UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Corso di Laurea in Ingegneria Informatica

PREVISIONE DI STRUTTURE SECONDARIE DELL'RNA TRAMITE RETE NEURALE ATTRAVERSO UN ALGORITMO ADAM BASED

Relatore:
Prof. Loris Nanni

Laureando:
Federico Finotto

Anno Accademico 2022/2023

16 Marzo 2023

Sommario

Background La predizione della struttura secondaria dell'RNA rappresenta un'area di grande interesse nella biologia computazionale, poiché la struttura secondaria è spesso indicativa della funzione biologica dell'RNA. Gli algoritmi di predizione della struttura secondaria si basano su un modello energetico che cerca di minimizzare l'energia libera termodinamica della struttura. La presenza di pseudonodi rende il problema della previsione della struttura secondaria NP-completo e quindi intrattabile in tempi polinomiali. Al fine di superare questa limitazione, sono stati sviluppati metodi di apprendimento automatico basati sui modelli energetici, che cercano di apprendere nuovi set di parametri e di trovare la struttura ottimale utilizzando algoritmi di apprendimento. Il Deep Learning è una tecnica di apprendimento automatico che ha rivoluzionato la predizione della struttura secondaria dell'RNA. Grazie alla sua capacità di apprendere rappresentazioni complesse dai dati, il Deep Learning sta aprendo nuove strade per lo sviluppo e la ricerca dei meccanismi biologici.

Risultati Viene presentato un modello di rete neurale, chiamato EbutRna, che rappresenta mediante tensori l'input e l'output utilizzato per prevedere la struttura secondaria dell'RNA. La rete neurale è stata addestrata su una vasta gamma di dati RNA eterogenei e il metodo di post-processing ha dimostrato di essere efficace nella predizione della struttura secondaria sui dataset di test. Inoltre, abbiamo confrontato due diversi ottimizzatori e abbiamo scoperto che Adam Optimization Algorithm ha prodotto risultati migliori rispetto a Stochastic Gradient Descent. La rete neurale EbutRna può essere utilizzata per la predizione della struttura secondaria dell'RNA in diverse applicazioni biologiche.

Conclusioni Nello studio proposto viene dimostrato come l'utilizzo di una rete neurale artificiale consenta di prevedere con alta precisione la struttura di una sequenza di RNA senza l'utilizzo di un modello energetico. Questo metodo ha dimostrato di essere efficace sui dataset pubblici, con risultati superiori o conformi rispetto ad altri metodi di predizione della struttura secondaria dell'RNA.

Indice

1 Introduzione **1**

1.1 Il problema della predizione 1

1.2 Progetti correlati 2

1.2.1 Metodi algoritmici basati sui modelli energetici 2

1.2.2 Metodi di apprendimento basati sui modelli energetici 2

1.2.3 Metodi basati sul Deep Learning 3

1.2.4 Considerazioni 4

2 Materiali e Metodi **5**

2.1 Definizione del problema 5

2.2 Rappresentazione di input e output in Tensori 6

2.3 Modello di predizione 7

2.3.1 Livelli della Rete Neurale 8

2.3.2 Scelta dell'ottimizzatore 9

2.3.3 Funzione Loss 9

2.4 Post-processing 10

3 Risultati **11**

3.1 Metodo di valutazione 11

3.2 Sistema di valutazione 11

3.3 Datasets 12

3.4 Fase di Test e Debug dell'addestramento 12

3.4.1 Loss durante l'addestramento 13

3.4.2 F1 Score su singoli elementi del Test Set 16

3.4.3 F1 Score sull'intero Test Set 17

3.5 Parametri del Modello 18

3.6 Addestramento con l'intero Train Set 19

4 Conclusione **21**

4.1 Limiti e sviluppi futuri 22

Bibliografia **23**

Elenco delle figure

2.1 Esempio di Predizione 6
2.2 Schema della Rete Neurale 8

3.1 Comparazione della variazione della Loss durante l'addestramento su dataset bprna 13
3.2 Comparazione della variazione della Loss durante l'addestramento su dataset ArchiveII 14
3.3 Comparazione della variazione della Loss durante l'addestramento su dataset rnastralign 15
3.4 Confronto dell'F1 Score su 10 Pattern casuali del Test Set di bprna 16
3.5 Confronto dell'F1 Score su 10 Pattern casuali del Test Set di ArchiveII 16
3.6 Confronto dell'F1 Score su 10 Pattern casuali del Test Set di rnastralign 17
3.7 Variazione della Loss durante l'addestramento della Rete con Train Set bprna . 19
3.8 Variazione della Loss durante l'addestramento della Rete con Train Set bprna . 20

Elenco delle tabelle

3.1 Valori dell'F1 Score nei primi test eseguiti con 1500 pattern nel Train Set . . . 17
3.2 Valori di precisione, recall e F1 score per diverse reti Addestrate con bpRNA . 19
3.3 Valori di precisione, recall e F1 score per diverse reti Addestrate con ArchiveII 20

Capitolo 1

Introduzione

1.1 Il problema della predizione

L'RNA è una delle molecole fondamentali della vita. È altamente versatile, infatti ricopre diversi ruoli chiave nei processi cellulari essenziali quali espressione e regolazione dei geni, oltre a trasportare segnali cellulari e a servire come catalizzatore polifunzionale. È una molecola polimerica lineare costituita da una struttura di zucchero-fosfato a cui sono legate unità nucleotidiche elementari con basi:

- Adenina (A),
- Citosina (C),
- Guanina (G),
- Uracile (U).

Ciascun filamento di RNA può piegarsi su se stesso seguendo le regole di accoppiamento di basi di Watson-Crick, dando così origine a conformazioni tridimensionali dotate di una specifica funzione biologica. Per comprendere, ed eventualmente controllare, questo processo di formazione è importante essere in grado di prevedere come una data sequenza di nucleotidi (ovvero la struttura primaria) si piega su se stessa per creare una struttura secondaria e, dunque, una struttura terziaria tridimensionale.
Dato che la previsione della struttura terziaria finale è molto complessa, la ricerca si è concentrata nel passaggio intermedio, ovvero la formazione della struttura secondaria.
In casi semplici, le strutture secondarie dell'RNA mostrano una disposizione gerarchica pulita composta da blocchi di accoppiamenti di basi corrispondenti (segmenti a elica) separati da intervalli di basi non accoppiate (loop).Questi casi non prevedono quindi la presenza di pseudonodi, è dunque possibile trovare in modo efficiente una struttura secondaria ottimale per una data sequenza mediante un approccio algoritmico, mentre nei casi in cui sono ammessi pseudonodi [1] il problema diventa NP-completo, diventando così intrattabile in tempi polinomiali.

1.2 Progetti correlati

Nella maggior parte dei casi gli approcci per la previsione della struttura secondaria utilizzano una funzione di punteggio (score) e, solo successivamente, calcolano le strutture appropriate rispetto alla funzione definita. Queste funzioni sono spesso basate su un modello energetico, l'obiettivo è quello di determinare una struttura a energia libera minima (MFE) del modello dato o di campionare le strutture secondo una distribuzione di probabilità di Boltzmann, associando a ciascuna conformazione della molecola un'energia corrispondente.
Tra i progetti correlati possiamo annoverare i metodi algoritmici basati sui modelli energetici, i metodi di apprendimento basati sui modelli energetici e i metodi basati sul Deep Learning.

1.2.1 Metodi algoritmici basati sui modelli energetici

I metodi algoritmici basati sui modelli energetici consistono in metodi che prevedono la struttura secondaria minimizzando l'energia libera termodinamica seguendo un dato modello energetico. L'approccio sviluppato da Zuker [2, 3], uno dei precursori di questa metodologia, presenta un metodo algoritmico per trovare una struttura MFE attraverso l'aggregazione di elementi strutturali localmente ottimali (Nearest Neighbour) rispetto a un modello energetico proposto.
Successivamente, Turner [4, 5] ha presentato un modello energetico Nearest Neighbour più completo, diventando il punto di riferimento per molti dei metodi ispirati all'algoritmo di Zuker come UNAFold [6], RNAStructure [7] e Vienna RNAfold [8].
Secondo Lyngsø e Pedersen[1], la ricerca della struttura MFE di strutture pseudonodate risulta essere un problema NP-completo. Di conseguenza, risulta quindi impossibile utilizzare metodi puramente algoritmici per la previsione di tali strutture senza comprometterne l'efficenza.
Esistono tuttavia metodi che utilizzando euristiche per prevedere anche strutture pseudonodate, ad esempio IPknot [9] e ProbKnot [10]

1.2.2 Metodi di apprendimento basati sui modelli energetici

I metodi di apprendimento basati sui modelli energetici, come la struttura MFE calcolata con gli algoritmi di ripiegamento, non sono sempre in grado di fornire la struttura target desiderata [11]. Inoltre, la precisione dei modelli energetici può essere limitata dalla scarsa disponibilità di dati sperimentali utilizzati per calcolare i parametri termodinamici [11]. Al contrario, i metodi di apprendimento automatico, come ContraFold [12], cercano di apprendere nuovi set di parametri e di trovare la struttura ottimale, utilizzando algoritmi di apprendimento invece di algoritmi di programmazione dinamica. In questo modo, i metodi di apprendimento automatico sono in grado di superare le limitazioni dei modelli energetici e di fornire una maggiore precisione nella previsione della struttura secondaria dell'RNA.

1.2.3 Metodi basati sul Deep Learning

I metodi basati sul Deep Learning utilizzano più strati per apprendere in modo autonomo da un set di dati le rappresentazioni di dati complessi come immagini, suoni o testi. Nelle banche dati si possono trovare vari esempi di metodi con questo approccio tra cui: CDPFold [13], E2Efold [14], SPOT-RNA [15], MXFold2 [16].
Nello specifico, CDPFold [13] utilizza una rete neurale convoluzionale per prevedere una score matrix. Da essa viene successivamente estratta la struttura dot-bracket, ovvero una notazione in cui le parentesi rappresentano una base "(" accoppiata con un'altra base ")", mentre i punti rappresentano le basi non accoppiate. Questa rappresentazione impedisce di rappresentare strutture che presentano pseudonodi. Inoltre, il post-processing utilizzato è inefficiente in termini di tempo per sequenze più lunghe di qualche centinaio di basi.
Il modello E2Efold [14] propone l'utilizzo di una rete Deep Learning basata su transformers [17] e strati convoluzionali per generare punteggi per tutte le possibili coppie in una sequenza di RNA. Inoltre, il modello utilizza una rete post-processing differenziale per convertire questi punteggi in una struttura secondaria. I transformer sono caratterizzati da strati che utilizzano un meccanismo di allert per dare maggiore importanza a determinati token all'interno della sequenza.
SPOT-RNA [15] è un modello di Deep Learning basato su strati convoluzionali e strati 2D-BiLSTM personalizzati (Bidirectional Long Short-Term Memory). Questo modello considera anche triplette (basi connesse ad altre due) e accoppiamenti non canonici. Tuttavia, è limitato a sequenze più brevi di 500 nucleotidi a causa della complessità del modello e dei limiti di memoria.
MXFold2 [16] è il modello più recente che contiene convoluzioni unidimensionali, bidimensionali e strati ricorrenti BiLSTM. L'output ha quattro punteggi diversi per ogni coppia, questi rappresentano: impilamento dell'elica, apertura dell'elica, chiusura dell'elica e regione non accoppiata.

1.2.4 Considerazioni

Negli ultimi anni, l'uso del Deep Learning in biologia computazionale è diventato sempre più diffuso, soprattutto per la previsione di strutture proteiche e di RNA. Ad esempio, AlphaFold [18], un algoritmo sviluppato dal team DeepMind di Google, utilizza diverse componenti di Deep Learning, come gli autoencoder variazionali, i meccanismi di attenzione (transformers) e le reti convoluzionali, per migliorare la precisione nella previsione delle strutture delle proteine. Il metodo di apprendimento per rinforzo utilizzato da Eastman [19] per la progettazione della sequenza di RNA è in grado di apprendere dalle esperienze passate per migliorare le prestazioni.
Inoltre, EternaBrain [20], un progetto di crowdsourcing per la progettazione di RNA, utilizza una rete neurale convoluzionale per prevedere la struttura secondaria dell'RNA a partire dalle sequenze. Grazie all'uso del Deep Learning, questi metodi hanno raggiunto livelli di precisione impensabili solo pochi anni fa, rivoluzionando il modo in cui la biologia computazionale affronta la previsione di strutture molecolari complesse.

Capitolo 2

Materiali e Metodi

2.1 Definizione del problema

Possiamo formulare come segue il problema della predizione della struttura secondaria di una sequenza RNA.
Data una sequenza di basi $b = (b_1, b_2, ..., b_L)$, dove ogni base b_i può assumere uno dei quattro valori A, U, C, G, il nostro obbiettivo consiste nel prevedere un insieme di accoppiamenti (qi, qj) che definiscono la struttura secondaria. Ad esempio, data la sequenza
UGAACCUGGUCAGGGCCGGAAGGCAGCAGCCA
è necessario prevedere gli accoppiamenti
(7,32), (8, 31), (9, 30), (10, 29), (15, 24), (16, 23), (17, 22)
Inoltre ci sono una serie di vincoli che devono essere soddisfatti:

- Sono consentiti gli accoppiamenti di Watson-Crick e i corrispondenti appaiamenti instabili:
  - (A, U),
  - (U, A),
  - (U, G),
  - (G, U),
  - (G, C),
  - (C, G)
- Ogni base può essere accoppiata con una sola altra base o non accoppiata. Se la base i è accoppiata con la base j, la base j è accoppiata con la base i.
- La distanza minima per l'accoppiamento è 3, quindi $|i - j| \geq 3$

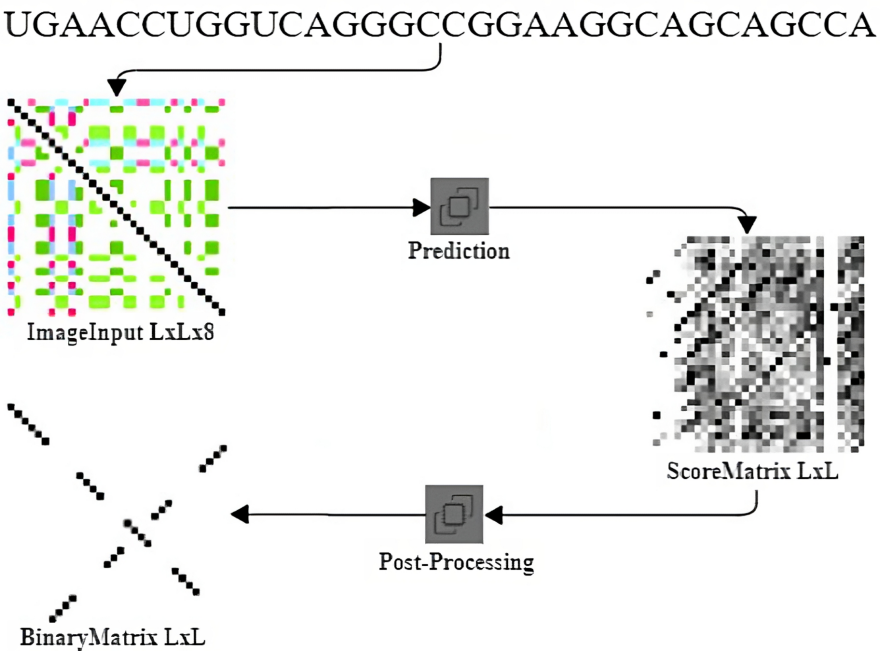

Figura 2.1: Esempio di Predizione

Nella Figura 2.1 viene riportato un esempio del risultato desiderato, ovvero una rete neurale che, presa una rappresentazione dei possibili accoppiamenti di un dato RNA, restituisca una score matrix che mediante post-processing verrà convertita in una matrice binaria contenente gli accoppiamenti previsti.

2.2 Rappresentazione di input e output in Tensori

La rappresentazione utilizzata nella rete neurale proposta fa riferimento a quella utilizzata nella rete CNNFold [21] in cui, presa una sequenza di RNA lunga L, i dati vengono rappresentati mediante un tensore di input LxLx8 e una matrice binaria di output LxL.
Nel tensore di input LxLx8, ciascun vettore (i,j) del tensore, rappresenta mediante la tecnica one-hot una delle 8 possibili relazioni previste tra i nucleotidi i e j.
In particolare:

- Posizione 1: rappresenta un accoppiamento invalido, nello specifico si tratta di accoppiamenti i,j che non soddisfano i vincoli indicati nella sezione precedente.
- Posizione 2: rappresenta basi potenzialmente non accoppiate, nello specifico questo livello rappresenta i casi in cui i=j, ovvero gli elementi sulla diagonale del tensore.
- Posizione 3: rappresenta il possibile accoppiamento Adenina-Uracile
- Posizione 4: rappresenta il possibile accoppiamento Uracile-Adenina
- Posizione 5: rappresenta il possibile accoppiamento Uracile-Guanina
- Posizione 6: rappresenta il possibile accoppiamento Guanina-Uracile

- Posizione 7: rappresenta il possibile accoppiamento Guanina-Citosina
- Posizione 8: rappresenta il possibile accoppiamento Citosina-Guanina

Per quanto riguarda la matrice binaria di output LxL, ciascuna cella (i,j) assume:

- 1: se le base i e j sono accoppiate
- 0: se le basi i e j non sono accoppiate

Questa rappresentazione comporta diversi vantaggi, fra cui la semplicità nel rappresentare sia accoppiamenti locali che a lunga distanza, distinti dalla distanza degli accoppiamenti dalla diagonale principale. D'altro canto, è importante segnalare anche la facile individuazione della presenza di segmenti stem: si possono osservare blocchi di basi consecutive accoppiate ad un altro blocco di basi, rappresentate da una sequenza di elementi non nulli nella matrice Y paralleli o ortogonali alla diagonale principale.

2.3 Modello di predizione

Nella sezione precedente è stato definito l'input come un tensore LxLx8. L'obbiettivo proposto consiste nell'addestrare una rete neurale che utilizzerà questo tensore per calcolare, mediante una serie di convoluzioni, una matrice score bidimensionale LxL, in cui ogni (i,j)esimo elemento rappresenterà la probabilità che quel dato accoppiamento si verifichi. La struttura predetta verrà successivamente estratta mediante il post-processing. Solo in fase di addestramento verrà confrontata con la matrice binaria di output per definire la Loss e i successivi parametri di addestramento durante l'apprendimento, oltre a valutarne la precisione in fase di valutazione. Il modello di rete neurale proposto utilizzerà principalmente strati convoluzionali e blocchi residuali.
Gli strati convoluzionali rappresentano una delle componenti fondamentali delle reti neurali convoluzionali, questi consentono di estrarre automaticamente caratteristiche significative dalle immagini o dai dati. In particolare, gli strati convoluzionali utilizzano una finestra mobile (il cosiddetto filtro o kernel) che si sposta sull'input e ne estrae le caratteristiche locali, come linee, curve o forme geometriche.
I blocchi residuali sono una tecnica di aggiornamento dei pesi che aiutano a risolvere un problema del Deep Learning, ovvero la scomparsa del gradiente durante la retropropagazione. I blocchi residuali utilizzano una connessione "skip" che consente al segnale di bypassare alcuni strati, mantenendo le informazioni di basso livello ed evitando che si disperdano durante la retropropagazione.
In sintesi, la nostra rete neurale utilizza una combinazione di strati convoluzionali e blocchi residuali per estrarre e apprendere automaticamente le caratteristiche rilevanti dai dati di input.

2.3.1 Livelli della Rete Neurale

Il modello della rete proposta EbutRna utilizza dunque una serie di livelli convoluzionali, e una serie di blocchi residuali a loro volta composti da 2 livelli convoluzionali.
Ciascun livello convoluzionale è seguito da un Batch Normalization Layer, che normalizza i dati per evitare l'aumento esponenziale dei gradienti e ridurre l'eccessiva specializzazione del modello, e un Leaky Relu Layer di attivazione.

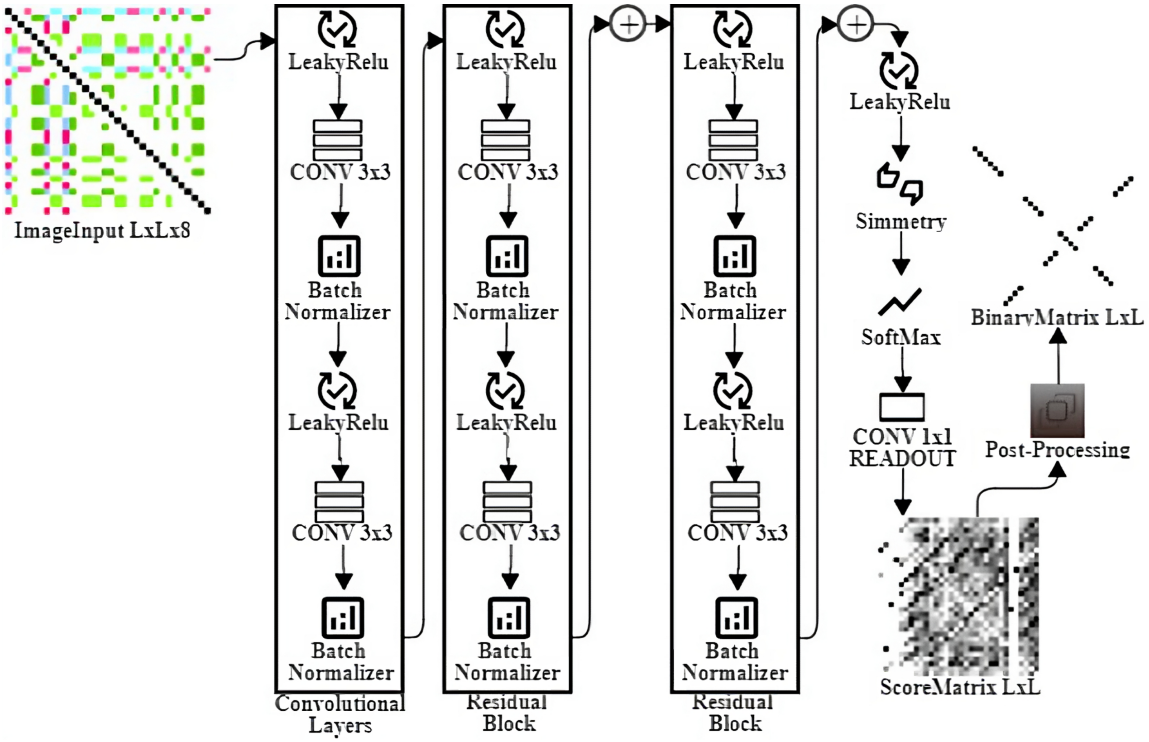

Figura 2.2: Schema della Rete Neurale

In Figura 2.2 viene rappresentata la struttura completa della rete composta da diversi strati che lavorano in sinergia per trasformare l'immagine di input in un'output predetto. In particolare, l'Image Input Layer converte l'immagine iniziale in un tensore tridimensionale, in questo livello non è necessario applicare alcuna normalizzazione dato che come input la rete riceverà un tensori contenenti solo zeri e uni, successivamente il tensore viene elaborato dal Leaky Relu Layer che, a differenza della classica attivazione Relu, prevede un valore di attivazione minimo diverso da zero per i valori di input negativi, impedendo così la saturazione dell'attivazione e prevenendo il problema del dying ReLU.
Successivamente i due Livelli Convoluzionali 3x3, rappresentati nella prima colonna della Figura 2.2, estraggono le caratteristiche principali del tensore, mentre i due Blocchi Residuali, rappresentati nella seconda e terza colonna, ne migliorano l'accuratezza. Il livello per garantire la simmetria calcola il valore medio dell'output e della sua trasposta, mantenendo così la simmetria nella rete neurale. In seguito il Soft Max Layer converte l'output del layer precedente in una distribuzione di probabilità, mentre il Convolutional2D Layer 1x1 viene utilizzato come READOUT per produrre l'output finale, ovvero la Score Matrix.

2.3.2 Scelta dell'ottimizzatore

Durante i test dello studio condotto per questo elaborato sono stati presi in considerazione due dei principali ottimizzatori: Stochastic Gradient Descent e Adam Optimization Algorithm.
Stochastic Gradient Descent (SGD) è un algoritmo di ottimizzazione molto utilizzato nell'addestramento di modelli di apprendimento automatico. Esso calcola il gradiente in modo stocastico, utilizzando un singolo esempio di addestramento alla volta, rendendo l'algoritmo molto più efficiente e adatto per grandi set di dati. Inoltre, SGD introduce una certa casualità nell'aggiornamento dei pesi del modello, in questo modo vengono evitati minimi locali indesiderati.
D'altra parte, l'algoritmo di ottimizzazione Adam è un algoritmo di discesa del gradiente stocastico utilizzato per l'ottimizzazione dei pesi in una rete neurale. Esso è di tipo adattativo, adattando il passo di apprendimento per ogni peso in modo da aggiornarli in modo più efficiente.
In pratica, Adam mantiene una stima del momento del primo ordine (la media mobile esponenziale dei gradienti) e del momento del secondo ordine (la media mobile esponenziale del quadrato dei gradienti) per ogni peso.
Utilizzando queste stime, l'algoritmo adatta il passo di apprendimento per ogni peso in modo da aggiornarli in modo più efficiente.
Adam è particolarmente utile per grandi dataset dato che combina i vantaggi di altre tecniche di ottimizzazione: RMSProp e Momentum.
RMSProp adatta la dimensione del passo di aggiornamento dei pesi in base alla storia delle precedenti iterazioni, mentre Momentum riduce l'impatto dei gradienti che variano ampiamente.
In seguito mostreremo come, durante i primi test effettuati Adam Optimization Algorithm, si sia rivelato più efficente sui dataset utilizzati.

2.3.3 Funzione Loss

Durante l'addestramento, ad ogni iterazione sul miniBatch viene calcolata la Loss sulla base della quale vengono successivamente aggiornati il gradiente e quindi i parametri di addestramento attraverso la funzione adamupdate. Applichiamo la seguente Loss ad ogni pattern presente nel miniBatch:

$$\mathrm{Loss_k}(Y_k, T_k) = \frac{1}{|N|} \sum_{ij \in N} (Y_{ij} - T_{ij})^2$$

Con:

- Y_k output del k-esimo elemento del Batch
- T_k Matrice T del k-esimo elemento del Batch
- N è l'insieme di tutte le coppie di nucleotidi

Successivamente la Loss dell'intero batch verrà calcolata con: $\mathrm{l} = \frac{1}{K} \sum_{k=1}^{K} Loss_k$

2.4 Post-processing

Dopo che la rete neurale ha prodotto una matrice di punteggi, che indica la probabilità di accoppiamento per ogni coppia di basi dell'RNA, è necessario effettuare un post-processing per ottenere la struttura finale. Gli approcci per questo processo possono essere diversi, come l'utilizzo di metodi basati sulla programmazione dinamica o sull'algebra dei grafi.
Ad esempio, l'algoritmo di Zuker [3], utilizzato per la predizione della struttura secondaria dell'RNA, può essere adattato all'utilizzo della score matrix prodotta dalla rete neurale.
Un altro approccio basato sull'algebra dei grafi è l'algoritmo di Blossom [22], che cerca di trovare il massimo accoppiamento perfetto in un grafo bipartito associato alla score matrix.

Nel nostro caso, abbiamo scelto di utilizzare un metodo di post-processing molto semplice, ma efficace. Grazie alla simmetria mantenuta dalla rete neurale, abbiamo selezionato l'accoppiamento a probabilità più alto per ogni colonna della matrice Y.
Ad esempio, se nella colonna i-esima il valore massimo è associato all'elemento j, abbiamo selezionato l'accoppiamento tra le basi i e j. Questo metodo è stato sufficiente per ottenere risultati soddisfacenti nei nostri esperimenti.

Capitolo 3

Risultati

3.1 Metodo di valutazione

Nello studio condotto è stato utilizzato come metodo di valutazione la cross-validation, o validazione incrociata, vengono valutate le prestazioni del modello di previsione utilizzando Train Set e Test Set distinti. Il dataset completo viene quindi diviso in due parti, un set di addestramento, che comprende l'80% del dataset, e un set di valutazione che comprende il restante 20%. La Cross-Validation può essere valutata in due modi diversi, basata sulle famiglie o sulle sequenze. Nella Cross-Validation sulle famiglie il Test Set ha le stesse famiglie del Train Set, le sequenze non sono ridondanti anche se possono esserci somiglianze strutturali. Nella Cross-Validation sulle sequenze, invece, il Test Set contiene campioni di famiglie diverse rispetto a quelle del set di allenamento. Anche in questo caso, possono verificarsi somiglianze strutturali, ma sono meno probabili. Le analisi condotte ai fini di questo studio utilizzeranno la cross-validation basata sulle sequenze. Le nostre analisi utilizzeranno la cross-validation basata sulle sequenze.

3.2 Sistema di valutazione

I modelli addestrati sono stati valutati utilizzando l'F1 Score.
In statistica, l'F1 score è una metrica che valuta l'accuratezza di un test di classificazione binaria. Tale metrica tiene conto della precisione e del recupero del test, dove la precisione rappresenta il numero di veri positivi (TP) diviso il numero di tutti i risultati positivi (TP + FP), mentre il recupero indica il numero di veri positivi (TP) diviso il numero di tutti i test che sarebbero dovuti risultare positivi (TP + FN). L'F1 score viene calcolato tramite la media armonica di precisione e recupero.

$$Precisione = \frac{TP}{TP+FP}$$

$$Recupero = \frac{TP}{TP+FN}$$

$$F1 = 2 \cdot \frac{Precision \cdot Recupero}{Precision + Recupero}$$

3.3 Datasets

Ci sono tre dataset comunemente usati per la predizione della struttura dell'RNA.

1. RNAStralign

   Campioni: 30438

   Lunghezze: variabili tra 30 e 1851 nucleotidi

   Famiglie RNA: 8

   Campioni: 3572

   Famiglie RNA: 10

   16S, 5S, Gruppo I Intron,

   RNaseP, SRP, telomerasi,

   tmRNA, tRNA

2. ArchiveII

   Campioni: 3572

   Lunghezze: variabili tra 28 e 2927 nucleotidi

   Famiglie RNA: 10

   16S, 5S, 23S,

   Gruppo I Intron, Gruppo II Intron,

   RNaseP, SRP, telomerasi,

   tmRNA, tRNA

3. bpRNA

   Campioni: 11992

   Lunghezze: variabili tra 22 e 498 nucleotidi

   Famiglie: diverse famiglie di RNA creati da Rfam 12.2

3.4 Fase di Test e Debug dell'addestramento

Nella fase di test e debug della rete neurale sono stati utilizzati entrambi gli ottimizzatori su ciascuno dei dataset in 20 epoche, limitando, per motivi di gestione di memoria e limiti temporali, il numero di Train Pattern a 1500.
Faremo ora un confronto tra i due ottimizzatori osservando le variazioni della Loss durante l'addestramento, l'F1 Score su alcuni degli elementi del Test Set presi singolarmente e l'F1 Score sull'intero Train Set.

3.4.1 Loss durante l'addestramento

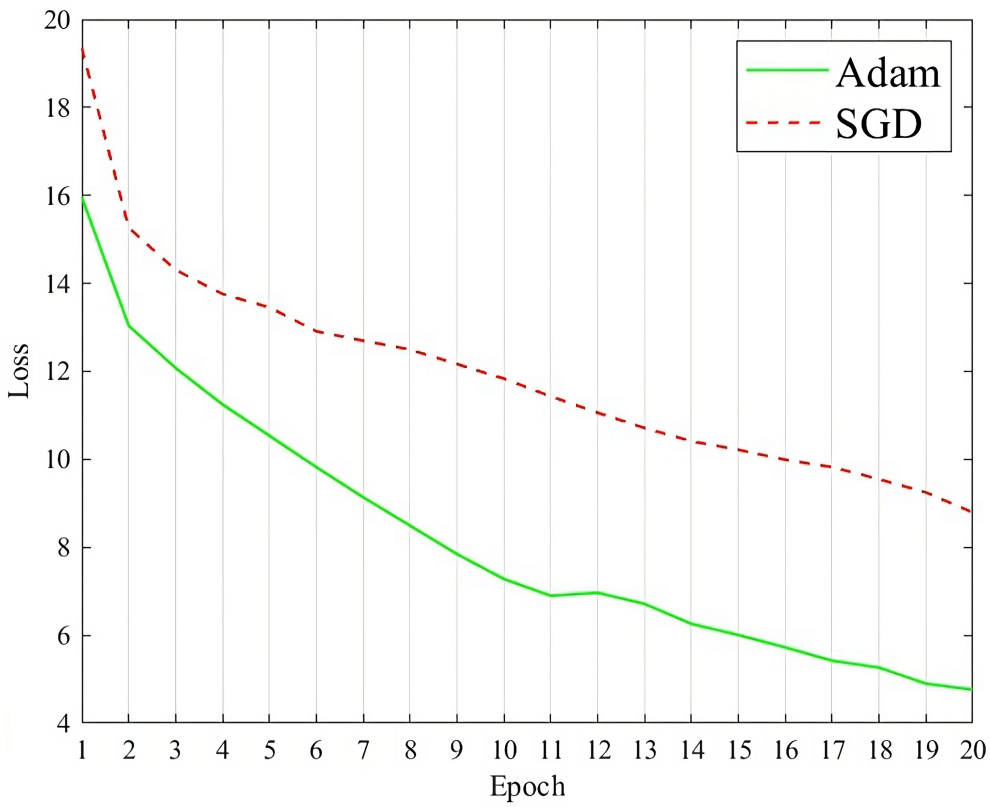

Figura 3.1: Comparazione della variazione della Loss durante l'addestramento su dataset bprna

La Figura 3.1 rappresenta le variazioni della Loss durante l'addestramento utilizzando il dataset bprna limitato a 1500 pattern. A fini rappresentativi per ciascuna Epoch è stata calcolata la media delle Loss, in tal modo è evidente la differenza nei due addestramenti. La linea verde rappresenta la variazione della Loss nell'addestramento effettuato con adamupdate, mentre la linea rossa rappresenta la variazione della Loss nell'addestramento che utilizza SGD. Notiamo che l'addestramento con l'ottimizzazione Adam Based porta fin da subito ad una Loss più bassa e dunque più precisa.

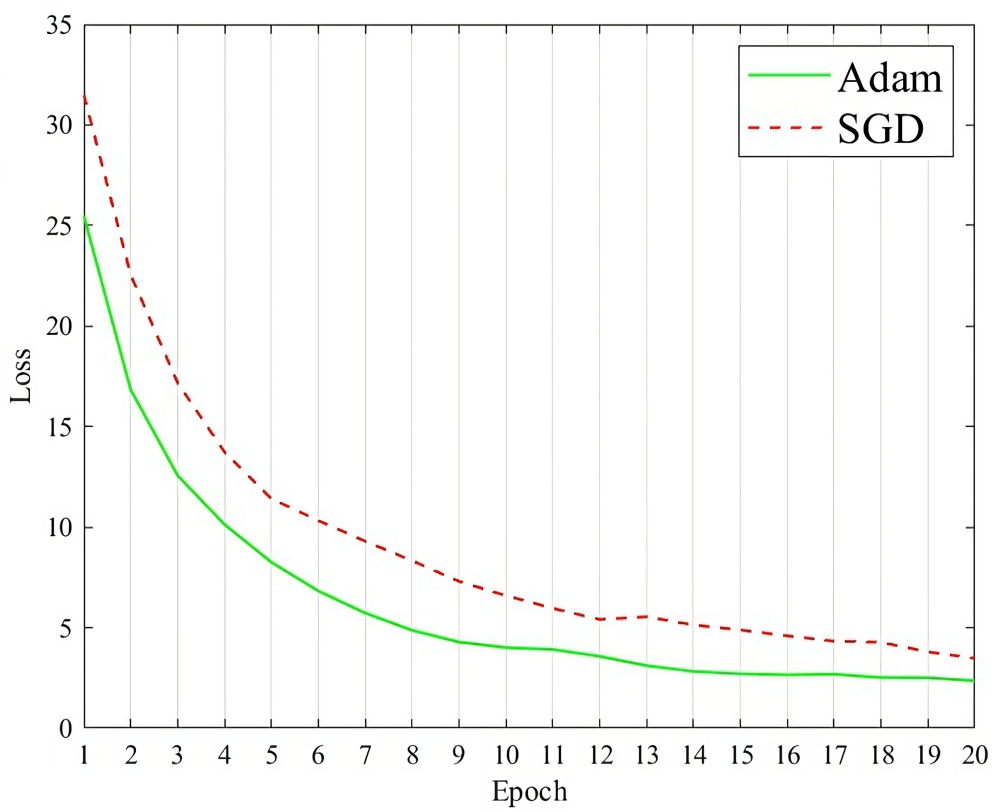

Figura 3.2: Comparazione della variazione della Loss durante l'addestramento su dataset ArchiveII

La Figura 3.2 rappresenta le variazioni della Loss durante l'addestramento utilizzando il dataset ArchiveII limitato a 1500 pattern. A fini rappresentativi per ciascuna Epoch è stata calcolata la media delle Loss, in tal modo è evidente la differenza nei due addestramenti. La linea verde rappresenta la variazione della Loss nell'addestramento effettuato con adamupdate, mentre la linea rossa rappresenta la variazione della Loss nell'addestramento che utilizza SGD. Notiamo che l'apprendimento segue praticamente lo stesso andamento con i diversi ottimizzatori ma Adam Optimizer porta comunque ad una rete meglio addestrata dato che riesce fin da subito a predirre in maniera più precisa di SGD.

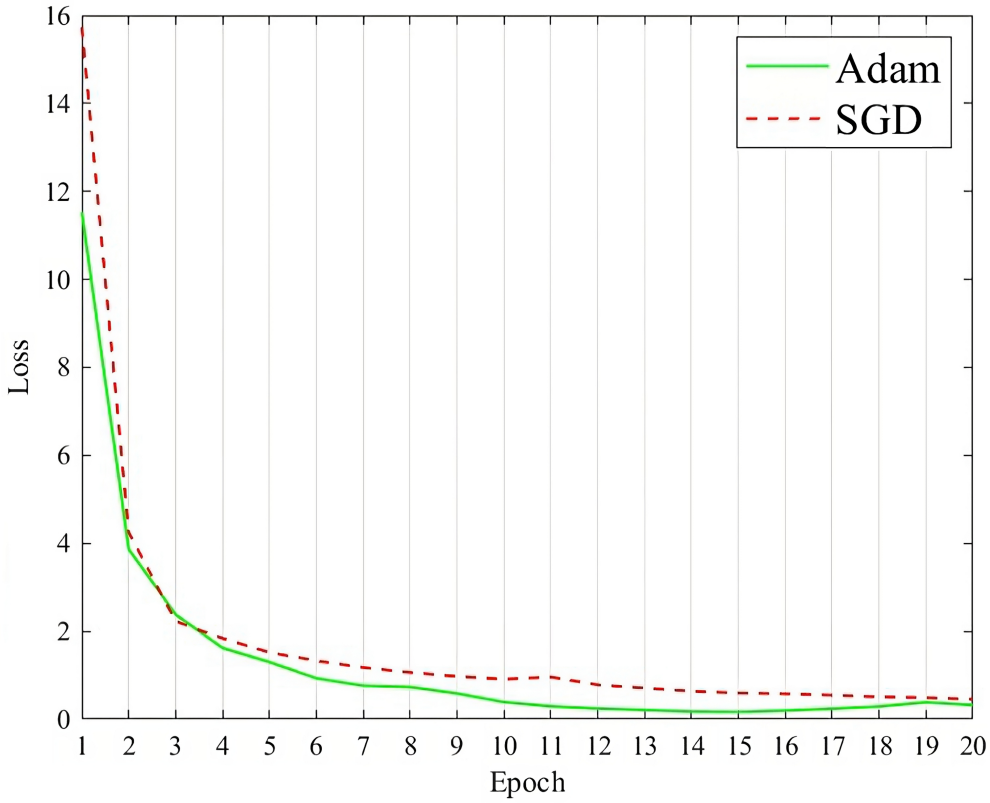

Figura 3.3: Comparazione della variazione della Loss durante l'addestramento su dataset rnastralign

La Figura 3.3 rappresenta le variazioni della Loss durante l'addestramento utilizzando il dataset rnastralign limitato a 1500 pattern. A fini rappresentativi per ciascuna Epoch è stata calcolata la media delle Loss, in tal modo è evidente la differenza nei due addestramenti. La linea verde rappresenta la variazione della Loss nell'addestramento effettuato con adamupdate, mentre la linea rossa rappresenta la variazione della Loss nell'addestramento che utilizza SGD. Notiamo che, nonostante l'andamento simile, Adam Optimizer ottiene un risultato migliore.

3.4.2 F1 Score su singoli elementi del Test Set

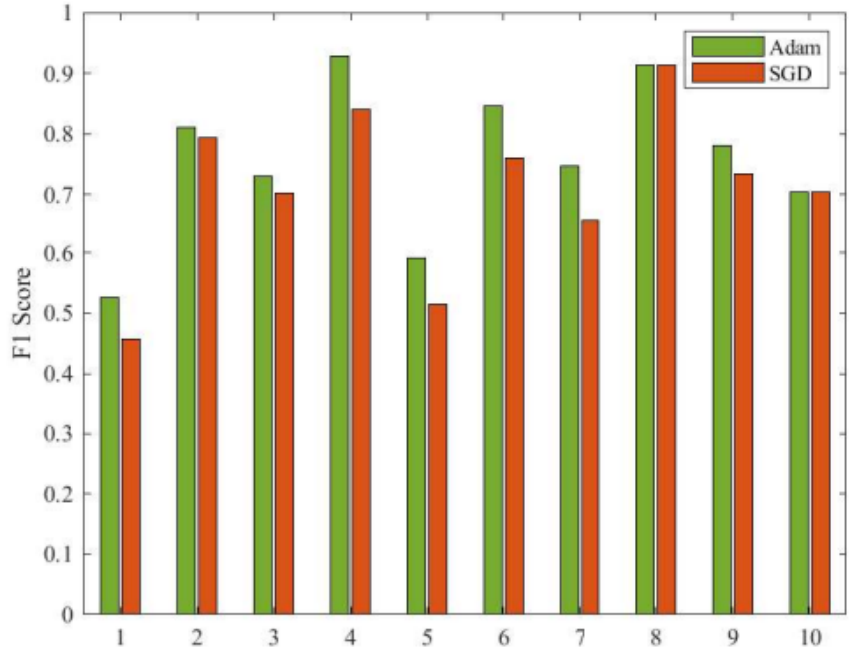

Figura 3.4: Confronto dell'F1 Score su 10 Pattern casuali del Test Set di bprna

Nella Figura 3.4 viene presentato un confronto dell'F1 Score ottenuto su 10 pattern casuali del Test Set di bprna con Adam Optimizer e SGD. Notiamo che in generale la precisione ottenuta dalla rete addestrata con Adam è superiore o uguale alla precisione ottenuta dalla rete addestrata con SGD.

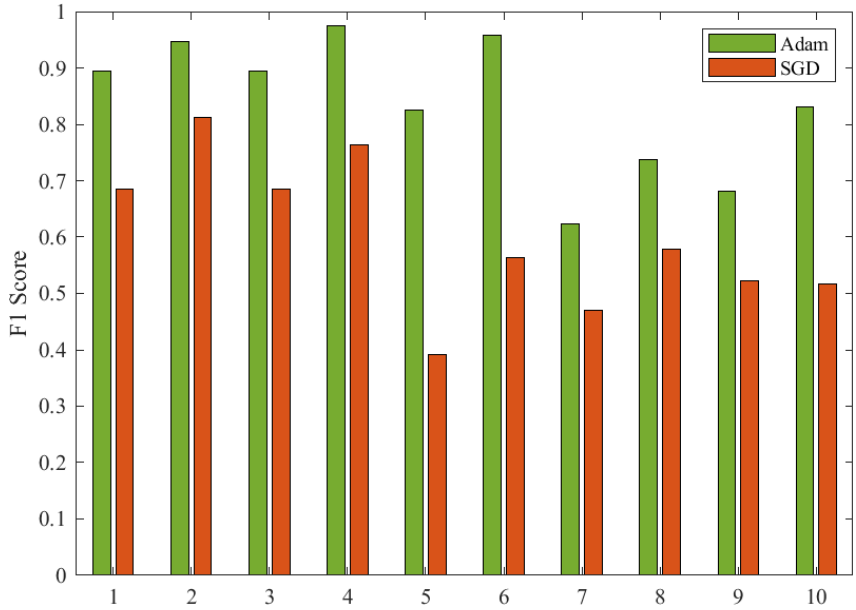

Figura 3.5: Confronto dell'F1 Score su 10 Pattern casuali del Test Set di ArchiveII

Nella Figura 3.5 viene presentato un confronto dell'F1 Score ottenuto su 10 pattern casuali del Test Set di ArchiveII con Adam Optimizer e SGD. Notiamo che la precisione ottenuta dalla rete addestrata con Adam è di molto superiore alla precisione ottenuta dalla rete addestrata con SGD.

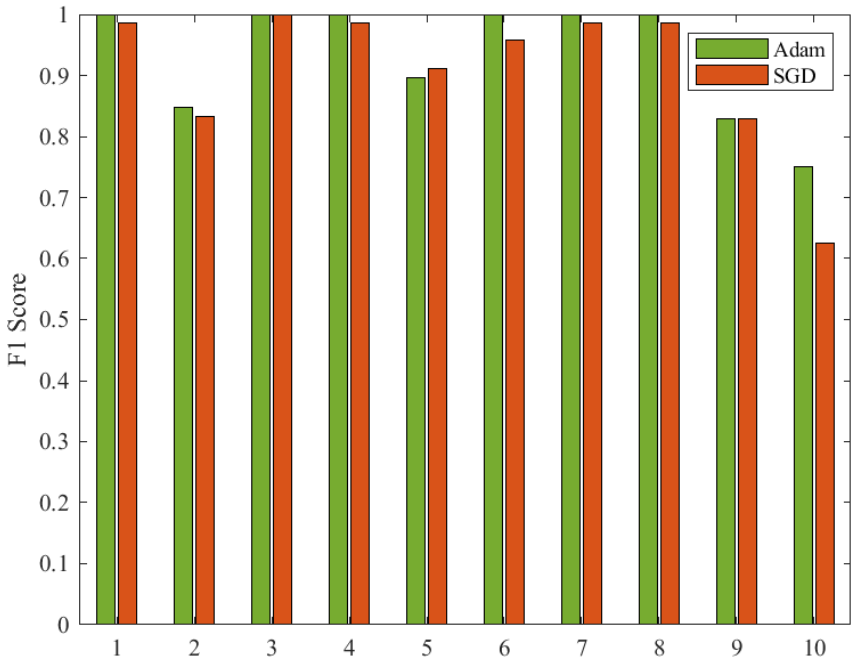

Figura 3.6: Confronto dell'F1 Score su 10 Pattern casuali del Test Set di rnastralign

Nella Figura 3.6 viene presentato un confronto dell'F1 Score ottenuto su 10 pattern casuali del Test Set di rnastralign con Adam Optimizer e SGD. Notiamo spesso risultati simili, che riflettono la vicinanza della Loss durante il training. Anche in questo caso però Adam Optimizer restituisce nella maggior parte dei casi risultati più precisi.

3.4.3 F1 Score sull'intero Test Set

	F1 Score	
	Adam	SGD
bpRNA	0.7493	0.7189
ArchiveII	0.9105	0.6609
rnastralign	0.9812	0.8208

Tabella 3.1: Valori dell'F1 Score nei primi test eseguiti con 1500 pattern nel Train Set

Nella Tabella 3.1 sono riportati i valori della precisione calcolata mediante F1 Score di tutte le reti precedentemente addestrate, utilizzando come Set di valutazione un sottoinsieme del Set completo contenente pattern lunghi al massimo quanto il pattern più lungo del Set di addestramento.
Da questi risultati e anche dalle precedenti analisi, notiamo che le reti addestrate con Adam Optimizer sono più precise e pertanto utilizzeremo questo ottimizzatore nell'addestramento delle reti con Dataset completo.

3.5 Parametri del Modello

Visti i risultati ottenuti durante la fase di test i parametri principali di addestramento sono stati gestiti da Adam Optimizer attraverso la funzione di ottimizzazione adamupdate di matlab.
La funzione adamupdate prende come input i valori correnti dei pesi, i gradienti calcolati per i pesi, un vettore contenente le statistiche del primo momento dei gradienti e un vettore contenente le statistiche del secondo momento dei gradienti. La funzione restituisce i nuovi valori dei pesi aggiornati tramite l'ottimizzatore ADAM.
Le reti sono state addestrate per 20 epoche e con un miniBatch di 10 pattern.
La dimensione delle immagini nel miniBatch è stata gestita attraverso il padding, ovvero attraverso l'aggiunta di dati supplementari a un pattern per raggiungere la lunghezza del pattern con dimensione maggiore all'interno del miniBatch, questo per evitare l'utilizzo di resize che avrebbero comportato una perdita di dati in fase di post-processing. Lo shuffle dei dati è stato escluso dopo i primi test dato che causava un aumento della dimensione media dei miniBatch, e di conseguenza un aumento esponenziale del tempo di addestramento.

3.6 Addestramento con l'intero Train Set

BpRNA

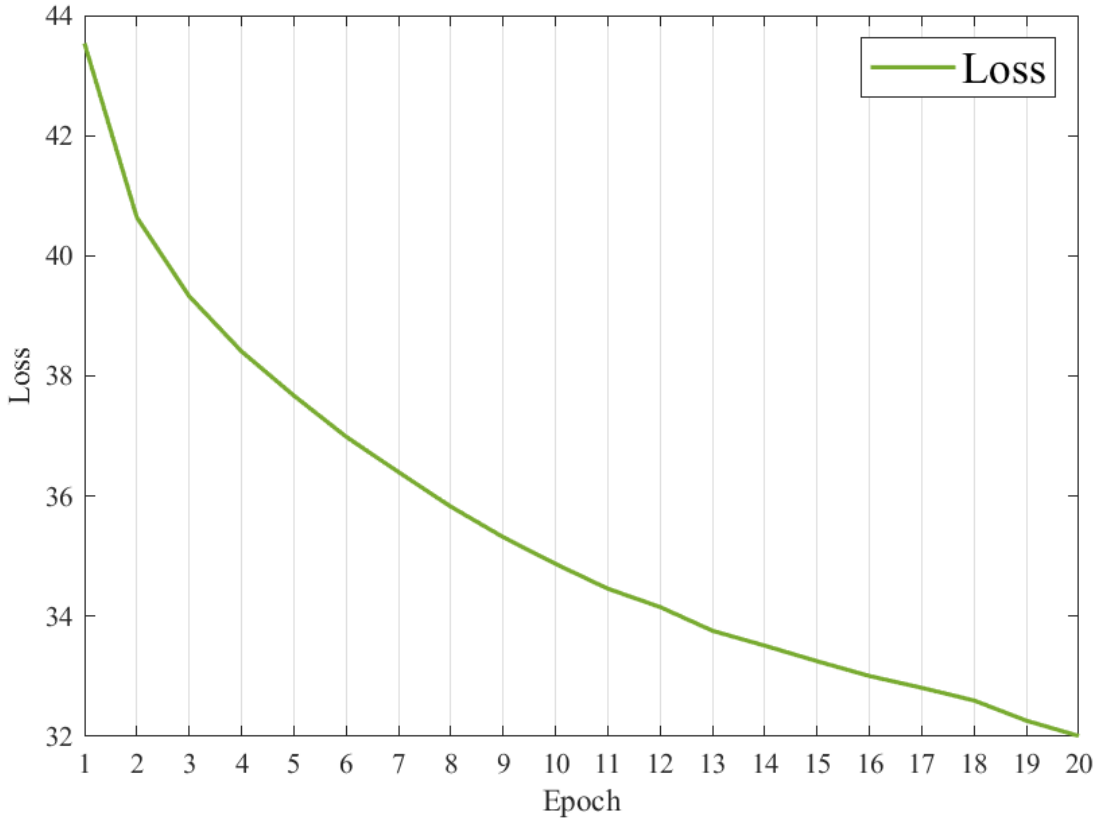

Figura 3.7: Variazione della Loss durante l'addestramento della Rete con Train Set bprna

La Figura 3.7 rappresenta la variazione media per epoca della Loss durante l'addestramento di EbutRna utilizzando il dataset BPRNA.

	Precision	**Recall**	**F1 score**
[15] SPOT-RNA	0.652	0.578	0.597
[16] MXFold2	0.520	0.682	0.575
[21] CNNFold-mix	0.640	0.566	0.582
EbutRna	0.662	0.662	0.662

Tabella 3.2: Valori di precisione, recall e F1 score per diverse reti Addestrate con bpRNA

Nella Tabella 3.2 viene confrontato il risultato dell'F1 Score per diverse reti addestrate con il dataset BpRNA. Notiamo che il risultato di EbutRna è di poco superiore a quello degli altri modelli analizzati.

ArchiveII

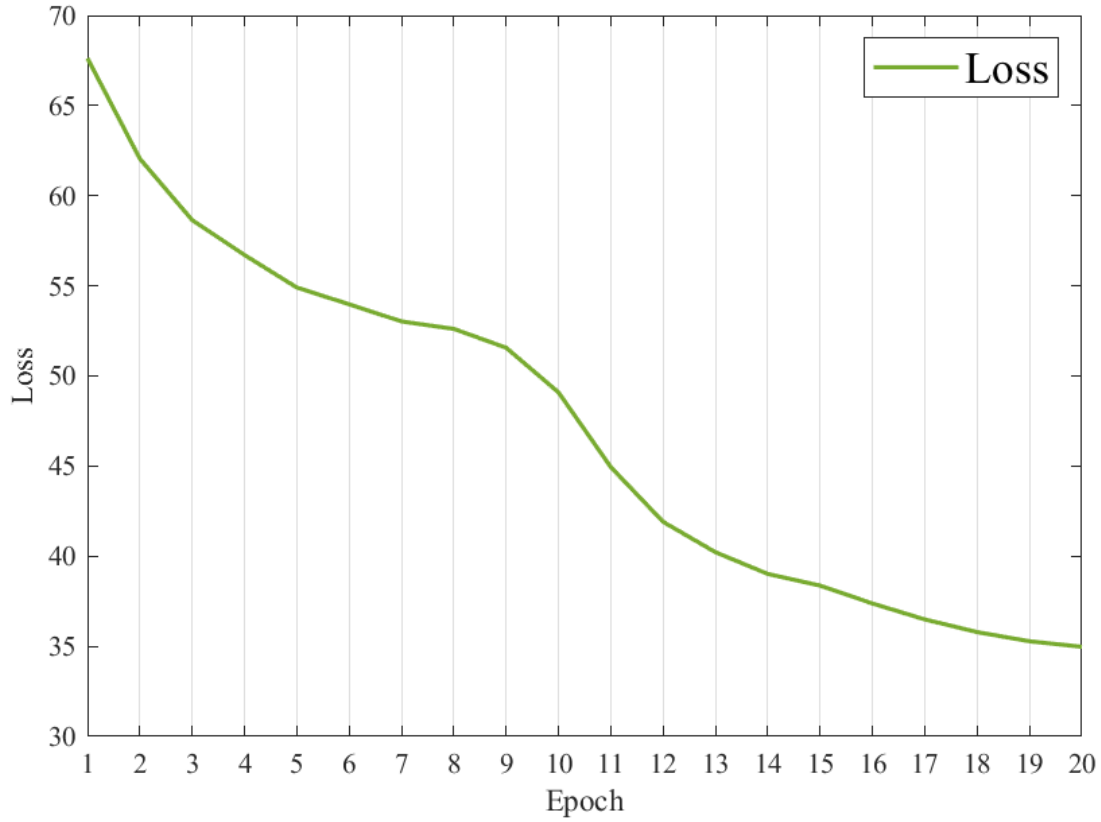

Figura 3.8: Variazione della Loss durante l'addestramento della Rete con Train Set bprna

La Figura 3.8 rappresenta la variazione media per epoca della Loss durante l'addestramento di EbutRna utilizzando il dataset ArchiveII contenente pattern lunghi al massimo 600 nucleotidi.

	Precision	**Recall**	**F1 score**
[6] Unafold	0.428	0.383	0.401
[13] CDPfold	0.557	0.535	0.545
[7] RNAstructure	0.563	0.615	0.585
[8] RNAfold	0.565	0.627	0.592
[23] LinearFold	0.641	0.617	0.621
[12] CONTRAfold	0.607	0.679	0.638
[14] E2Efold	0.734	0.660	0.686
[16] MXFold2	0.790	0.815	0.800
[21] CNNFold-mix	0.928	0.879	0.897
EbutRna	0.671	0.671	0.671

Tabella 3.3: Valori di precisione, recall e F1 score per diverse reti Addestrate con ArchiveII

Nella Tabella 3.3 viene confrontato il risultato dell'F1 Score per diverse reti addestrate con il dataset ArchiveII. Notiamo che il risultato di EbutRna nella media rispetto agli altri metodi analizzati ma significativamente peggiore del metodo CNNFold, probabilmente a causa della semplificazione del post-processing.

Capitolo 4

Conclusione

Negli ultimi anni, la predizione della struttura secondaria dell'RNA ha acquisito sempre più importanza grazie alla possibilità di fornire informazioni preziose sulla funzione biologica dell'RNA. In questo lavoro è stato presentato un metodo basato sull'utilizzo di una rete neurale convoluzionale chiamato EbutRna per la predizione della struttura secondaria dell'RNA.
Il metodo proposto ha dimostrato di essere efficace nel fornire risultati promettenti su diversi dataset pubblici, la cui precisione è stata misurata utilizzando l'F-measure. Inoltre, confrontando i risultati ottenuti con quelli di altri metodi di predizione della struttura secondaria dell'RNA, è stato dimostrato che EbutRna ha ottenuto risultati superiori o conformi su diversi dataset.
Tuttavia, ci sono ancora diverse sfide da affrontare in questo campo, nonostante l'efficacia dimostrata dal metodo proposto. Nella sezione seguente, verranno esaminate alcune delle limitazioni del modello e i possibili sviluppi futuri, al fine di migliorare ulteriormente la predizione della struttura secondaria dell'RNA.

4.1 Limiti e sviluppi futuri

Nonostante la buona performance del modello, sono presenti alcune limitazioni che devono essere affrontate. Ad esempio, attualmente il modello è in grado di prevedere solo campioni di famiglie utilizzate nel TrainSet, poiché l'apprendimento non considera alcun dato termodinamico come l'energia libera. L'aggiunta di informazioni termodinamiche durante il training del modello potrebbe aiutare nella generalizzazione.
Inoltre, per considerare la lunghezza della sequenza RNA, potrebbe essere addestrato un insieme di reti specializzate per una determinata lunghezza dei pattern. In questo modo, il modello potrebbe prevedere con precisione la struttura secondaria dell'RNA in modo più efficace su diverse lunghezze di sequenza.

Un'altra possibile implementazione sarebbe quella di considerare le famiglie dell'RNA, poiché le performance del modello dipendono dalle famiglie utilizzate nel training. In tal caso, potrebbe essere implementata una fase di rilevazione della famiglia e successivamente l'utilizzo di una rete specifica per la famiglia rilevata.

Infine, potrebbe essere generato un insieme di strutture sub-ottimali, ovvero implementare diversi metodi di post-processing, come ad esempio euristiche basate sull'MFE.
In generale, è importante comprendere le limitazioni del metodo proposto e di altri algoritmi basati su Deep Learning per la predizione della struttura secondaria dell'RNA. L'accuratezza di questi modelli è eccezionale, ma è fondamentale capire quanto bene il modello può predire elementi strutturali biologicamente importanti. Ad esempio, gli pseudonodi sono difficili da prevedere, ma altrettanto importanti dato che la loro presenza o assenza nella struttura tridimensionale ha un effetto significativo sulle proprietà funzionali della struttura. Ci sono ancora molte sfide da affrontare nella predizione della struttura secondaria dell'RNA, ma le reti neurali convoluzionali come EbutRna rappresentano un importante passo avanti verso la realizzazione di algoritmi di predizione più accurati e affidabili.
Il codice del modello EbutRna realizzato sarà disponibile nella relativa repository github.

Bibliografia

[1] R.B. Lyngsø e C.N.S. Pedersen. «Pseudoknots in RNA secondary structures». In: *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology* (2000).

[2] M. Zuker e D. Sankoff. «RNA secondary structures and their prediction». In: *Bulletin of mathematical biology* (1984).

[3] M. Zuker e P. Stiegler. «Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information». In: *Nucleic acids research* (1981).

[4] T. Xia et al. «Thermodynamic parameters for an expanded nearest-neighbor model for formation of RNA duplexes with Watson-Crick base pairs». In: *Biochemistry* (1998).

[5] D.H. Turner e D.H. Mathews. «NNDB: the nearest neighbor parameter database for predicting stability of nucleic acid secondary structure». In: *Nucleic Acids Research* (2009).

[6] N.R. Markham e M. Zuker. «UNAFold: software for nucleic acid folding and hybridization». In: *Bioinformatics* (2008).

[7] S. Bellaousov et al. «RNAstructure: web servers for RNA secondary structure prediction and analysis». In: *Nucleic acids research* (2013).

[8] R. Lorenz et al. «ViennaRNA Package 2.0». In: *Algorithms for Molecular Biology* (2011).

[9] K. Sato et al. «IPknot: fast and accurate prediction of RNA secondary structures with pseudoknots using integer programming». In: *Bioinformatics* (2011).

[10] S. Bellaousov e D. H. Mathews. «ProbKnot: Fast prediction of RNA secondary structure including pseudoknots». In: *RNA* (2010).

[11] J.B. Chaires, J.A. McFail e T.L. Johnston. «Assessing the accuracy of free energy calculation methods for RNA molecules». In: *Biopolymers* (2015).

[12] C.B. Do, D.A. Woods e S. Batzoglou. «CONTRAfold: RNA secondary structure prediction without physics-based models». In: *Bioinformatics* (2006).

[13] H. Zhang et al. «A new method of RNA secondary structure prediction based on convolutional neural network and dynamic programming». In: *Frontiers in Genetics* (2019).

[14] X. Chen et al. «RNA Secondary Structure Prediction By Learning Unrolled Algorithms». In: *International Conference on Learning Representations* (2020).

[15] J. Singh et al. «RNA secondary structure prediction using an ensemble of two-dimensional deep neural networks and transfer learning». In: *Nature Communications* (2019).

[16] K. Sato, M. Akiyama e Y. Sakakibara. «Rna secondary structure prediction using deep learning with thermodynamic integration». In: *Nature Communications* (2021).

[17] A. Vaswani et al. «Attention is all you need». In: *Advances in Neural Information Processing Systems* (2017).

[18] A.W. Senior et al. «Improved protein structure prediction using potentials from deep learning». In: *Nature* (2020).

[19] P. Eastman et al. «Solving the RNA design problem with reinforcement learning». In: *PLoS computational biology* (2018).

[20] R. V Koodli et al. «Eternabrain: automated RNA design through move sets and strategies from an internet-scale RNA videogame». In: *PLoS computational biology* (2019).

[21] M. Saman Booy, A. Ilin e P. Orponen. «RNA secondary structure prediction with convolutional neural networks». In: *BMC Bioinformatics* (2022).

[22] Z. Galil. «Efficient algorithms for finding maximum matching in graphs». In: *ACM Computing Surveys (CSUR)* (1986).

[23] D. Deng et al. «LinearFold: Linear-Time Prediction of RNA Secondary Structures». In: *bioRxiv* (2018).