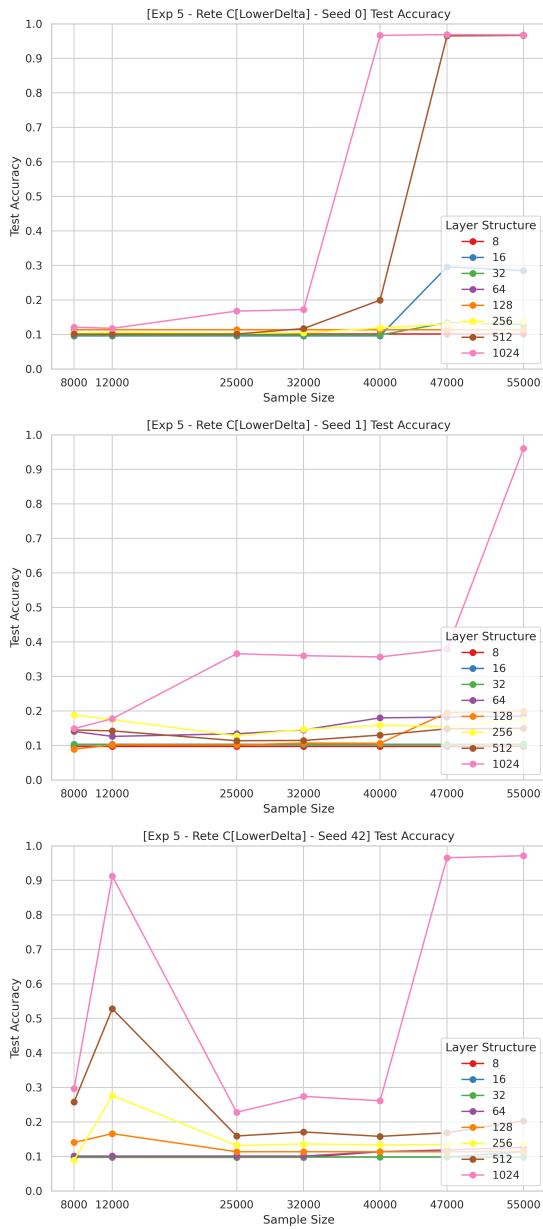


Figura 3.9. *Exp5 - Rete C[LowerDelta]*
Accuracy su Sample size, seed by seed

un confronto con i risultati di *Exp4* (v. 3.5).

L'impatto più significativo riguarda tuttavia le architetture più ampie, che — in questo contesto sperimentale — presentano sì un'elevata capacità di generalizzazione ma, proprio poiché ampie, tendono ad attivare l'early stopping molto prima rispetto a reti più piccole. Tale fenomeno era già intuibile confrontando le curve di accuracy della Rete C tra *Exp4* e *Exp5*: nel primo caso il modello era vincolato a un minimo di 20 epoche, mentre nel secondo poteva interrompere l'addestramento anticipatamente. Per rendere esplicito questo effetto, nelle figure seguenti vengono riportate alcune proiezioni 3D in cui il terzo asse rappresenta il numero effettivo di epoche su cui i

Isolando l'analisi su seed diversi possiamo apprezzare meglio l'effetto della sample complexity. I seed 0 e 1 hanno un comportamento progressivo nelle architetture con layer più ampi. Invece per il seed 42 la situazione è chiaramente non monotona e molte delle fluttuazioni nel grafico mediato sono da attribuire a ciò. In tutti e tre gli scenari si riscontra la stessa struttura nell'accuracy della Rete C dell'esperimento precedente *Exp4* (Fig. 3.5).

Nonostante questa operazione abbia effettivamente ridotto parte del rumore nei grafici, essa non è sufficiente per comprendere in modo chiaro la dinamica osservata nel seed 42, in particolare l'anomalia presente a sample size = 12.000 e poi il netto crollo registrato a sample size = 25.000.

Dopo aver esplorato diverse ipotesi esplicative, è emersa una conclusione coerente anche con la teoria della sample complexity: le prestazioni mostrano una marcata dipendenza dal numero di epoche di addestramento, ovvero da quanto a fondo permettiamo all'algoritmo ERM di cercare il miglior "candidato" nella classe (v. Lemma 2.1). Il ruolo delle epoche e del criterio di early stopping è già visibile nella Rete A: in *Exp5* l'accuratezza supera il 90%, risultato non raggiungibile senza un numero maggiore di epoche disponibili come riscontrabile con

modelli sono stati allenati. Sono riportate solo delle istanze esemplificative. Una collezione più ampia di plot 3D è esplorabile interattivamente tramite la repository GitHub associata a questa tesi.

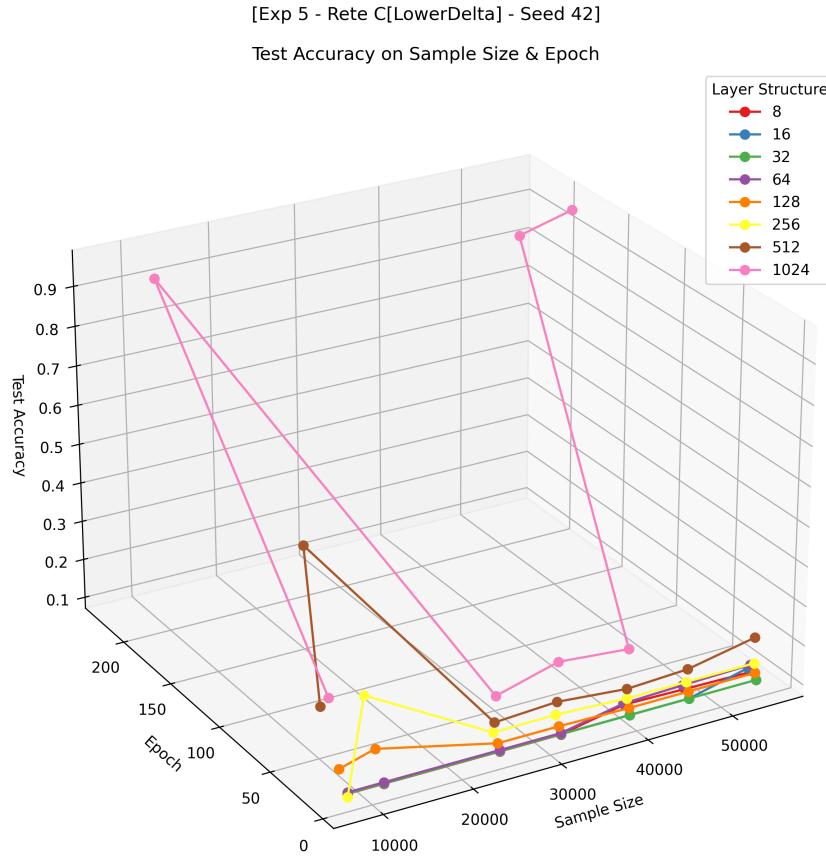


Figura 3.10. *Exp5 - Rete C[LowerDelta] - Seed 42*

Aggiungendo una terza dimensione il contributo del tempo di addestramento diviene palese. Ad esempio tramite Fig. 3.10 possiamo motivare il comportamento anomalo su sample size = 12.000 e sample size = 25.000. Sebbene questa intuizione "con un maggior numero di epoche migliora la perfomance" non sia nuova in Machine Learning e sebbene non sia il principale oggetto di indagine di questo lavoro, costituisce la prima netta discrepanza empirica con la teoria della sample complexity tramite il già citato Lemma 2.1. Infatti il comportamento monotono osservabile in *Exp3*, concorde con il bound $O(W \log W)$, è disturbato dal numero di epoche o, almeno, vi è una forte correlazione.

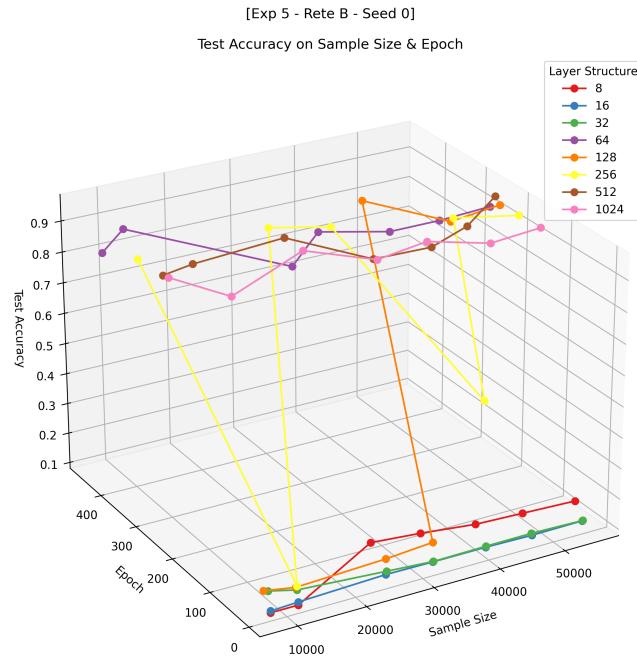


Figura 3.11. Exp5 - Rete B - Seed 0

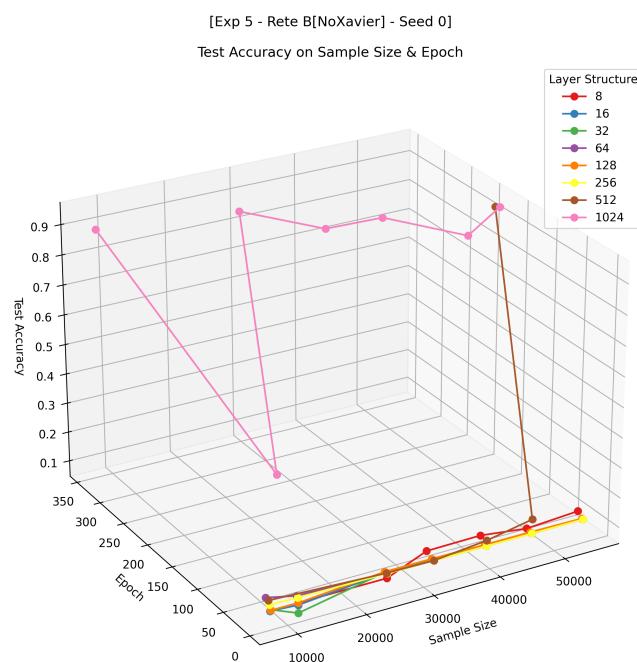


Figura 3.12. Exp5 - Rete B[NoXavier] - Seed 0

La visualizzazione tridimensionale è anche utile per comparare gli effetti di inizializzazioni diverse sulla Rete B. Dai plot, Fig. 3.11 e Fig. 3.12, si osserva una notevole differenza tra l'accuracy delle due reti dedurre una correlazione positiva tra l'inizializzazione Xavier e il numero di epoche. Però, per esaminare ulteriormente questo scenario e andare oltre una correlazione visiva sarebbero necessari ulteriori esperimenti, in particolare un confronto su un numero di epoche fisso, senza l'early stopping, ovvero ripetere *Exp4* per le due versioni della Rete B.

3.3 Studio del rango

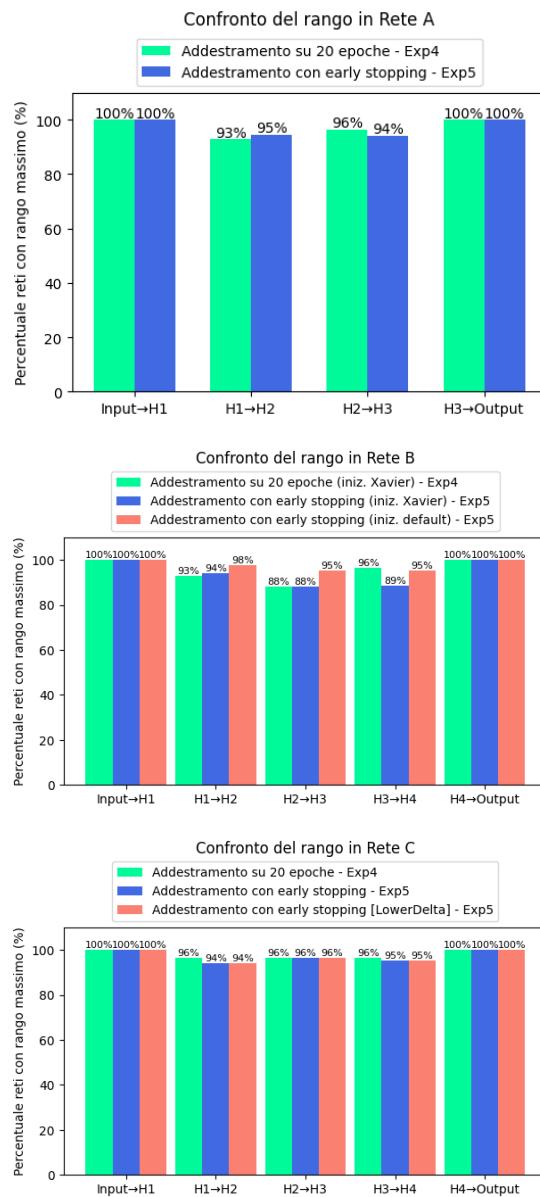


Figura 3.13. Percentuale di matrici dei pesi con rango massimo

Dopo l'addestramento, è stato calcolato il rango delle matrici dei pesi $\mathcal{W}^{(l)}$ associate ai layer di ciascuna rete. In particolare, per ciascun l con $\mathcal{W}^{(l)}$ di dimensione $m \times n$, il rango è stato determinato tramite metodo SVD con l'utilità `numpy.linalg.matrix_rank` di Numpy.

Il rango rappresenta il numero di colonne linearmente indipendenti. Dalla costruzione di $\mathcal{W}^{(l)}$ possiamo interpretare le colonne come i pesi di un "neurone di input". Con neurone di input si intende, in una FNN e considerando due layer adiacenti, i neuroni contenuti nel layer di sinistra. Il rango quindi controlla la dimensione della sottovarietà delle rappresentazioni interne accessibili al modello, in un certo senso la complessità delle funzioni replicabili. Se non massimo suggerisce un'eventuale contrazione dello spazio delle ipotesi \mathcal{H} e, di conseguenza, indica che il bound superiore sulla VC-dimension potrebbe risultare non sufficientemente restrittivo. I risultati sono sintetizzati nella figura 3.13, invece nella Repository Github vi sono i csv con i dati completi. Se non per rare eccezioni, queste matrici hanno rango pieno e quelle poche eccezioni sono molte vicine ad avere rango pieno.

Tali osservazioni risultano difficilmente comparabili con la letteratura esistente, in quanto la maggior parte degli studi si concentra su modelli low-rank o su tecniche con compressione del rango che preservano le prestazioni. Alcuni esempi includono Martin and Mahoney (2021) e Feng et al. (2022).

Si può formulare un'estensione naturale dell'esperimento: analizzare varianti della rete B in cui il primo hidden layer non solo è non addestrabile su diverse inizializzazioni ma ha anche una dimensione molto più grande dell'input. Ciò permetterebbe di osservare se, espandendo lo spazio di ricerca, i punti del dataset diventino sin da subito linearmente divisibili. Facendo ciò potremmo sperare di osservare il comportamento più tipico di weight decay e di regolarizzazione implicita nei layer successivi. Tale studio può anche essere condotto in fase di training piuttosto che alla fine per osservarne anche l'evoluzione.

Capitolo 4

Criticità del framework e alternative alla VC-Dimension

La VC-dimension, negli anni, si è rivelata troppo grossolana o pessimistica, soprattutto nel caso delle reti neurali profonde (v. Martin and Mahoney (2021)). Per questo motivo sono state sviluppate misure alternative, in grado di catturare aspetti più sottili, facendo ricorso a ipotesi diverse o aggiuntive. In questo capitolo presentiamo alcune di queste misure, con l'obiettivo di chiarire le intuizioni che le guidano e di collegarle alle criticità osservabili sia nel framework teorico sia negli esperimenti.

4.1 Misure con convergenza uniforme

Il teorema 2.2 di Vapnik-Chervonenkis stabilisce un'equivalenza tra VC-dimension finita e convergenza uniforme. Altre misure sviluppano questa equivalenza, di seguito ne sono presentate alcune con caratteristiche leggermente diverse.

4.1.1 Natarajan dimension

La Natarajan dimension è una generalizzazione della VC-dimension al caso multiclasse, introdotta in Natarajan (1989) per caratterizzare la PAC learnability e la convergenza uniforme in contesti in cui lo spazio delle etichette contiene più di due classi.

Osservazione sui limiti della VC-dimension 1.

È stato osservato in Daniely et al. (2015) che ridurre un task multiclasse a quello binario tramite One-VS-All o All Pairs può generare modelli potenzialmente peggiori. Solitamente questa cosa avviene perché più difficilmente l'algoritmo ERM riesce a minimizzare con successo. Ciò implica una capacità di generalizzazione più bassa e una sample complexity maggiore rispetto al necessario.

Definizione 4.1 (Natarajan Dimension).

Sia \mathcal{H} uno spazio di funzioni che assegnano a ciascun punto di input un'etichetta in un insieme finito \mathcal{Y} . Un insieme $S = \{x_1, \dots, x_d\} \subseteq \mathcal{X}$ si dice Natarajan-shattered da \mathcal{H} se esistono due funzioni

$$f, g : S \rightarrow \mathcal{Y}, \quad \text{con } f(x_i) \neq g(x_i) \quad \forall i,$$

tali che per ogni sottoinsieme $T \subseteq S$ esiste un'ipotesi $h_T \in \mathcal{H}$ con

$$h_T(x_i) = \begin{cases} f(x_i), & x_i \in T, \\ g(x_i), & x_i \notin T. \end{cases}$$

La Natarajan dimension di \mathcal{H} , denotata $\text{Ndim}(\mathcal{H})$, è la dimensione massima di un insieme Natarajan-shattered.

Osservazione sui limiti della VC-dimension 2.

Il teorema fondamentale dell'apprendimento statistico, nel caso binario, richiede che la loss sia 0-1. Questo consente di utilizzare la disegualanza di Hoeffding e, di conseguenza, ottenere il corollario sulla convergenza uniforme (Shalev-Shwartz and Ben-David, 2014, Lemma 4.5 e Corollario 4.6). Questo approccio, nel caso multiclass, non è sufficiente (Shalev-Shwartz and Ben-David, 2014, sezione 29.4).

Tuttavia, applicando il Natarajan Lemma (Shalev-Shwartz and Ben-David, 2014, Lemma 29.4), è possibile estendere il teorema fondamentale dell'apprendimento statistico al contesto multiclass. In particolare, per classi di ipotesi con dimensione di Natarajan finita, si ottiene il seguente risultato.

Teorema 4.1 (Teorema Fondamentale dell'Apprendimento Statistico Multiclass – Vers. Quantitativa - Restrizione ad Agnostic PAC Learnability).

Esistono costanti assolute $C_1, C_2 > 0$ tali che quanto segue vale. Sia \mathcal{H} una classe di ipotesi di funzioni da \mathcal{X} a $[k]$, con dimensione di Natarajan $\text{Ndim}(\mathcal{H}) = d$.

Allora \mathcal{H} è agnostic PAC learnable con sample complexity

$$\frac{C_1(d + \log(1/\delta))}{\epsilon^2} \leq m_{\mathcal{H}}(\epsilon, \delta) \leq \frac{C_2(d \log(k) + \log(1/\delta))}{\epsilon^2}.$$

4.1.2 Rademacher complexity

La complessità di Rademacher, introdotta in Bartlett and Mendelson (2002), fornisce una misura data-dependent della capacità di una classe di ipotesi.

Definizione 4.2 (Rademacher Complexity Empirica).

Data una classe di modelli \mathcal{H} e un training set S , la complessità di Rademacher empirica $\hat{\mathcal{R}}_S(\mathcal{H})$ è definita come:

$$\hat{\mathcal{R}}_S(\mathcal{H}) = \mathbb{E}_{\sigma} \left[\sup_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^m \sigma_i h(x_i) \right]$$

dove σ_i sono variabili di Rademacher indipendenti, cioè

$$P(\sigma_i = +1) = P(\sigma_i = -1) = \frac{1}{2}$$

La complessità di Rademacher misura quanto bene la classe di modelli \mathcal{H} può adattarsi a rumore casuale. Intuitivamente, se la classe è molto flessibile, sarà in grado di correlare anche con etichette completamente casuali, risultando in una Rademacher Complexity elevata; una classe più semplice mostrerà invece una bassa correlazione con il rumore.

Un aspetto pratico importante è che $\hat{\mathcal{R}}_S(\mathcal{H})$ è difficile da calcolare esattamente per classi di ipotesi complesse, come le reti neurali profonde, poiché richiede di massimizzare la somma pesata su tutte le funzioni possibili della classe. Tuttavia, rimane uno strumento teorico potente per stimare lo scarto tra rischio empirico e rischio, grazie all'esistenza di bound. Questa misura si estende naturalmente anche a spazi di output continui, permettendo l'analisi di problemi di regressione senza dover introdurre discretizzazioni artificiali. Il bound in questione e l'estensione a regressione sono presenti sempre in questo lavoro Bartlett and Mendelson (2002).

Osservazione sui limiti della VC-dimension 3.

Introdurre la VC-dimension comporta il trattamento della regressione come un task di classificazione multiclasse. Sebbene non proibitivo, varie fonti come Stewart et al. (2023) e Ben-David et al. (1992) osservano che classificazione e regressione utilizzano loss differenti e caratteristiche diverse delle variabili (categorical vs continue/ordinate), il che può portare a risultati differenti a seconda dell'approccio scelto.

4.1.3 Fat-Shattering Dimension

La fat-shattering dimension Bartlett et al. (1996) è l'estensione della VC-dimension al caso in cui la classe delle ipotesi \mathcal{H} sia composta da funzioni a valori reali e introduce la nozione di separazione con margine.

Definizione 4.3 (Fat-Shattering Dimension). *Sia \mathcal{H} una classe di modelli e sia $\gamma > 0$. Un insieme $\{x_1, \dots, x_d\} \subseteq \mathcal{X}$ è γ -shattered da \mathcal{H} se esistono soglie $r_1, \dots, r_d \in \mathbb{R}$ tali che, per ogni vettore di segni $b \in \{-1, +1\}^d$, esiste un'ipotesi $h_b \in \mathcal{H}$ per cui:*

$$h_b(x_i) \begin{cases} \geq r_i + \gamma & \text{se } b_i = +1, \\ \leq r_i - \gamma & \text{se } b_i = -1. \end{cases}$$

La fat-shattering dimension di \mathcal{H} è definita come:

$$\text{fat}_\gamma(\mathcal{H}) = \max \{d : \exists \{x_1, \dots, x_d\} \text{ } \gamma\text{-shattered da } \mathcal{H}\}.$$

La dipendenza dal margine γ è cruciale: al suo diminuire di γ $\text{fat}_\gamma(\mathcal{H})$ tende ad aumentare. Nel limite $\gamma \rightarrow 0$, su funzioni binarie, si recupera un comportamento analogo alla $\text{VCdim}(\mathcal{H})$. Per questo motivo la fat-shattering dimension risulta utile per l'analisi di modelli basati sul margine, come le Support Vector Machines, e per classi di funzioni continue come quelle implementate dalle reti neurali.

4.2 Altre tipologie di misure

Esistono approcci alternativi che abbandonano la convergenza uniforme o introducono dipendenze esplicite dall'algoritmo di apprendimento utilizzato.

Queste misure sono spesso definite *algorithm-dependent*, poiché non caratterizzano solo la classe \mathcal{H} , ma anche il processo di selezione dell'ipotesi. Tale dipendenza può apparire come una limitazione rispetto al carattere universale della VC-dimension ma consente di ottenere bound più stretti e realistici.

4.2.1 PAC-Bayesian Theory

L'approccio PAC-Bayes, introdotto da McAllester (1998), abbandona l'idea della convergenza uniforme e considera distribuzioni sulle ipotesi, utilizzando classicamente una prior come leva. In questo quadro, non si stima una singola ipotesi h ma una distribuzione su \mathcal{H} che assegna pesi in funzione alla 'plausibilità' delle ipotesi.

Osservazione sui limiti della VC-dimension 4.

La VC-dimension, come osservato a fine sezione 2.3, tramite la definizione di PAC Learning, dipende solo da \mathcal{H} : è distribution-free. Ciò cattura la nostra scarsa conoscenza a priori sulla distribuzione, ma tende a sovrastimare la sample complexity. Nella pratica, questa sovrastima emerge probabilmente perché, anche se non conosciamo la struttura della distribuzione, esiste una certa regolarità che la definizione non cattura Anthony and Bartlett (1999); Bishop (2006). O meglio, visto che il bound deve valere su ogni distribuzione dovrà valere anche sulla distribuzione peggiore e non sempre nelle applicazioni pratiche è questo il caso.

Teorema 4.2 (PAC-Bayes Bound versione da Shalev-Shwartz and Ben-David (2014)). *Sia \mathcal{D} una distribuzione arbitraria sul dominio degli esempi \mathcal{Z} . Sia \mathcal{H} una classe di ipotesi e sia $\ell : \mathcal{H} \times \mathcal{Z} \rightarrow [0, 1]$ una funzione di loss. Sia P una distribuzione a priori su \mathcal{H} e sia $\delta \in (0, 1)$. Allora, con probabilità almeno $1 - \delta$ rispetto alla scelta di un training set i.i.d. S campionato secondo \mathcal{D} , per ogni distribuzione Q su \mathcal{H} (anche dipendente da S):*

$$\mathbb{E}_{h \sim Q}[L_{\mathcal{D}}(h)] \leq \mathbb{E}_{h \sim Q}[L_S(h)] + \sqrt{\frac{KL(Q||P) + \log(n/\delta)}{2n}},$$

dove $KL(Q||P)$ è la divergenza di Kullback–Leibler.

Il termine $KL(Q||P)$ quantifica quanto la distribuzione a posteriori Q si discosti dal prior P , penalizzando modelli che richiedono molta informazione per essere giustificati. Questo framework è particolarmente utile per ensemble methods e per modelli con pesi stocastici, nonché in scenari di online learning.

4.2.2 Stabilità algoritmica

La stabilità algoritmica, da Bousquet and Elisseeff (2002), rappresenta un approccio alternativo alla generalizzazione: si studia la complessità della classe \mathcal{H} tramite variazioni dell'output di un algoritmo A quando un singolo esempio del training set viene modificato.

Osservazione sui limiti della VC-dimension 5.

Nel Lemma 2.2.1 abbiamo stabilito una relazione tra $L_S(h)$ e $L_{\mathcal{D}}(h)$ tramite l'algoritmo ERM. Se non è possibile risolvere precisamente il problema di minimizzazione in questione, si introduce ulteriore errore indipendentemente dalla rappresentatività del sample. Algoritmi come SGD o Adam, comunemente usati per addestrare reti neurali, non garantiscono di trovare il minimo globale. In questo caso la VC-dimension sottostima il fabbisogno del modello, semplicemente perché non troviamo il modello h^ adatto.*

Esistono diverse definizioni di stabilità, proprio come la convergenza, di seguito ne è riportata una tra le più usate.

Definizione 4.4 (Stabilità uniforme).

Un algoritmo di apprendimento $A : \mathcal{Z} \rightarrow \mathcal{H}$ è β -uniformemente stabile se, per ogni coppia di training set S, S' che differiscono in un solo esempio, vale:

$$\sup_{z \in \mathcal{Z}} |\ell(A(S), z) - \ell(A(S'), z)| \leq \beta.$$

L'idea è che, se l'algoritmo è stabile rispetto a perturbazioni del training set, allora non può fare eccessivamente overfitting sui dati. Ciò conduce a un bound di generalizzazione, sempre da Bousquet and Elisseeff (2002).

Teorema 4.3 (Generalizzazione tramite Stabilità uniforme).

Se A è β -uniformemente stabile, allora con probabilità almeno $1 - \delta$:

$$L_{\mathcal{D}}(A(S)) \leq L_S(A(S)) + 2\beta + O\left(\sqrt{\frac{\log(1/\delta)}{n}}\right).$$

Questo approccio è distribution-free, non ci sono ipotesi su \mathcal{D} quindi vale per ogni distribuzione, ma dipende dall'algoritmo. Ciò lo rende adatto ad analizzare algoritmi regolarizzati (ridge regression, LASSO), varianti di Stochastic Gradient Descent e algoritmi di boosting.

4.2.3 Information-Theoretic Generalization Bounds

Gli information-theoretic bounds analizzano la generalizzazione tramite la quantità di informazione che l'algoritmo estrae dal training set, basandosi sulla mutua informazione tra S e l'ipotesi $h_S = A(S)$. L'idea, dovuta a Russo and Zou (2016), è che valori elevati della mutua informazione $I(S; h_S)$ indicano memorizzazione dei dati, mentre valori ridotti favoriscono la generalizzazione.

Teorema 4.4 (Bound bias via mutual information). *Sia $\phi = (\phi_1, \dots, \phi_m) : \Omega \rightarrow \mathbb{R}^m$ un vettore di variabili casuali definite su uno spazio di probabilità comune, e sia $T : \Omega \rightarrow \{1, \dots, m\}$ che determina la scelta di ϕ_T . Sia $\mu = (\mu_1, \dots, \mu_m)$ il vettore dei valori attesi, cioè $\mu_i = \mathbb{E}[\phi_i]$. Supponiamo che, per ogni $i \in \{1, \dots, m\}$, la variabile $\phi_i - \mu_i$ sia σ -subGaussian.*

Allora vale:

$$|\mathbb{E}[\phi_T] - \mathbb{E}[\mu_T]| \leq \sigma \sqrt{2 I(T; \phi)},$$

dove $I(T; \phi)$ denota l'informazione mutua tra T e ϕ .

Questo mostra che la generalizzazione dipende dalla quantità di informazione estratta dall'algoritmo, non dalla complessità della classe \mathcal{H} . Ciò risulta particolarmente rilevante per modelli sovrapparametrizzati come deep neural networks, per i quali le misure classiche risultano spesso troppo pessimistiche.

Capitolo 5

Riepilogo e riflessioni finali

Questo lavoro ha esplorato empiricamente la sample complexity delle reti neurali feedforward, confrontando le predizioni teoriche derivate dalla VC-dimension con il comportamento osservato su tre architetture di complessità crescente addestrate sul dataset MNIST. È importante sottolineare che le osservazioni presentate in questo lavoro si basano su correlazioni empiriche e non implicano necessariamente relazioni causali dirette. Ad esempio, la correlazione tra il numero di epoch e l'accuracy non dimostra che l'early stopping sia l'unica causa del comportamento osservato; altri fattori, come la geometria del loss landscape, la scelta dell'ottimizzatore, o la struttura del dataset, potrebbero contribuire in modo significativo. Similmente, l'apparente convergenza della Rete B alla Rete A per n crescente potrebbe essere influenzata da interazioni complesse tra inizializzazione, dimensione del layer, e dinamiche di ottimizzazione. Ulteriori esperimenti, inclusi test su dataset diversi, con differenti funzioni di attivazione e regimi di regolarizzazione, sarebbero necessari per validare in modo robusto le ipotesi avanzate.

Discrepanza tra teoria e pratica. I risultati sperimentali evidenziano una discrepanza significativa tra il bound teorico $O(W \log W)$ e i requisiti pratici di dati. Le reti analizzate raggiungono livelli di accuracy superiori all'80% con quantità di dati molto inferiori a quelle suggerite dal bound, anche in presenza di un numero di parametri che supera la dimensione dell'intero dataset. Questa osservazione conferma che il bound, pur costituendo una garanzia teorica rigorosa, risulta eccessivamente conservativo nella pratica.

Ruolo dei parametri addestrabili. L'analisi della Rete B, caratterizzata da un primo layer con pesi non addestrabili, suggerisce che la sample complexity dipenda principalmente dal numero di parametri effettivamente ottimizzati dall'algoritmo di apprendimento, piuttosto che dal numero totale di parametri della rete. Questa ipotesi è supportata dal comportamento osservato: per $n < 784$, la Rete B mostra performance simili alla Rete A, nonostante abbia un numero totale di parametri maggiore. Tale risultato indica che la caratterizzazione della complessità tramite il solo conteggio dei parametri potrebbe non essere sufficiente, e che la struttura dell'ottimizzazione gioca un ruolo fondamentale.

Tempo di addestramento e convergenza. Un'altra osservazione cruciale riguarda l'influenza del numero di epoche di addestramento. Negli esperimenti con un numero fisso di 20 epoche, si osservano andamenti relativamente monotoni e coerenti con le aspettative teoriche. Tuttavia l'introduzione dell'early stopping rivela dinamiche più complesse: reti con maggiore capacità tendono a interrompere l'addestramento anticipatamente, compromettendo le performance nonostante la loro capacità di generalizzazione potenzialmente superiore. Questo fenomeno sottolinea come il framework PAC, basato sull'algoritmo ERM, non catturi adeguatamente gli effetti pratici legati alla convergenza dell'ottimizzatore. Come evidenziato dal Lemma 2.1, la qualità dell'output dipende dalla capacità dell'algoritmo di minimizzare il rischio empirico, aspetto che la sola VC-dimension non modella.

Studio del rango. L'analisi del rango delle matrici dei pesi (Sezione 3.3) ha rivelato che, nella quasi totalità dei casi, le matrici hanno rango pieno o molto prossimo al rango massimo. Questo risultato suggerisce che le reti addestrate non subiscono una contrazione significativa dello spazio delle ipotesi, contrariamente a quanto ci si potrebbe aspettare.

Necessità di framework alternativi. Come discusso nel Capitolo 4, esistono diverse alternative alla VC-dimension che affrontano alcune delle sue limitazioni. Tuttavia, nessuna di queste misure è ancora pienamente soddisfacente per le reti neurali profonde: molte risultano difficili da calcolare, altre forniscono bound ancora troppo grandi, altre ancora richiedono ipotesi forti sulla struttura del problema.

Prospettive future.

- Estensione degli esperimenti:
analizzare reti con primo hidden layer espanso e non addestrabile, per verificare se uno spazio di ricerca più ampio faciliti la separabilità lineare dei dati e attivi meccanismi di regolarizzazione implicita nei layer successivi.
- Studio dinamico del rango:
monitorare l'evoluzione del rango delle matrici dei pesi durante l'addestramento, per comprendere se e come la struttura delle rappresentazioni interne cambia nel tempo.
- Confronto tra inizializzazioni:
condurre esperimenti sistematici su diverse strategie di inizializzazione dei pesi non addestrabili, controllando per il numero di epoche e la dimensione del dataset.

La sample complexity delle reti neurali feedforward resta un problema aperto. Questo lavoro ha mostrato che il framework classico basato su VC-dimension non cattura adeguatamente le dinamiche di apprendimento osservate empiricamente. Comprendere queste dinamiche è fondamentale non solo per migliorare la teoria dell'apprendimento statistico, ma anche per guidare la progettazione di architetture più efficienti, capaci di generalizzare bene anche in presenza di dati limitati.

Bibliografia

- Anthony, M. and Bartlett, P. (1999). *Neural Network Learning: Theoretical Foundations*. Cambridge University Press.
- Bartlett, P. L., Harvey, N., Liaw, C., and Mehrabian, A. (2019). Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63):1–17.
- Bartlett, P. L., Long, P. M., and Williamson, R. C. (1996). Fat-shattering and the learnability of real-valued functions. *J. Comput. Syst. Sci.*, 52(3):434–452.
- Bartlett, P. L. and Mendelson, S. (2002). Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482.
- Ben-David, S., Cesa-Bianchi, N., and Long, P. M. (1992). Characterizations of learnability for classes of 0, . . . , n-valued functions. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, page 333–340, New York, NY, USA. Association for Computing Machinery.
- Bishop, C. (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer.
- Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. (1989). Learnability and the vapnik-chervonenkis dimension. *J. ACM*, 36(4):929–965.
- Boucheron, S., Bousquet, O., and Lugosi, G. (2005). Theory of classification: A survey of some recent advances. *ESAIM: Probability and Statistics*, v.9, 323–375 (2005).
- Bousquet, O. and Elisseeff, A. (2002). Stability and generalization. *Journal of Machine Learning Research*, 2:499–526.
- Cesa-Bianchi, N. and Lugosi, G. (2006). *Prediction, Learning, and Games*. Cambridge University Press.
- Daniely, A., Sabato, S., Ben-David, S., and Shalev-Shwartz, S. (2015). Multiclass learnability and the erm principle. *Journal of Machine Learning Research*, 16:2377–2404.
- Feng, R., Zheng, K., Huang, Y., Zhao, D., Jordan, M., and Zha, Z.-J. (2022). Rank diminishing in deep neural networks. In *Proceedings of the 36th International*

- Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.
- Folland, G. (1999). *Real Analysis: Modern Techniques and Their Applications*. Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts. Wiley.
- Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. <http://www.deeplearningbook.org>.
- Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer series in statistics. Springer.
- Khadka, R., Jha, D., Hicks, S., Thambawita, V., Riegler, M. A., Ali, S., and Halvorsen, P. (2022). Meta-learning with implicit gradients in a few-shot setting for medical image segmentation. *Computers in Biology and Medicine*, 143:105227.
- Kotsiantis, S., Kanellopoulos, D., and Pintelas, P. (2006). Data preprocessing for supervised learning. *International Journal of Computer Science*, 1:111–117.
- LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- Martin, C. H. and Mahoney, M. W. (2021). Implicit self-regularization in deep neural networks: Evidence from random matrix theory and implications for learning. *Journal of Machine Learning Research*, 22(165):1–73.
- McAllester, D. A. (1998). Some pac-bayesian theorems. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 230–234.
- Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2018). *Foundations of Machine Learning*. The MIT Press, 2nd edition.
- Natarajan, B. K. (1989). On learning sets and functions. *Machine Learning*, 4(1):67–97.
- Russo, D. and Zou, J. (2016). Controlling bias in adaptive data analysis using information theory. In Gretton, A. and Robert, C. C., editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 1232–1240, Cadiz, Spain. PMLR.
- Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press.
- Stewart, L., Bach, F., Berthet, Q., and Vert, J.-P. (2023). Regression as classification: Influence of task formulation on neural network features. In Ruiz, F., Dy, J., and van de Meent, J.-W., editors, *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 11563–11582. PMLR.

- Vapnik, V. (1999). *The Nature of Statistical Learning Theory*. Information Science and Statistics. Springer New York.
- Vapnik, V. N. and Chervonenkis, A. Y. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280.
- Vovk, V., Papadopoulos, H., and Gammerman, A. (2015). *Measures of Complexity: Festschrift for Alexey Chervonenkis*. Computer Science. Springer International Publishing.
- Wolpert, D. (2001). The supervised learning no-free-lunch theorems. In *Proceedings of the 6th Online World Conference on Soft Computing in Industrial Applications*.
- Wolpert, D. and Macready, W. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82.