UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

"Analisi comparativa delle tecniche di deep learning per l'identificazione di virus"

Relatore: Prof. Matteo Comin

Laureando: Filippo Galli

ANNO ACCADEMICO 2022 – 2023

Data di laurea 28 Settembre 2023

Indice

Abstract

Attraverso l'impiego delle recenti tecnologie di sequenziamento, l'identificazione di virus attraverso il metagenoma è diventato un aspetto sempre più cruciale, superando le limitazioni dei tradizionali metodi basati sulle colture di laboratorio e consentendo l'analisi di un'ampia diversità virale. Tra i vari approcci disponibili, l'utilizzo di tecniche di Deep Learning si dimostra particolarmente promettente in questa sfida, poiché permette di apprendere automaticamente le caratteristiche genomiche virali e di rilevare sequenze virali all'interno di campioni metagenomici con un'elevata accuratezza, soprattutto per sequenze di input di breve lunghezza. Questa tesi fornisce una valutazione dettagliata della struttura e delle prestazioni di alcuni metodi di Deep Learning che fanno uso di diverse tipologie di reti neurali. Attraverso il loro confronto, si mira a identificare il metodo in grado di fornire le prestazioni più accurate e affidabili nella rilevazione delle sequenze virali.

Capitolo 1

Introduzione

Grazie alle nuove tecnologie di sequenziamento, l'identificazione dei virus a partire dal metagenoma ha assunto un ruolo sempre più significativo per i ricercatori.

A differenza dei metodi tradizionali che isolano i virus attraverso colture di laboratorio, il sequenziamento metagenomico permette di sequenziare efficacemente tutti i tipi di materiale genetico presente in una comunità microbica, indipendentemente dalla loro coltivabilità. Ciò apre la strada allo studio di una diversità virale molto più ampia, superando le limitazioni dei metodi convenzionali [1].

L'identificazione dei frammenti virali direttamente dal genoma può tuttavia presentare alcune problematiche. Uno dei principali problemi riguarda la presenza di un grande numero di sequenze brevi che, proprio a causa della loro lunghezza limitata, rendono difficile la distinzione tra sequenze virali e sequenze provenienti da altri organismi o da contaminazioni esterne [2].

Nel corso degli ultimi anni sono stati sviluppati numerosi metodi e strumenti per affrontare questa problematica. In questa sezione forniremo una panoramica di questi metodi e metteremo in evidenza le loro caratteristiche.

1.1 Metodi basati sull'allineamento

I metodi basati sull'allineamento permettono di identificare le sequenze di DNA confrontandole con i genomi dei virus presi in esame con lo scopo di trovare delle regioni di somiglianza. Ciò avviene mediante l'uso di software di allineamento come ad esempio DIAMOND [3] e ProViDE [4] che cercano di allineare le sequenze di query con quelle di riferimento [5].

Il processo di allineamento implica la creazione di una scoring matrix in cui, dal confronto dei nucleotidi, viene assegnato un punteggio che si basa sulla loro somiglianza. La

somiglianza tra due sequenze si misura attraverso due parametri principali: l'identità e la conservazione della sequenza.

Il primo indica la percentuale di nucleotidi corrispondenti tra le due sequenze mentre il secondo indica la percentuale di caratteri che si trovano nella stessa posizione e che hanno proprietà fisico-chimiche simili.

Dopo l'allineamento viene valutato il grado di somiglianza tra la sequenza di query e quella di riferimento e, se la somiglianza supera un determinato punteggio di soglia, la sequenza di query viene identificata come virus noto.

I principali svantaggi di questo metodo sono il fatto che esso necessita di un grande tempo di calcolo dovuto all'allineamento[6] delle sequenze e la scarsa precisione, dovuta invece alla tendenza delle sequenze di mutare nel tempo rendendo più difficile il loro allineamento con i genomi di riferimento [7].

1.2 Metodi basati sui geni

I metodi basati sui geni consentono di identificare sequenze virali estraendo i geni presenti all'interno delle sequenze prese in esame. Questi vengono poi mappati e confrontati con un database di geni virali noti, al fine di determinare la presenza di virus nel campione.

L'estrazione dei geni dalle sequenze di metagenomi avviene tramite l'uso di software specializzati, come Prodigal [8], che ne individua i codoni di inizio e fine estraendo la sequenza di nucleotidi compresa tra questi punti.

Se un gene rappresentato è omologo a un gene virale noto, viene classificato come virale. La presenza di più geni virali nella stessa sequenza aumenta la probabilità che la sequenza sia di origine virale.

Tra gli svantaggi di questo metodo vi è la sua limitata affidabilità nella previsione di sequenze di metagenomi di breve lunghezza poiché potrebbero non contenere geni. Inoltre, gli analisti devono selezionare manualmente le caratteristiche dei geni da considerare, rendendo questa selezione spesso soggettiva e non sempre ottimale [2].

1.3 Metodi basati sui k-mers

I metodi basati sui k-mers sono metodi computazionali utilizzati per l'identificazione di somiglianze tra sequenze di DNA. Questi metodi si basano sull'analisi dei k-mer, ovvero sottostringhe di lunghezza fissa k in cui vengono suddivise le sequenze. Attraverso la suddivisione di ciascuna sequenza in sottostringhe si crea un elenco composto da tutti i suoi k-mer che verranno confrontati con altri k-mer provenienti da sequenze di riferimento.

Esempi di metodi basati sui k-mers sono Kraken [9] e CLARK [10] i quali utilizzano un database pre-costruito di k-mer per confrontare i k-mer delle sequenze di query e determinare l'origine tassonomica delle sequenze.

Altri metodi come ad esempio Taxonomer [11] utilizzano i k-mer per fornire un'indicazione sulla frequenza di specifici k-mer all'interno di una sequenza che può essere utilizzata come segnale per rilevare la presenza di virus.

I metodi basati su k-mer presentano tuttavia alcune limitazioni. Una di queste è dovuto al fatto che molti virus possono superare i meccanismi di difesa degli ospiti mimando le loro sequenze e rendendone difficile l'identificazione. Inoltre, quando le sequenze sono molto brevi, la frequenza dei k-mer può essere troppo scarsa per fornire abbastanza informazioni discriminanti, riducendo di conseguenza le prestazioni nell'identificazione di sequenze virali corte.

Un'altra considerazione importante riguarda il valore di k. Un valore di k più grande porta a un maggior numero di possibili combinazioni di k-mer unici e richiede perciò maggiori risorse di calcolo. Tuttavia, un valore di k più grande potrebbe allo stesso tempo migliorare le prestazioni dell'estrazione delle caratteristiche.

È importante notare che esistono anche altri approcci e varianti dei metodi basati su k-mer che possono mitigare alcune di queste limitazioni. Ad esempio, l'uso di tecniche di machine learning può consentire una migliore classificazione delle sequenze in base ai k-mer, come accade ad esempio nel metodo VirFinder [12].

1.4 Metodi basati sul Deep Learning

I metodi basati sul deep learning si basano principalmente sull'utilizzo dei Deep Neural Network (DNN) per apprendere automaticamente le caratteristiche genomiche virali al fine di rilevare sequenze virali all'interno di sequenze metagenomiche. Questi modelli vengono addestrati utilizzando un gran numero di sequenze e riescono a raggiungere un'accuratezza relativamente elevata nell'identificazione delle sequenze virali, soprattutto per quanto riguarda sequenze di input di breve lunghezza.

Possiamo suddividere questi metodi in tre sottocategorie in base alle reti neurali utilizzate [13]. È molto comune che i metodi di identificazione basati sul deep learning facciano uso di più tipologie di reti neurali.

- **Reti neurali convoluzionali (CNN)**: Le reti neurali convoluzionali sono in grado di estrarre automaticamente caratteristiche rilevanti dalle sequenze attraverso l'utilizzo di strati convoluzionali [14].

  Uno strato convoluzionale consiste in diversi filtri che hanno lo stesso numero di canali dei dati di input. Ogni filtro agisce come una finestra scorrevole esplorando

l'input passo dopo passo e calcolando un output per ciascuna posizione. Ciò consente alla CNN di rilevare pattern locali o caratteristiche rilevanti nell'input.

Le CNN sono infatti ottimi strumenti per analizzare dati con dipendenze spaziali. Poiché la sequenza del DNA è un dato unidimensionale, quando si applicano le CNN è solito utilizzare la codifica "one-hot" per gestire le quattro basi del DNA. Ad esempio è possibile codificare ogni base del DNA come A = [1,0,0,0], G = [0,1,0,0], C = [0,0,1,0], T = [0,0,0,1], in modo che una sequenza di DNA diventi una matrice con quattro colonne a cui possa essere applicato il CNN [13]. Tuttavia questa metodologia rende ogni entità codificata tramite vettori one-hot indipendente dalle altre, il che non rispecchia la realtà. Ad esempio, tre basi di DNA possono essere strutturate in un codone, e le basi con una lunghezza fissa possono essere rappresentate come un gene. Le caratteristiche ottenute dalla codifica one-hot sono discrete e sparse, e queste limitazioni hanno un effetto negativo sull'accuratezza dell'identificazione delle sequenze virali. Per questa ragione è molto meglio esprimere una sequenza di nucleotidi attraverso diverse dimensioni significative, ognuna delle quali contiene un numero reale continuo per rappresentare diversi gradi invece di valori binari. Alcuni strumenti che fanno uso delle CNN sono ad esempio DeepVirFinder [15] e PPR-META [1].

- **Reti neurali ricorrenti (RNN)**: Le reti neurali ricorrenti sono particolarmente adatte per l'analisi di sequenze, in quanto sono in grado di considerare il contesto sequenziale delle informazioni riconoscendo pattern ricorrenti o regioni che si ripetono all'interno della sequenza [13]. Mentre le reti neurali convoluzionali possono essere utilizzate per catturare informazioni spaziali nei dati, le reti neurali ricorrenti vengono utilizzate per catturarne il comportamento dinamico temporale. Nell'ambito dell'identificazione di sequenze di virus il funzionamento è il seguente: un input, come ad esempio una sequenza codificata di DNA, viene combinato con l'output dell'ultimo strato nascosto della RNN precedente, per essere poi utilizzato come input per la successiva RNN, consentendo alla rete di catturare il comportamento delle sequenze di virus all'interno del DNA.

  Ci sono due caratteristiche chiave che rendono le RNN particolarmente adatte nell'identificazione di sequenze di virus nel DNA.

  Innanzitutto, le RNN consentono di gestire sequenze di lunghezze diverse per l'input e l'output. Questo è importante perché quando trattiamo con sequenze di basi nucleotidiche nel genoma virale, mentre l'input può essere una sequenza di una determinata lunghezza, l'output può essere sia una sequenza diversa, uguale o una classe che indica la presenza o l'assenza di un virus specifico. Le RNN sono in grado di adattarsi a queste variazioni di lunghezza in modo flessibile, consentendo di lavorare con sequenze eterogenee adattandosi a diversi casi di studio.

Le RNN, inoltre, condividono i parametri tra i diversi passi temporali all'interno della stessa rete neurale. Questo semplifica l'addestramento del modello e riduce la complessità computazionale complessiva permettendo alle RNN di apprendere da un insieme di dati più ampio senza dover aumentare eccessivamente il numero di parametri da ottimizzare. Alcuni strumenti che fanno uso delle RNN sono Virtifier[2] e Detire[16] (CNN + RNN)

- **Reti neurali generative**: Le reti neurali generative, come le Variational Autoencoders (VAE) e i Generative Adversarial Networks (GAN), possono essere utilizzate per generare nuove sequenze genomiche simili a quelle dei virus noti. Questi modelli possono apprendere la distribuzione statistica delle sequenze virali generando a partire da questa nuove sequenze virali simili a quelle di addestramento [13]. Questo approccio può essere utile per identificare sequenze virali che non corrispondono esattamente a quelle presenti nel genoma di riferimento.

Capitolo 2

Metodi di Deep Learning

Nel corso di questa sezione verranno illustrati i metodi di identificazione di deep learning presi in esame, fornendo una panoramica completa delle metodologie utilizzate e delle soluzioni offerte per massimizzare la possibilità di una corretta identificazione dei virus all'interno del genoma. Tutti i metodi esaminati si basano sull'utilizzo del database RefSeq [17] per l'addestramento e la valutazione dei modelli.

2.1 VirFinder

VirFinder [12] è un metodo che si basa sull'analisi della frequenza dei k-mer tramite machine learning con lo scopo di identificare frammenti di genoma virale. Piuttosto che basarsi su somiglianze genetiche note o su confronti con database di sequenze virali, VirFinder si concentra sul riconoscimento delle caratteristiche distintive dei k-mer virali rispetto ai k-mer degli organismi ospiti presenti nel campione. Questo approccio consente di identificare sequenze virali anche in assenza di conoscenze pregresse sulle sequenze o sui geni specifici, basandosi invece sulla sola distinzione delle firme di k-mer tra virus e procarioti.

2.1.1 Metodi

Preparazione delle sequenze Il modello prende in considerazione frammenti di sequenze lavorando su singole parole w, definite come sequenze di nucleotidi di lunghezza arbitraria k, e dalla loro controparte complementare $\overline{w}$. Per ogni parola w, viene calcolato il numero di occorrenze N(w) all'interno della sequenza, sia per la parola stessa che per il suo complementare N($\overline{w}$). Le frequenze delle parole vengono poi normalizzate dividendo il numero di occorrenze per la somma di tutte le occorrenze delle parole w presenti nella sequenza:

$$V(\mathbf{w}) = \frac{N(\mathbf{w})}{\sum_{\mathbf{w}} N(\mathbf{w})}, \mathbf{w} \in \mathcal{A}^k \equiv \{A, C, G, T\}$$

Si ottiene così la firma della sequenza, rappresentata da V(w). A partire dal dataset di addestramento, composto dallo stesso numero di sequenze virali e sequenze ospiti, si fa uso di un test statistico, il test t [18], per confrontare la frequenza media delle varie parole w nelle sequenze virali e nelle sequenze ospiti con l'obiettivo di verificare se esiste una differenza significativa tra le frequenze delle parole nei due gruppi. Per superare il problema della multicollinearità, viene esclusa la parola con il valore p più alto. Ciò viene fatto per evitare che due o più variabili siano altamente correlate tra loro causando problemi nella stima dei coefficienti del modello.

Modello di regressione logistica e funzione obiettivo In seguito, sulla base delle parole ottenute, viene utilizzato il modello di regressione logistica per costruire un classificatore binario e calcolare la probabilità che una sequenza S_i sia virale ($Y_i = 1$) od ospite ($Y_i = 0$), con i=1,2,. . . ,k.

Dalla modellazione otteniamo la seguente funzione obiettivo:

$$-\frac{1}{\mathrm{n}} \sum_{i=1}^{n} \log l\left(Y_i \mid V_i(\mathbf{w}), \beta(\mathbf{w}), \beta_0\right) + \lambda \sum_{\mathbf{w} \in \mathcal{A}^k} |\beta(\mathbf{w})|$$

$-\frac{1}{\mathrm{n}} \sum_{i=1}^{n} \log l\left(Y_i \mid V_i(\mathbf{w}), \beta(\mathbf{w}), \beta_0\right)$ rappresenta il valore medio della log-likelihood per tutte le n sequenze nel dataset, utilizzato per misurare quanto bene il modello di regressione logistica si adatta ai dati di addestramento.

$\lambda \sum_{\mathbf{w} \in \mathcal{A}^k} |\beta(\mathbf{w})|$ rappresenta invece la regolarizzazione del modello, utilizzata per evitare l'overfitting[19] e per promuovere la sparsità dei coefficienti, inducendo alcuni di essi a diventare zero. Questo può aiutare a ridurre l'effetto di coefficienti non significativi o ridondanti nel modello. Il parametro che controlla l'intensità della regolarizzazione è λ.

L'obiettivo finale è minimizzare questa funzione, trovando i valori del coefficiente $\beta(\mathbf{w})$ che massimizzi la verosimiglianza dei dati. La scelta del parametro λ viene effettuata utilizzando la validazione incrociata a 10 fold per massimizzare il punteggio AUROC.

2.2 DeepVirFinder

DeepVirFinder [15] è un metodo di deep learning sviluppato per prevedere la presenza di sequenze virali all'interno di sequenze di DNA attraverso l'utilizzo di una rete neurale convoluzionale. Il modello prende in input le sequenze di DNA apprendendone le caratteristiche e permettendo di classificare una sequenza come virus o procariote, restituendo come output un punteggio di predizione compreso tra 0 e 1.

Questo metodo si basa sui risultati ottenuti dal predecessore VirFinder, il quale ha dimostrato che i virus e i procarioti hanno preferenze diverse nell'uso dei k-mers. Si passa

per questa ragione dall'utilizzo dei k-mers ad una loro generalizzazione in motivi, che vengono rappresentati mediante matrici di peso posizionali.

2.2.1 Metodi

L'idea alla base di DeepVirFinder è quella di utilizzare questi motivi come proprietà del modello, migliorando la sua accuratezza di previsione. Ciò avviene tramite una rete neurale convoluzionale in cui i filtri, ovvero i motivi, sono in grado di catturare i pattern delle sequenze permettendo di estrarne le informazioni rilevanti.

Fornita in input una sequenza di DNA X_R, questa viene elaborata dalla rete neurale attraverso una serie di passaggi: convoluzione, pooling, dense layer e applicazione della funzione sigmoide per generare il punteggio di previsione. Essi sono visibili in Figura 2.1.

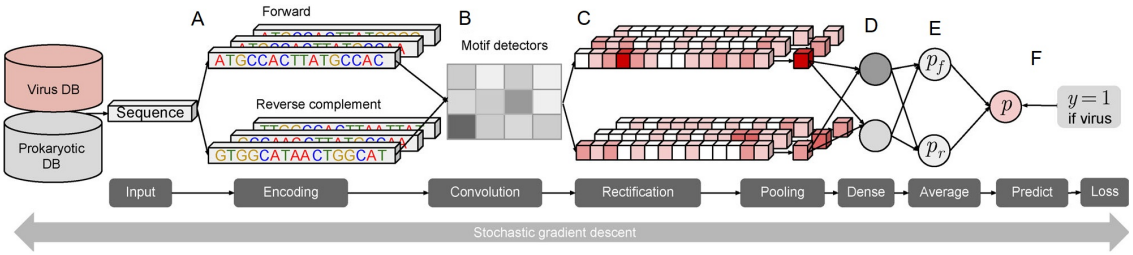


Figura 2.1: Rappresentazione del framework di DeepVirFinder.

Codifica delle sequenze e strato convoluzionale

Inizialmente la sequenza di lunghezza L viene codificata utilizzando la codifica one-hot che converte ogni base nucleotidica in un vettore di dimensione 4, in cui il valore contenuto in ciascuna posizione indica la presenza o l'assenza di una specifica base. In caso di nucleotidi ambigui questi vengono codificati in vettori specifici $\left[\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right]^{\mathrm{T}}$. Al termine del processo l'input sarà rappresentato come una matrice $Z^{(1)}$ di dimensione 4 × L.

Successivamente la matrice codificata viene passata attraverso uno strato convoluzionale con attivazione rectifier. Lo strato convoluzionale contiene M motivi di lunghezza K, dove ogni motivo viene rappresentato da una matrice U_m di dimensione 4 × k. Ogni riga di questa matrice rappresenta una base nucleotidica e ogni colonna rappresenta una posizione nel motivo. Le singole celle della matrice contengono invece i coefficienti che determinano le probabilità di avere una specifica base nucleotidica in una determinata posizione del motivo.

Ognuno di questi motivi viene convoluto sulla matrice $Z^{(1)}$ per ottenere una serie di intensità del motivo. Le intensità dei motivi risultanti per tutti i M motivi vengono infine rappresentate come un vettore $Z^{(2)}$ di dimensione M × (L - K + 1).

Rectification e max pooling

In seguito, per ogni motivo, viene applicata un'attivazione ReLU che restituisce una matrice $Z^{(3)}$ della stessa dimensione del vettore di intensità del motivo. La funzione del ReLU è quella di sostituire qualsiasi valore negativo della matrice con 0 lasciando invece inalterati i valori positivi e rendendo quindi la rete neurale più efficiente.

Viene in seguito applicato uno strato di max pooling per ridurre la dimensione della matrice mantenendo solo l'intensità massima per ogni motivo, restituendo una matrice di dimensione M × 1.

Dense layer e output score

La matrice di output viene poi passata attraverso uno strato completamente connesso che contiene N neuroni, aventi ciascuno un vettore di peso e un bias associato. L'output del layer completamente connesso è una matrice di dimensione N × 1.

Infine, dopo aver applicato un altro ReLU, l'output viene sommato utilizzando uno strato denso con funzione sigmoide per generare un punteggio di previsione compreso tra 0 e 1. Un punteggio più alto indica una maggiore probabilità di essere una sequenza virale, mentre un punteggio più basso indica una maggiore probabilità di essere una sequenza procariotica. Poiché una sequenza di DNA è costituita da un doppio filamento e le sequenze reali possono derivare da entrambi i filamenti è essenziale fare in modo che il punteggio di previsione sia indipendente dalla direzione di lettura. Per questo motivo viene applicata la stessa rete neurale al complemento inverso della sequenza di input originale (X_F) e il punteggio di previsione finale sarà ottenuto calcolando la media dei punteggi di previsione delle sequenze originali e delle corrispondenti sequenze complementari $Y_{\text{final}} = \frac{Y(X_{\text{F}})+Y(X_{\text{R}})}{2}$

2.2.2 Effetto delle mutazioni sulla previsione del modello

Il modello è stato sottoposto a diverse prove per valutarne la robustezza alle mutazioni genetiche. I virus presentano tassi di mutazione più elevati rispetto ai batteri, ed è fondamentale valutare la capacità del modello di gestire tali mutazioni e la sua sensibilità agli errori di sequenziamento.

Per valutare la robustezza del modello alle mutazioni genetiche, è stato utilizzato il dataset di test per introdurre mutazioni casuali all'interno delle sequenze. Le mutazioni sono state introdotte in base a tre diverse frequenze: 0,001, 0,01 e 0,1. Per ogni posizione all'interno della sequenza il nucleotide originale viene perciò sostituito casualmente con un altro nucleotide con la stessa probabilità, in base alla frequenza specificata.

Successivamente, è stata utilizzata l'AUROC per valutare e confrontare le prestazioni predittive del modello in base alle diverse frequenze di mutazione.

Dall'analisi condotta, lo studio ha evidenziato che i punteggi di AUROC hanno subito una riduzione minima anche in presenza di mutazioni. In particolare, i punteggi sono diminuiti di meno dello 0,06% a una frequenza di mutazione del 0,001, dello 0,66% a una frequenza del 0,01 e del 7,96% a una frequenza del 0,1. Questi risultati, visibili nella Figura 2.2, dimostrano che il modello è in grado di mantenere le sue capacità predittive anche di fronte a diverse frequenze di mutazione.

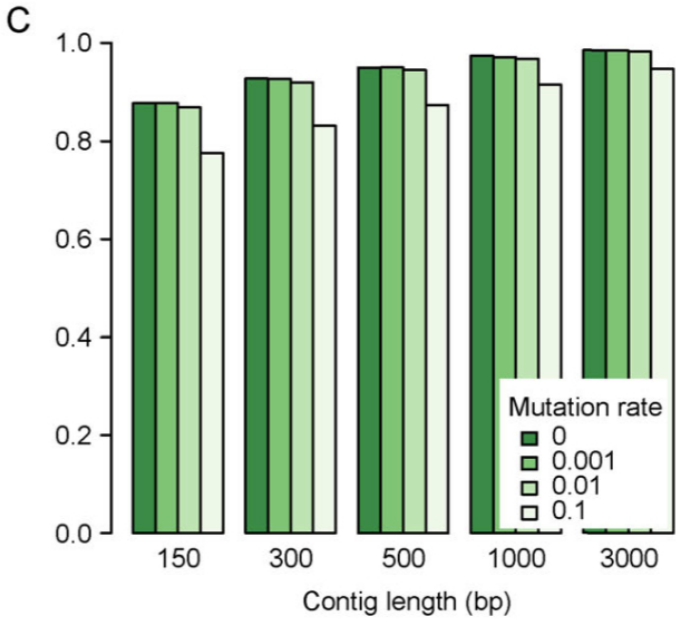


Figura 2.2: prestazioni di DeepVirFinder in base alla frequenza delle mutazioni

2.3 PPR-META

PPR-Meta [1] è un classificatore a tre classi progettato per l'identificazione simultanea di frammenti di fagi, plasmidi e cromosomi all'interno di sequenze metagenomiche. Questo metodo fa uso di un Bi-path Convolutional Neural Network, una rete neurale convoluzionale costituita da due percorsi e progettata per migliorare le prestazioni nella classificazione di frammenti di DNA di lunghezza ridotta a partire da due parametri di ingresso.

2.3.1 Metodi

PPR-Meta rappresenta ogni sequenza di DNA nel dettaglio attraverso una matrice one-hot per le basi (BOH) e una matrice one-hot per i codoni (COH). Questo approccio è stato adottato a causa delle frequenti fluttuazioni delle frequenze dei k-mer all'interno di sequenze brevi. Tali fluttuazioni possono variare notevolmente anche tra sequenze geneticamente simili, influenzando negativamente la capacità del modello di riconoscere correttamente le sequenze biologiche. In questa rappresentazione BOH rappresenta le singole basi e la loro posizione nella sequenza, mentre COH tiene conto della struttura dei codoni e delle relazioni tra di essi. L'uso combinato di entrambe le rappresentazioni fornisce al modello

una maggiore capacità di apprendere e riconoscere i pattern, migliorando le prestazioni complessive.

Nella rappresentazione BOH, ogni base nucleotidica viene rappresentata tramite un vettore "one-hot" di dimensione 4, come avviene nei modelli precedentemente presentati. Considerando anche il filamento complementare, una sequenza di lunghezza L può essere rappresentata da una matrice BOH di dimensione $2 \times L \times 4$.

Per quanto riguarda COH, ogni sequenza viene prima suddivisa in 6 fasi sotto forma di codoni. Ad esempio, una sequenza come 5'-ACGTTCGAACG-3' viene divisa nelle seguenti 6 sequenze di codoni:

ACG, TTC, GAA

CGT, TCG, AAC

GTT, CGA, ACG

CGT, TCG, AAC

GTT, CGA, ACG

TTC, GAA, CGT

In seguito ciascun codone ottenuto viene rappresentato attraverso un vettore one-hot di dimensione 64, una per ogni possibile codone. Pertanto, una sequenza di lunghezza L può essere rappresentata da una matrice di lunghezza $2 \times L \times 64$.

Struttura della rete neurale

La struttura della rete neurale, chiamata BiPathCNN, è stata progettata per migliorare le prestazioni nella classificazione dei frammenti di DNA di lunghezza ridotta. Ciò avviene attraverso l'utilizzo di da due percorsi principali: il "base path" e il "codon path", che ricevono in input rispettivamente le matrici BOH e COH.

Ciascuno dei due percorsi paralleli è composto da diversi tipi di layer, ciascuno con una funzione specifica, che contribuiscono alla trasformazione dei dati di input. Questi layer sono progettati per eseguire operazioni come la convoluzione, il max pooling, la normalizzazione batch e la concatenazione, al fine di estrarre le caratteristiche importanti dai dati e generare una classificazione accurata delle sequenze.

I layer, illustrati nella Figura 2.3, possono essere suddivisi nelle seguenti tipologie:

Layer di convoluzione (b1, c1, b7, c7): utilizzano un kernel di convoluzione per estrarre le informazioni locali dai dati di input. Per introdurre non linearità nella rete, viene applicata la funzione di attivazione ReLU.

Layer di max pooling (b2, c2): riducono la dimensione delle mappe delle caratteristiche estraendo le informazioni più rilevanti.

Layer di normalizzazione batch (b3, c3): normalizzano i valori delle mappe delle caratteristiche per migliorare la convergenza e ridurre l'overfitting.

Layer di convoluzione successivi (b4-b6, c4-c6): contengono un numero maggiore di kernel di convoluzione per estrarre ulteriori informazioni di alto livello.

Layer di average pooling (b8, c8): calcolano la media globale delle mappe delle caratteristiche.

Layer di concatenazione (9-11): combinano l'output del percorso delle basi e del percorso dei codoni. Dopo un layer completamente connesso, viene utilizzato il layer softmax per calcolare la probabilità che il frammento di input sia un fago, un cromosoma o un plasmide.

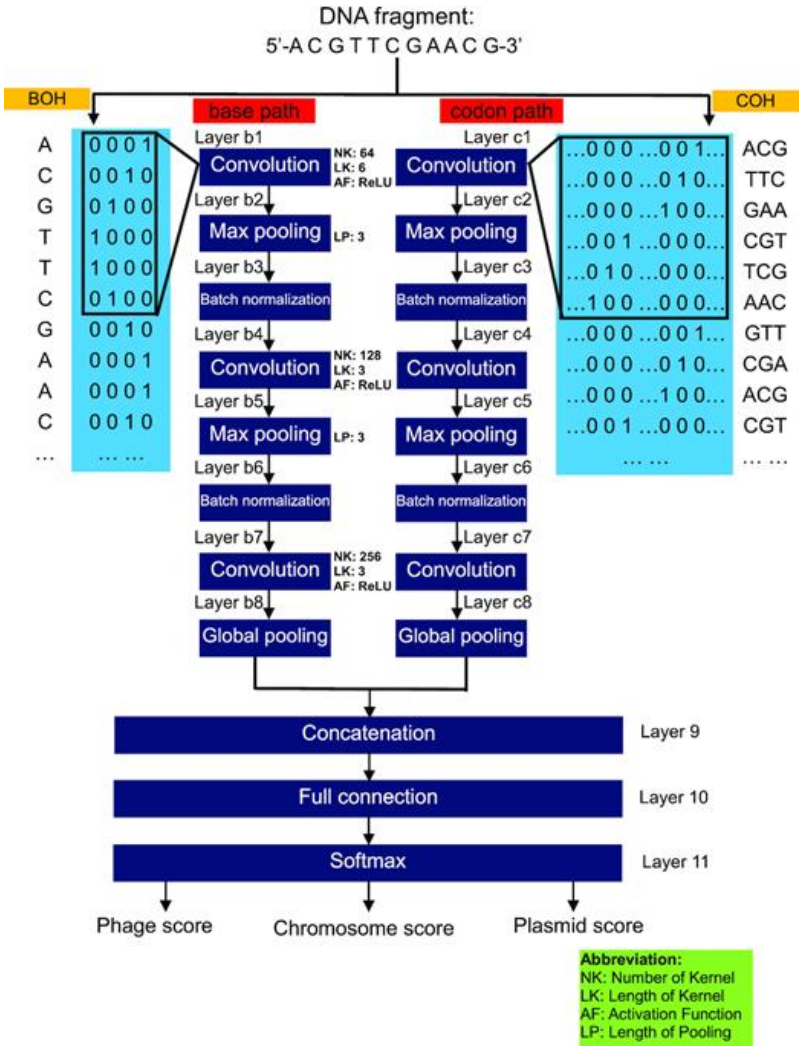


Figura 2.3: Struttura di BiPathCNN

2.3.2 Utilizzo di diversi gruppi per l'addestramento delle reti neurali

Per fare in modo che il modello si adatti in maniera ottimale in base alle diverse dimensioni delle sequenze, sono state addestrate tre diverse reti neurali per ciascun gruppo di sequenze. La rete neurale BiPathCNN A gestisce le sequenze di lunghezza compresa tra 100 e 400 bp, BiPathCNN B gestisce le sequenze tra 400 e 800 bp mentre BiPathCNN C gestisce le sequenze tra 800 e 1.200 bp. Inoltre, le sequenze di lunghezza compresa tra 1.200 bp e 5.000 bp vengono gestite in modo diverso: invece di utilizzare una rete neurale specifica, si ricorre all'utilizzo delle reti BiPathCNN A, B e C per predire le sottosequenze all'interno di finestre di lunghezza 1.200 bp. Infine, si calcola una media ponderata delle predizioni delle finestre per ottenere il punteggio finale per l'intera sequenza.

2.4 Virtifier

Virtifier [2] è un identificatore virale basato sul deep learning che si distingue dai metodi precedentemente analizzati per l'utilizzo di un processo di codifica delle sequenze, chiamato Seq2Vec, e per l'impiego di una rete LSTM basata sull'attenzione (Figura 2.4).

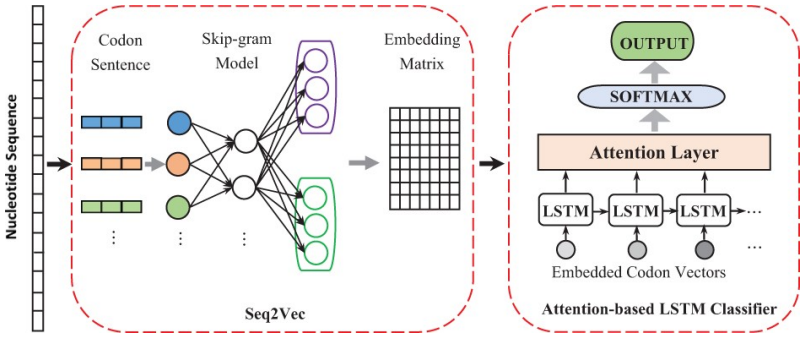


Figura 2.4: Struttura di Virtifier

Analizziamo entrambi i componenti nel dettaglio.

2.4.1 Seq2Vec

Seq2Vec è un metodo utilizzato per codificare sequenze di nucleotidi tramite un dizionario di codoni e una matrice di embedding. A differenza del solo utilizzo della codifica one-hot, che tratta ogni base in modo indipendente l'una dall'altra, Seq2Vec permette di esprimere una sequenza tramite diverse dimensioni significative in cui ciascuna di esse, invece di contenere valori binari, contiene un numero reale in grado di tenere conto del contesto sequenziale delle informazioni.

Addestramento matrice

Inizialmente la sequenza di nucleotidi presa in esame viene convertita in stringhe di codoni con una finestra di avanzamento pari a uno. Questi codoni vengono prima di tutto mappati all'interno di un dizionario contenente tutte le 64 possibili combinazioni univoche di codoni, per poi essere rappresentati attraverso vettori one-hot. Successivamente i vettori verranno moltiplicati da sinistra per una matrice di embedding.

Questa matrice viene addestrata utilizzando l'algoritmo Skip-Gram con lo scopo di rappresentare ogni codone come un vettore continuo di valori reali. Lo Skip-Gram è un tipo di rete neurale che predice la probabilità di una parola in base al contesto delle parole circostanti. In questo caso la parola che vogliamo predire è il codone di input centrale e il contesto sono i codoni circostanti nella sequenza.

L'obiettivo è infatti quello di convertire ogni codone in una rappresentazione numerica che tenga conto del suo significato biologico e della sua relazione con gli altri codoni nella sequenza.

Per addestrare l'embedding matrix viene prima di tutto convertita la sequenza in un formato accettabile per lo SG. Il processo è illustrato nella Tabella 2.1 in cui la prima colonna contiene tutti i codoni di input mentre i codoni di contesto corrispondenti sono nella seconda colonna. Inoltre, sempre all'interno della tabella, la matrice a destra contiene la forma mappata e codificata in vettori one-hot dell'input.

Input	Output		ATA	GCC	TGA	AAG	CTT	GCA	TTG
ATA	GCC	Datapoint1	1	0	0	0	0	0	0
GCC	ATA	Datapoint2	0	1	0	0	0	0	0
TGA	GCC	Datapoint3	0	0	1	0	0	0	0
TGA	AAG	Datapoint4	0	0	1	0	0	0	0
AAG	TGA	Datapoint5	0	0	0	1	0	0	0
AAG	CTT	Datapoint6	0	0	0	1	0	0	0
CTT	AAG	Datapoint7	0	0	0	0	1	0	0
CTT	GGA	Datapoint8	0	0	0	0	1	0	0
GGA	CTT	Datapoint9	0	0	0	0	0	1	0
GGA	TTG	Datapoint10	0	0	0	0	0	1	0

Tabella 2.1: Struttura del set di dati di addestramento per l'incorporamento dei codoni

Il modello SG viene in seguito addestrato per mappare ogni vettore one-hot in un vettore di attivazione nascosto. Questo viene poi moltiplicato per i pesi tra il livello di input e quello nascosto per ottenere le attivazioni nascoste, che vengono poi a loro volta moltiplicate per i pesi tra il livello nascosto e quello di output per calcolare le uscite finali del modello. Ciò è osservabile nella Figura 2.5, in cui V rappresenta la dimensione di input, N è la dimensione di embedding, mentre C è il numero di codoni di contesto.

Alla fine dell'addestramento, i pesi tra il livello nascosto e quello di output vengono considerati come i vettori di rappresentazione dei codoni, cioè la matrice di embedding di valori reali.

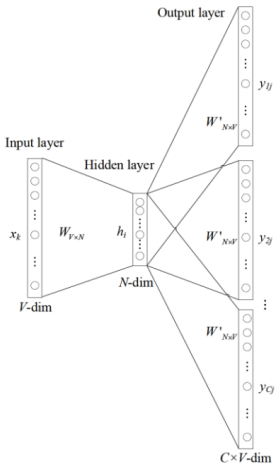


Figura 2.5: Modello Skip-Gram

Il numero delle colonne della matrice di embedding è stata impostata a 20, ovvero il numero totale di amminoacidi, poiché è stato empiricamente dimostrato aumentare le performance. La matrice, di conseguenza, avrà una dimensione di 64x20.

2.4.2 LSTM

In seguito viene utilizzato un classificatore LSTM (Long Short-Term Memory) basato sull'attenzione con lo scopo di considerare il contesto e le dipendenze tra le informazioni, consentendo una migliore comprensione delle sequenze in esame.

Memoria a breve termine

LSTM utilizza una tecnica chiamata "gate" che consente di controllare quali informazioni devono essere memorizzate o dimenticate dalla memoria attraverso un meccanismo formato da un livello sigmoidale e un'operazione di moltiplicazione punto per punto.

Dopo aver elaborato la sequenza di codoni tramite l'embedding matrix, ogni codone, rappresentato come un vettore di embedding di dimensione 20, viene immesso in successione in una cella LSTM, seguendo l'ordine dei codoni della sequenza.

Le celle LSTM consentono di far passare le informazioni importanti tra i diversi passi temporali riducendo gli effetti della memoria a breve termine. Esse sono composta da tre unità moltiplicative, come si può vedere in Figura 2.6: porta di input, porta di output e porta forget. Le porte controllano il flusso delle informazioni nella cella di memoria determinando quali informazioni mantenere.

Meccanismo di attenzione

Tuttavia, quando un input viene codificato tramite LSTM in un vettore di stato di lunghezza fissa, ogni elemento condividerà lo stesso peso e ciò potrebbe portare a una perdita di informazioni importanti poiché, in casi reali, gli elementi potrebbero avere pesi diversi a seconda del contesto in cui si trovano. Per risolvere questo problema viene utilizzato

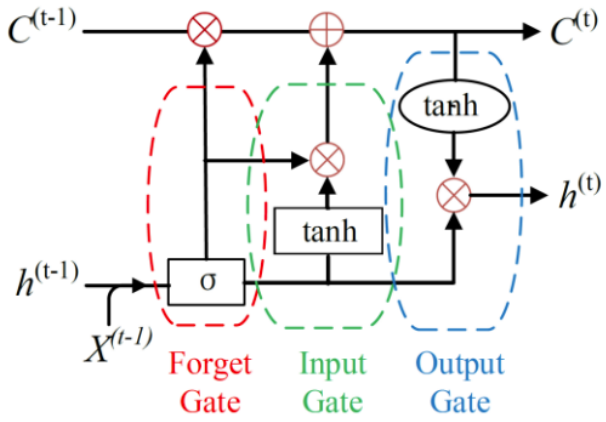


Figura 2.6: Una cella di memoria del LSTM

un meccanismo di attenzione Figura 2.7 che codifica i vettori di stato generati dalle celle LSTM assegnando un punteggio di importanza per ogni elemento ricevuto, con lo scopo quindi di dare maggiore importanza a determinati elementi della sequenza.

Esaminiamo nel dettaglio il meccanismo di attenzione e l'output finale:

Per prima cosa viene calcolato un vettore di contesto m_t utilizzando la seguente formula:

$$m_t = \tanh\left(W_a h_t + b\right)$$

Dove h_t è l'elemento corrente dell'input mentre W_a e b sono i pesi e i bias che contengono le informazioni di contesto. Viene in seguito calcolato il punteggio di attenzione normalizzato α_t utilizzando il vettore di contesto m_t e la matrice di pesi W_m attraverso uno strato softmax. Il punteggio di attenzione α_t rappresenta l'importanza relativa di ogni elemento dell'input:

$$\alpha_t = \frac{\exp\left(m_t^T W_m\right)}{\sum_t \exp\left(m_t^T W_m\right)}$$

Infine si calcola il vettore di contesto ponderato s come media pesata degli elementi dell'input utilizzando i punteggi di attenzione α_t.

$$s = \sum_t \alpha_t h_t$$

Questo vettore di contesto influisce sull'output finale del modello, consentendo di concentrarsi sui segmenti di sequenza più rilevanti per la classificazione delle sequenze virali. Il processo del meccanismo di attenzione appena descritto è riassunto nella Figura 2.7.

Infine, i punteggi di attenzione ottenuti e i punteggi generati dall'LSTM vengono utilizzati per determinare la predizione finale. I punteggi più alti indicano una maggiore probabilità che la sequenza di query sia virale, mentre i punteggi più bassi indicano una maggiore probabilità che la sequenza non sia virale.

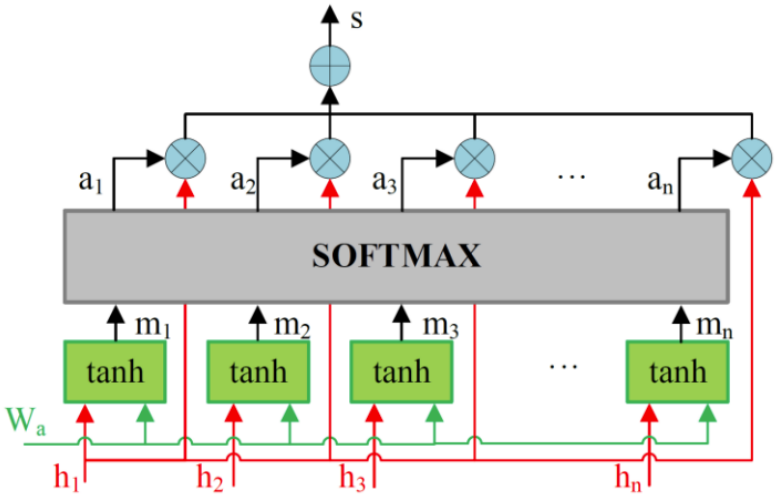


Figura 2.7: Struttura del meccanismo di attenzione

2.5 Detire

Detire [16] è un modello ibrido di deep learning costituito da un incorporatore di sequenze basato su grafi convoluzionali (GCN) e un modello di deep learning a due percorsi (Figura 2.8). Esso si distingue dai metodi precedentemente analizzati in quanto è il primo modello che fa uso sia di una rete ricorrente, BiLSTM, sia di una rete convoluzionale che, insieme, permettono di apprendere contemporaneamente le caratteristiche spaziali e sequenziali al fine di generare una vasta gamma di caratteristiche per le sequenze corte.

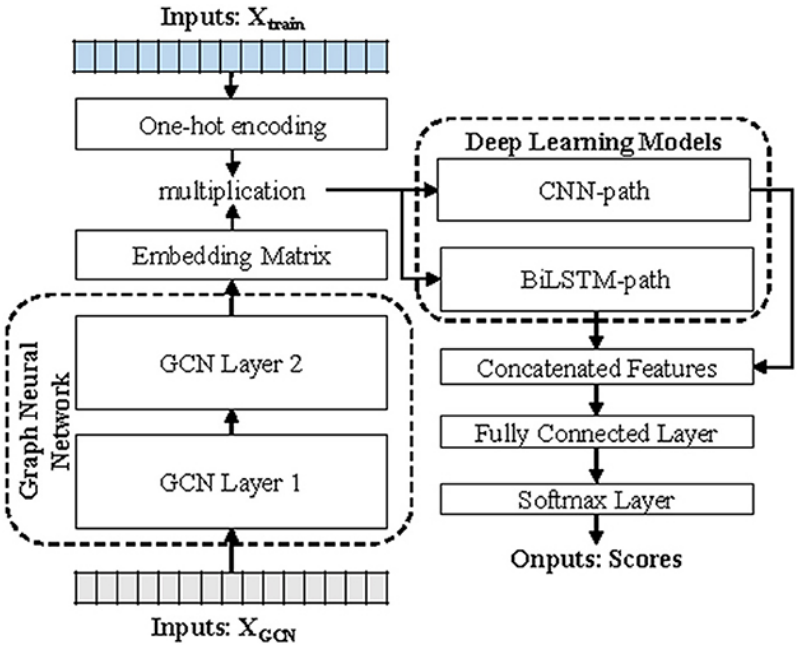


Figura 2.8: Struttura di Detire

2.5.1 Struttura GCN

Inizialmente, il metodo prevede che ogni sequenza di nucleotidi venga convertita in un insieme di k-mer, con k=3 e una finestra di avanzamento pari a uno. Successivamente,

un modello di rete neurale a grafo, denominato TextGCN, viene utilizzato per ottenere rappresentazioni significative e di alto livello di tutti i frammenti delle sequenze nucleotidiche. Questi frammenti vengono in questo modo organizzati in un grafo eterogeneo, in cui i nodi rappresentano i k-mer e gli archi catturano le co-occorrenze globali tra di essi (Figura 2.9). Il grafo costruito viene poi utilizzato in un GCN a due strati per apprendere le rappresentazioni dei frammenti e delle sequenze.

L'architettura del primo strato comprende la costruzione dei nodi e degli archi del grafo, in cui i pesi tra i nodi e gli archi vengono calcolati utilizzando la tecnica Term Frequency-Inverse Document Frequency (TF-IDF) [20] per riflettere l'importanza dei frammenti nelle sequenze. Successivamente viene utilizzata la Point-wise mutual information [21] al fine di calcolare i pesi tra tutte le coppie di nodi.

Nel secondo strato, vengono in seguito apprese le rappresentazioni dei frammenti e delle sequenze di ciascun nodo, che vengono utilizzate per addestrare un classificatore softmax, utilizzando come funzione di costo l'errore di entropia incrociata.

La struttura del GCN è illustrata in Figura 2.9, in cui ogni sequenza di nucleotidi nel dataset di addestramento e tutti i k-mers derivati da essa sono rappresentati come nodi individuali del grafo. Come si può vedere, non esistono collegamenti diretti tra tutte le diverse sequenze di nucleotidi, mentre gli archi sono costruiti tra i k-mers e le sequenze originali da cui provengono. Dopo la fase di addestramento, i vettori dei nodi corrispondenti ai codoni nella seconda fase del GCN vengono combinati in una matrice di embedding.

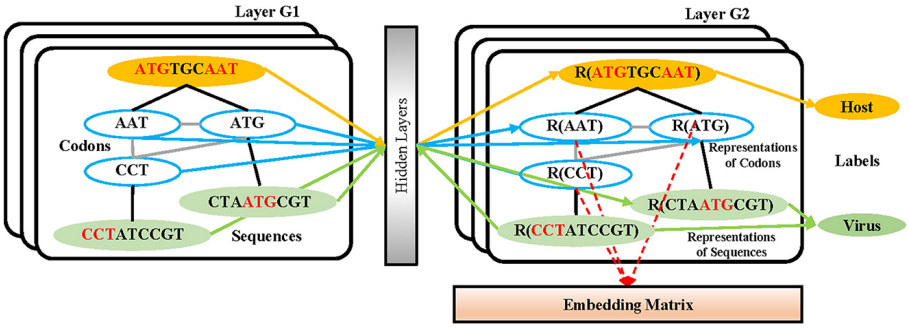


Figura 2.9: Struttura del GCN

2.5.2 Fase di classificazione (CNN + RNN)

Nella fase di classificazione delle sequenze, come accennato in precedenza, Detire utilizza due modelli di deep learning in parallelo: una rete neurale convoluzionale (CNN) e una rete neurale ricorrente bidirezionale (BiLSTM).

Nel percorso della CNN, le sequenze incorporate sono trattate come immagini con lo scopo di estrarre le caratteristiche spaziali attraverso tre set di layer convoluzionali con diverse dimensioni di filtro e un max pooling, seguiti da una funzione di normalizzazione batch e una funzione di attivazione ReLU.

Nel percorso BiLSTM, i k-mer incorporati sono invece inseriti nelle celle BiLSTM per ottenerne le caratteristiche sequenziali, similmente a come avviene il Virtifier.

I risultati dei due percorsi vengono infine combinati tramite strati densi per generare una rappresentazione complessiva delle sequenze. Lo strato softmax finale produce due punteggi che indicano la probabilità che la sequenza in input sia un virus o meno.

Capitolo 3

Prestazioni e confronto dei metodi

In questo capitolo verranno esaminate le prestazioni dei metodi precedentemente analizzati, concentrandosi sulla metodologia di addestramento della rete neurale, sulla lunghezza delle sequenze utilizzate nella fase di test e sui criteri di valutazione specifici del singolo metodo. Successivamente, i risultati ottenuti saranno confrontati col fine di identificare il metodo in grado di fornire le prestazioni più accurate e affidabili nella rilevazione di sequenze virali.

3.1 AUROC

Per effettuare questo confronto è fondamentale definire prima di tutto i criteri per valutare le performance dei classificatori.

Un metodo ampiamente adottato consiste nell'utilizzare la matrice di confusione, che fornisce diverse metriche utili per valutare la capacità di classificare correttamente gli esempi positivi e negativi. La matrice di confusione è composta da quattro statistiche fondamentali: i veri positivi (TP), i falsi positivi (FP), i veri negativi (TN) e i falsi negativi (FN).

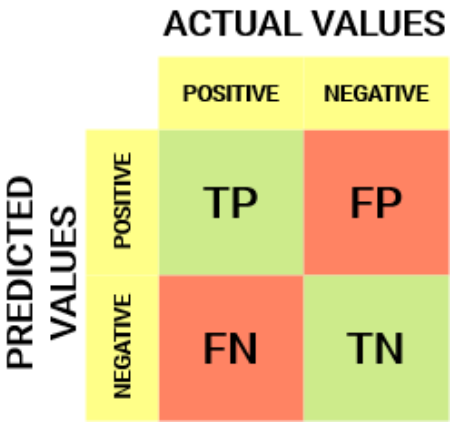


Figura 3.1: Understanding and Interpreting Confusion Matrices for Machine Learning - Aniruddha Bhandari

A partire da questi valori, possiamo derivare il true positive rate (TPR) e il false positive rate (FPR) che vengono utilizzati per costruire la curva ROC, ovvero una rappresentazione grafica delle prestazioni del modello, in cui il tasso dei falsi positivi è rappresentato sull'asse delle ascisse e il tasso dei veri positivi sull'asse delle ordinate.

L'area sotto la curva ROC, ovvero l'AUROC è invece un indicatore della performance di predizione del modello di predizione, dove un valore più alto indica una migliore performance del modello. Ad esempio un valore di AUROC pari a 1 indica un classificatore perfetto in grado di distinguere correttamente tutte le classi, mentre un valore di AUROC pari a 0,5 indica un classificatore che effettua predizioni casuali e non è in grado di distinguere le classi. Allo stesso modo, un valore di AUROC maggiore di 0,8 indica un classificatore con ottime prestazioni.

Nel caso in cui il dataset sia altamente sbilanciato, contenendo ad esempio un numero sbilanciato di virus rispetto ai procarioti, le curve precision-recall (PR) forniscono una visione più informativa delle prestazioni rispetto alle curve ROC. Pertanto, per valutare le prestazioni in questo contesto, verrà utilizzata l'area sotto la curva PR (AUPRC).

3.2 Analisi delle prestazioni dei metodi

3.2.1 VirFinder

VirFinder è stato costruito e testato attraverso l'impiego di due differenti set di virus e procarioti: nel processo di addestramento, sono stati utilizzati geni sequenziati prima del 1° Gennaio 2014, mentre per la fase di testing sono state utilizzate un numero uguale di sequenze virali e procariote ottenute dopo questa data. Al fine di simulare sequenze metagenomiche frammentate, i genomi virali provenienti da RefSeq sono stati divisi in frammenti di varie lunghezze (Tabella 3.1)

Fragment length (*L*)	Before 1 January 2014	After 1 January 2014	Total
500 bp	154,640	50,350	204,990
1000 bp	77,014	25,087	102,101
3000 bp	25,263	8246	33,509
5000 bp	14,881	4878	19,759
10000 bp	7120	2357	9477

Tabella 3.1

Parametri utilizzati

Dai risultati degli studi condotti è emerso che il punteggio AUROC, e quindi le prestazioni, migliorano all'aumentare della lunghezza dei k-mer utilizzati nel processo di classificazione. Nello specifico, come illustrato in Figura 3.2A, le prestazioni risultano relativamente stabili per frammenti di lunghezza ≥ 3.000 bp impostando a 6 la lunghezza dei k-mer. Per frammenti superiori a 1.000 bp i valori dell'AUROC si stabilizzano per k-mer di lunghezza ≥ 8, mentre per frammenti di 500 bp le prestazioni aumentano ancora leggermente per

lunghezza dei k-mer superiori a 8. Sulla base di questi risultati, è stata scelta per tutte le applicazioni dello strumento una lunghezza dei k-mer pari a 8. I risultati ottenuti sono illustrati in dettaglio in Figura 3.2B.

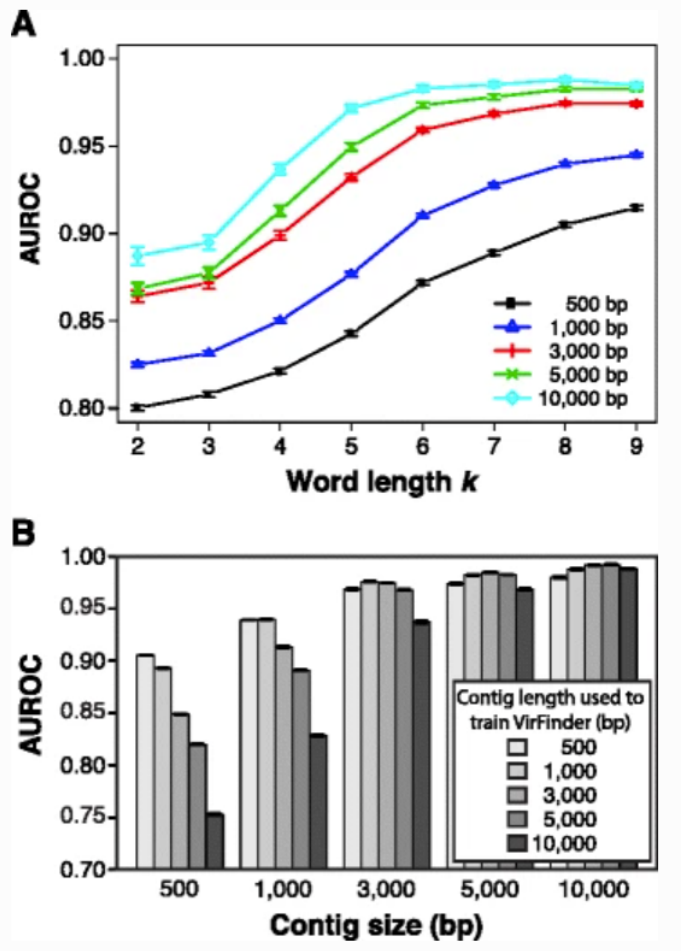


Figura 3.2

3.2.2 DeepVirFinder

Addestramento e selezione dei parametri

L'affidabilità e le prestazioni di DeepVirFinder sono determinate da due parametri chiave delle reti neurali convoluzionali: il numero e la lunghezza dei motivi (filtri).Per identificare le migliori impostazioni di tali parametri il modello è stato addestrato su diverse loro combinazioni basandosi su sequenze ottenute prima del 1°Maggio 2015.

I risultati ottenuti hanno rivelato che, aumentando la lunghezza dei motivi, l'AUROC di convalida aumentava rapidamente raggiungendo il valore più alto con una lunghezza dei motivi pari a 10, rimanendo sostanzialmente stabile con ulteriori incrementi (vedi Figura 3.3A, curve rosse).

Successivamente, fissata la lunghezza dei motivi a 10, è stata valutata l'influenza del numero di motivi sulle prestazioni del modello, variandone il numero da 100 a 1.500. Dai test effettuati si è notato che l'AUROC aumentava gradualmente con il numero di motivi

(Figura 3.3B) e perciò è stato scelto di utilizzare 1000 motivi nel livello convoluzionale e 1000 neuroni nel livello denso nel modello finale.

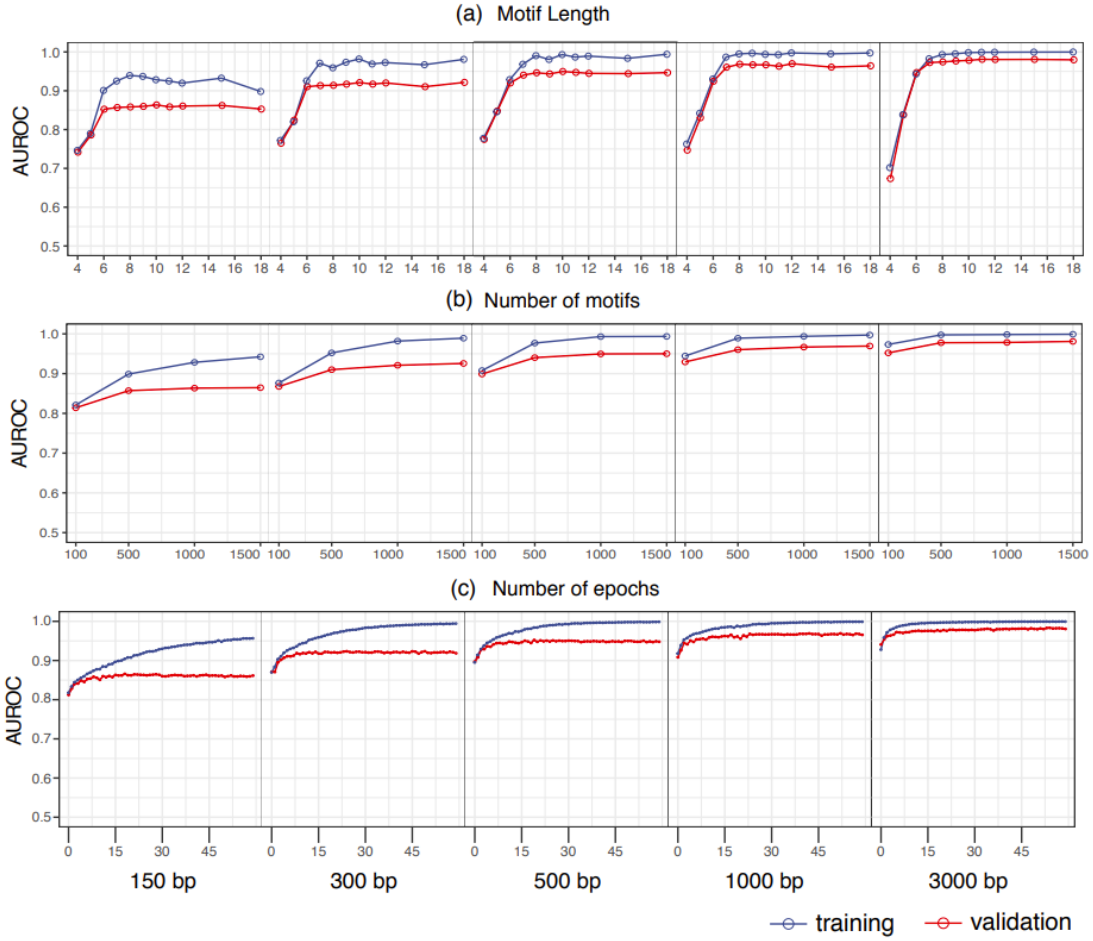


Figura 3.3

Prestazioni e testing

Una volta determinati i parametri ottimali, le prestazioni del modello sono state valutate su sequenze ottenute dopo Maggio 2015 per ottenere una valutazione imparziale della performance del modello indipendente da tutto il processo di addestramento, convalida e taratura dei parametri.

È interessante notare che il modello addestrato con sequenze di 150 bp ha ottenuto il punteggio AUROC più alto nella predizione di sequenze di 150 bp. In modo analogo, come si vede in Figura 3.4B, il modello addestrato con sequenze di 300 bp ha dimostrato le migliori prestazioni nella previsione delle sequenze di 300 bp, e lo stesso vale per le sequenze di 500 e 1000 bp.

Inoltre è stato osservato in linea generale un aumento del punteggio AUROC all'aumentare della lunghezza dei frammenti. Ciò si può sempre osservare in Figura 3.4B in cui i punteggi AUROC per frammenti con lunghezza inferiore a 300 bp, compresa tra 300 e 500 bp, tra 500 e 1000 bp e superiore a 1000 bp sono stati rispettivamente 0.8317, 0.8767, 0.8966 e 0.9451.

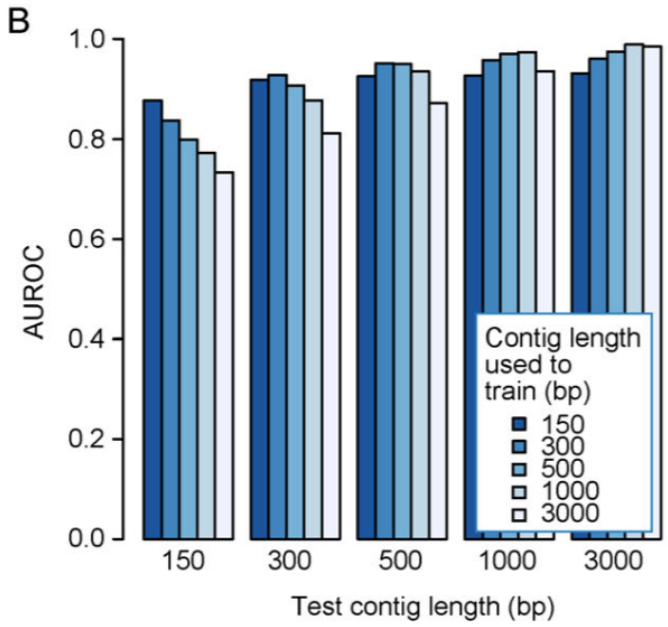


Figura 3.4

3.2.3 PPR-META

Suddivisione in gruppi di test

Le performance complessive di PPR-Meta sono state valutate su 4 gruppi di set di test, ciascuno costituito da frammenti di diversa lunghezza. In particolare in Figura 3.5 sono rappresentate quattro matrici di confusione a tre classi che indicano le prestazioni in base al gruppo di sequenze analizzato (Gruppi A-D). L'algoritmo calcola 3 punteggi per ogni frammento di input, ciascuno dei quali rappresenta la probabilità che il frammento appartenga rispettivamente a un fago, a un plasmide o a un cromosoma, selezionando per la predizione la categoria col punteggio più alto. Come è accaduto coi metodi precedenti, anche PPR-Meta ha dimostrato una migliore capacità di discriminazione con sequenze più lunghe. È stato inoltre verificato che la sua capacità di riconoscimento dei fagi è migliore rispetto ai plasmidi, spesso confusi coi cromosomi ospiti a causa della presenza di sequenze tra loro condivise.

Prestazioni BiPathCNN

Uno degli elementi fondamentali che ha notevolmente influenzato le prestazioni di PPR-Meta è l'utilizzo del BiPathCNN, che sfrutta sia informazioni sulla sequenza delle basi sia sulla sequenza dei codoni. Attraverso l'esperimento di rimuovere uno dei due percorsi e successivamente riaddestrare PPR-Meta, è stato dimostrato che l'uso combinato di entrambi i percorsi tramite BiPathCNN offre prestazioni nettamente superiori rispetto all'utilizzo di un solo percorso. Questa superiorità delle prestazioni di BiPathCNN è confermata dai risultati riportati nella Tabella 3.2, dove, nonostante la performance dei singoli percorsi possa variare a seconda del gruppo di test utilizzato, le prestazioni generali di BiPathCNN risultano costantemente migliori.

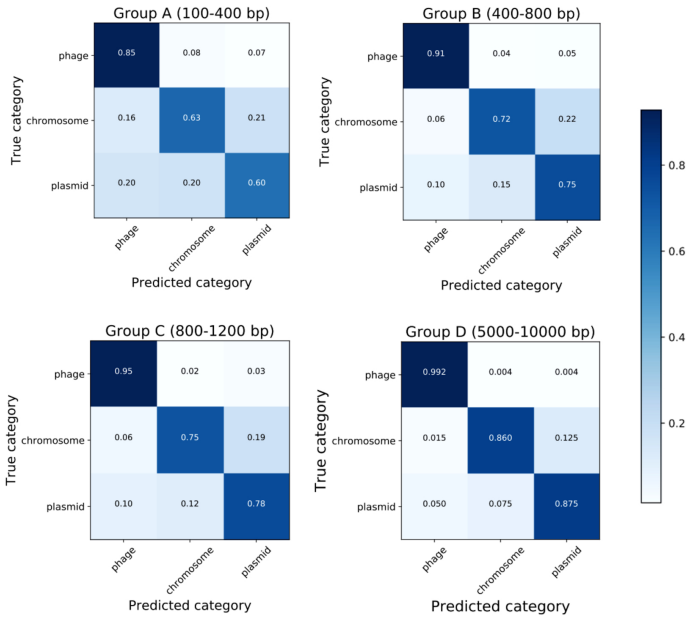


Figura 3.5: Matrice di confusione di PPR-Meta

Group	Tool	Evaluation on phage (%)			Evaluation on plasmid (%)		
		TPR	FPR	AUC	TPR	FPR	AUC
Group A 100–400 bp	BiPathCNN	84.96	18.01	91.82	59.91	14.14	83.05
	Base path–only	81.86	24.58	87.50	56.96	17.60	78.50
	Codon path–only	86.84	20.47	91.85	62.15	16.65	82.26
Group B 400–800 bp	BiPathCNN	90.75	8.37	97.21	74.56	13.37	89.64
	Base path–only	88.76	17.46	93.87	72.37	18.86	85.57
	Codon path–only	84.95	5.93	96.57	82.98	23.10	88.32
Group C 800–1,200 bp	BiPathCNN	95.24	7.75	98.54	78.09	10.95	91.84
	Base path–only	92.09	17.71	95.47	73.31	15.12	88.02
	Codon path–only	94.60	12.44	97.55	73.17	12.41	89.22

Tabella 3.2

3.2.4 Virtifier

Virtifier si differenzia dai metodi precedenti per le sue prestazioni ottimali su sequenze brevi, risultando tuttavia meno efficace nell'analisi di sequenze superiori a 500 bp.

Come nei metodi precedenti, anche in Virtifier è stato evidenziato che le sequenze di test ottengono un punteggio AUROC più elevato quando la loro lunghezza corrisponde a quella utilizzata per l'addestramento del modello, come illustrato nella Figura 3.6.

Tuttavia, con l'impiego del modello LSTM, è stata verificata una drastica diminuzione delle prestazioni quando la lunghezza delle sequenze utilizzate durante la fase di allenamento è maggiore della lunghezza delle sequenze di query. Per risolvere questa problematica, sono stati addestrati due modelli separati, utilizzando sequenze di lunghezza pari a 300

bp per prevedere sequenze più corte di 300 bp e sequenze di lunghezza pari a 500 bp per prevedere sequenze comprese tra 300 e 500 bp.

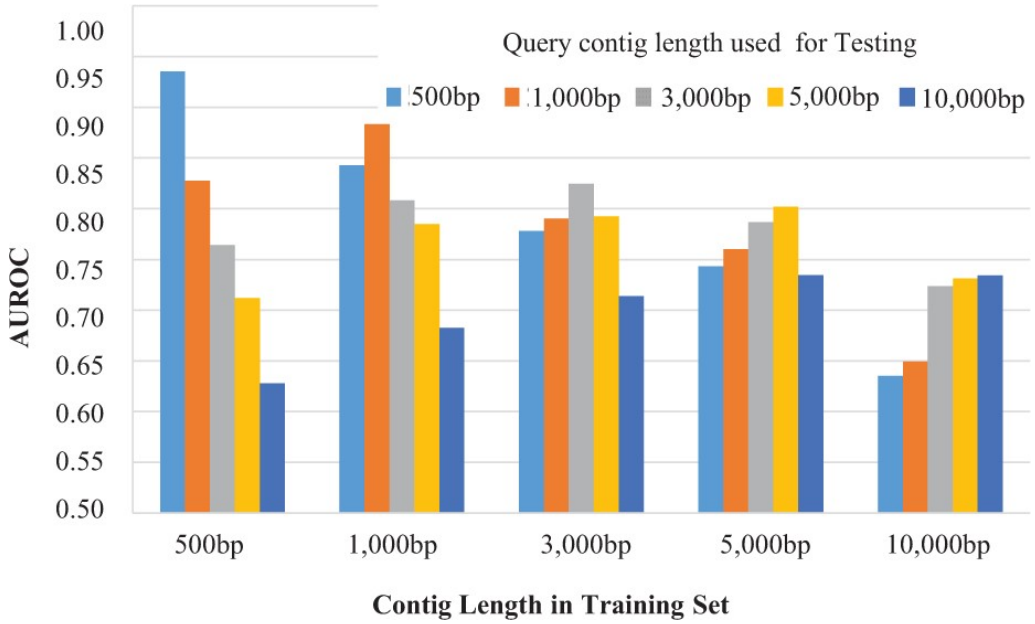


Figura 3.6

Influenza del LSTM nell'identificazione di sequenze lunghe

Grazie all'impiego della LSTM, Virtifier si dimostra altamente preciso nell'identificazione di sequenze brevi. Questa caratteristica comporta tuttavia una significativa riduzione delle prestazioni quando si valutano sequenze più lunghe di 1000 bp, indipendentemente dalla lunghezza delle sequenze utilizzate per l'addestramento e il test del modello. Al fine di mantenere elevate le prestazioni e prevenire problemi di memoria, si è quindi deciso di ottimizzare ulteriormente Virtifier per focalizzarsi sull'identificazione di virus con lunghezze inferiori a 500 bp.

Nel caso in cui venga identificata una sequenza di query più lunga di 500 bp, Virtifier adotta un approccio di suddivisione in diverse sottosequenze non sovrapposte di 500 bp. Questo permette al modello di gestire sequenze più lunghe in modo efficiente e di mantenere la sua capacità di identificazione precisa.

Effetti della dimensione di embedding e delle parole

Successivamente, è stata condotta un'analisi per determinare il numero ottimale di k, ovvero la lunghezza dei k-mer in cui dividere la sequenza di input. Per fare ciò sono stati creati dizionari di lunghezza 16, 64, 256 e 1024 suddividendo ogni sequenza con lunghezza di 500 bp in parole con due, tre, quattro e cinque basi, rispettivamente.

Allo stesso tempo sono state utilizzate diverse dimensioni di embedding per confrontare le performance dei dizionari. Per ciascuno di questi valori è stato creato e addestrato un modello da cui ricavare le prestazioni tramite l'AUROC.

Dai risultati ottenuti è emerso che una lunghezza di tre, rappresentante un codone, e una dimensione di embedding di 20,ovvero il numero di amminoacidi, si sono rivelate ottime per le prestazioni di Virtifier. Questi risultati sono presentati nella Figura 3.7.

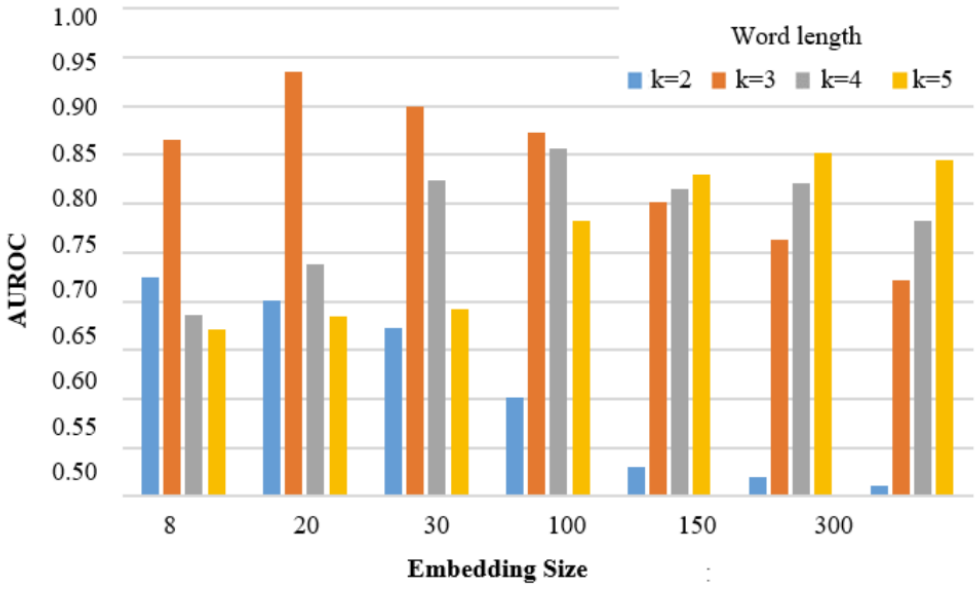


Figura 3.7

3.2.5 Detire

Allo stesso modo di come accade in Virtifier, i due parametri fondamentali che influenzano le prestazioni del metodo sono la dimensione delle sottosequenze di lunghezza k e la dimensione di embedding. Per trovare una combinazione ottimale di questi due parametri, sono stati addestrati numerosi modelli facendo uso di un dataset di addestramento e un dataset di test. Ogni modello corrisponde a un set specifico di parametri, con k scelto tra 1 e 10, e la dimensione di embedding scelta tra 4 e 1.000. I risultati di valutazione vengono mostrati nella Figura 3.8., che riporta le prestazioni dei modelli con diverse combinazioni di k e dimensioni di embedding sul dataset di test, misurate in termini di valore AUROC.

Dai risultati ottenuti, emerge che quando la dimensione di embedding è appena inferiore a 4 volte il valore di k, l'AUROC raggiunge i valori più elevati. Di conseguenza, per il modello finale, sono stati scelti k-mer di dimensione 3 e una dimensione di embedding pari a 30, che dimostra prestazioni ottimali mantenendo allo stesso tempo contenuta la dimensione della matrice.

Inoltre, poiché il dataset di addestramento utilizzato per la rete neurale a grafo è costituito da sequenze di lunghezza fissa di 500 bp, è stato evidenziato che all'aumentare della lunghezza delle sequenze di input, il vantaggio derivante dall'utilizzo del metodo Detire diventa sempre meno evidente.

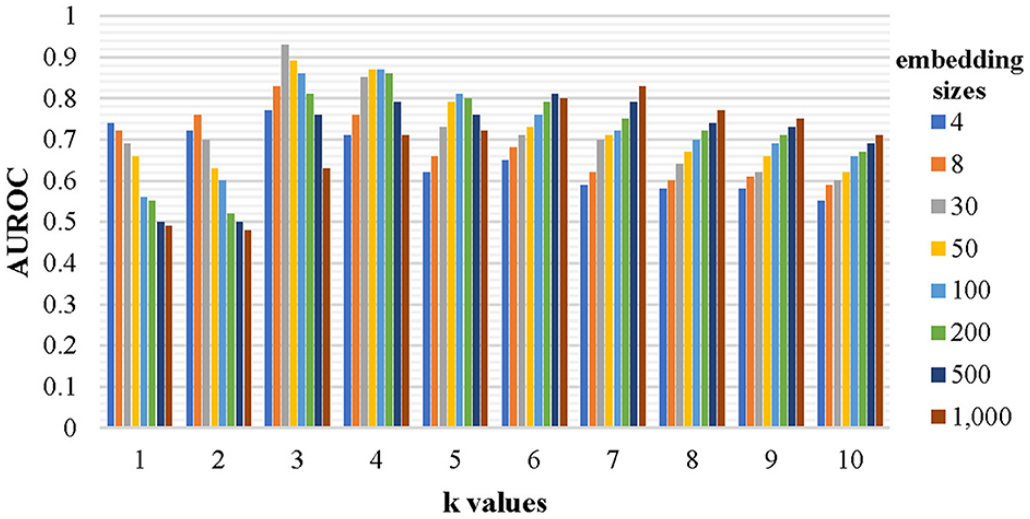


Figura 3.8

Di conseguenza, la matrice di embedding addestrata risulta più adatta per identificare sequenze virali brevi, riuscendo a identificare con buona precisione sequenze fino ai 1.000 bp.

3.3 Confronto tra le prestazioni dei metodi

3.3.1 Confronto VirFinder - DeepVirFinder

Durante la valutazione delle prestazioni di DeepVirFinder, i risultati ottenuti sono stati comparati con quelli di VirFinder evidenziando fin da subito la superiorità di DeepVirFinder rispetto al suo predecessore per tutte le lunghezze di sequenza analizzate.

Come evidenziato in Figura 3.9A, le curve ROC di DeepVirFinder hanno infatti costantemente superato quelle di VirFinder per tutte le lunghezze da 150 a 3.000 bp.

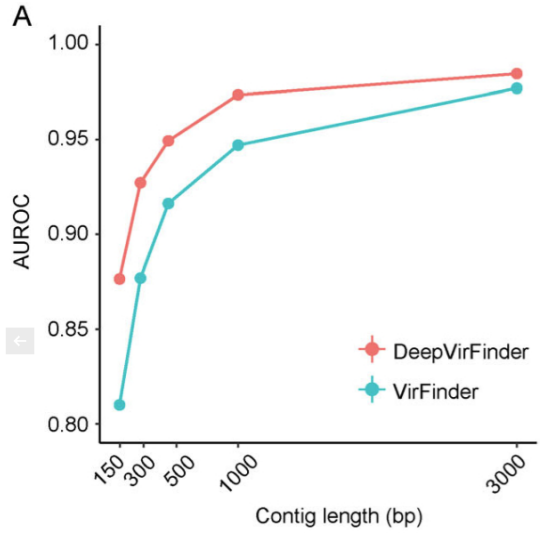


Figura 3.9

L'aumento dell'AUROC è stato particolarmente significativo per le sequenze brevi inferiori a 1000 bp. Ad esempio, come illustrato nella Tabella 3.3 ,per sequenze di 150, 300 e 500 bp, se VirFinder ha ottenuto un AUROC rispettivamente di 0.8101, 0.8771 e 0.9163, i punteggi AUROC di DeepVirFinder sono stati di 0.8766, 0.9272 e 0.9494, mostrando un miglioramento, in base alle diverse lunghezze, rispettivamente del 8.2%, 5.7% e 3.6%.

Per le sequenze di 1000 bp, DeepVirFinder ha invece incrementato l'AUROC da 0.9471 a 0.9735, mostrando un aumento del 2.8%. Inoltre per sequenze di 3000 bp, è stato mostrato un aumento, seppur minimo, da 0.977 a 0.9847.

Contig Length (bp)	VirFinder	DeepVirFinder
150	0.8101 (0.0007)	0.8766 (0.0007)
300	0.8771 (0.0012)	0.9272 (0.0005)
500	0.9163 (0.0012)	0.9494 (0.0011)
1000	0.9471 (0.0012)	0.9735 (0.0004)
3000	0.9770 (0.0008)	0.9847 (0.0009)

Tabella 3.3

I risultati mettono chiaramente in evidenza come le prestazioni di DeepVirFinder siano superiori al suo predecessore per tutte le sequenze analizzate, soprattutto nel caso di sequenze brevi.

3.3.2 Confronto Virtifier - VirFinder - DeepVirFinder - PPR Meta

Nel documento [2] è stato condotto un confronto tra i metodi Virtifier, VirFinder, DeepVirFinder e PPR-Meta, da cui provengono le tabelle e immagini mostrate in questo paragrafo. Per questo confronto sono stati utilizzate più tipologie e lunghezze di sequenze provenienti da diversi database. Di conseguenza l'analisi verrà suddivisa in base a questi elementi appena citati.

Prestazioni sui genomi di test dei virus con sequenze brevi

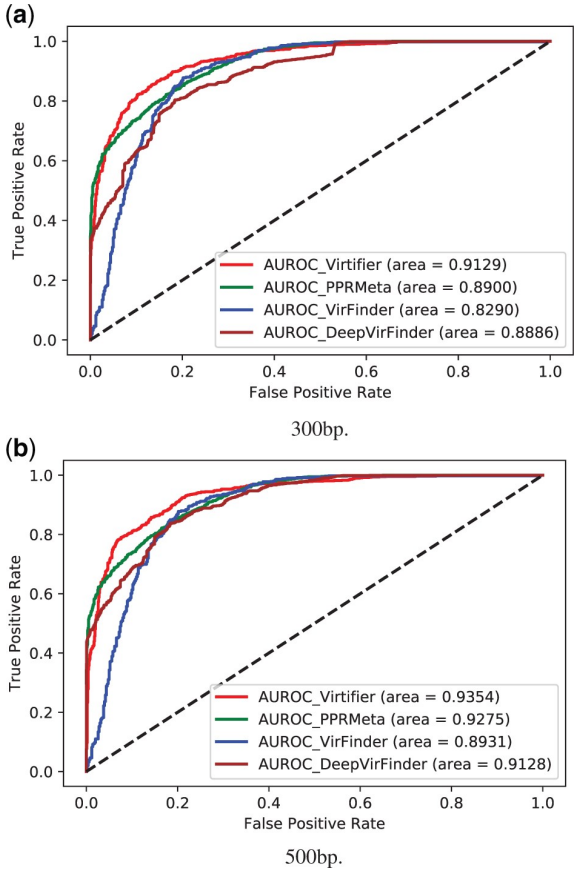
Inizialmente, i metodi sono stati valutati sulla base di due gruppi di dataset di prova contenenti sequenze corte rispettivamente di lunghezza 300 e 500 bp, riguardanti genomi di virus e ospiti provenienti dal database RefSeq.

Durante il confronto, Virtifier ha dimostrato risultati superiori rispetto agli altri tre metodi presi in esame, sia per le sequenze di 300 bp che per quelle di 500 bp. Ciò è evidenziato nella Tabella 3.4, dove i valori di TPR, precisione e punteggio F1 di Virtifier risultano maggiori rispetto a quelli degli altri metodi.

Criteria		VirFinder	DeepVirFinder	PPR-Meta	Virtifier
300bp	Recall	0.7740	0.9100	0.8220	**0.8720**
	Precision	0.7898	0.8198	0.8509	**0.8790**
	F1 Score	0.7818	0.8149	0.8362	**0.8755**
500bp	Recall	0.8420	0.8780	0.8960	**0.9260**
	Precision	0.8287	0.8833	0.9014	**0.9114**
	F1 Score	0.8353	0.8806	0.8987	**0.9186**

Tabella 3.4

Le prestazioni sono state ulteriormente valutate tramite le curve ROC, confermando che i punteggi AUROC ottenuti da Virtifier sono stati superiori per entrambi i gruppi di dati, come si vede in Figura 3.10.



300bp.

500bp.

Figura 3.10

Questi risultati dimostrano, oltre alle ottime prestazioni generali di Virtifier, anche la sua superiorità nell'identificazione di sequenze brevi rispetto ai metodi concorrenti.

Prestazioni in dataset sbilanciati

Successivamente, in questa sezione, i metodi sono stati testati utilizzando il database CAMI-high, caratterizzato da un dataset altamente sbilanciato costituito da 1097 sequenze, di cui 46 sequenze virali e 1051 sequenze ospiti.

Per questa ragione, ai fini del confronto, viene utilizzata la curva AUPCR. Anche in questo caso Virtifier ha dimostrato un netto miglioramento rispetto agli altri tre metodi, ottenendo un punteggio AUPRC significativamente più alto, seppur non ottimale (Figura 3.11).

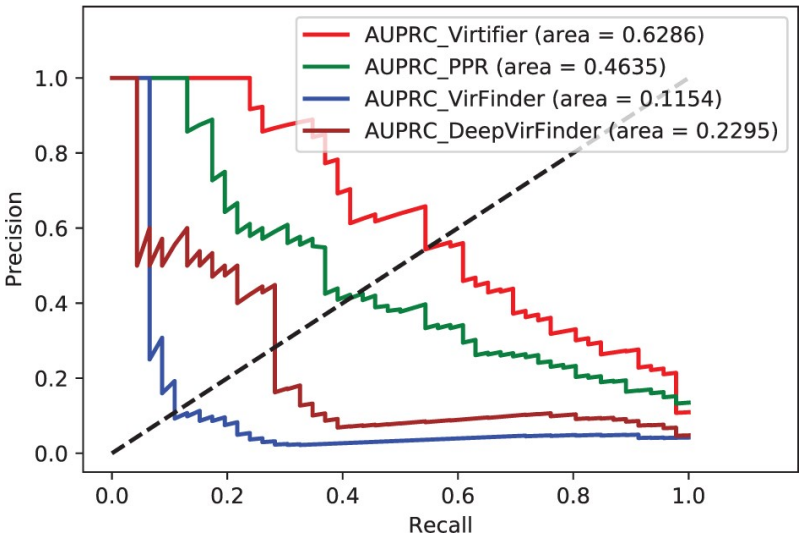


Figura 3.11

Inoltre, sono stati calcolati i valori di TPR, precisione e punteggio F1, che confermano anche in questo caso la superiorità di Virtifier, come mostrato nella Tabella 3.5.

Criteria	VirFinder	DeepVirFinder	PPR-Meta	Virtifier
Recall	0.3796	0.6522	0.8913	**0.9130**
Precision	0.0249	0.0943	0.2370	**0.2781**
F1 Score	0.0467	0.1648	0.2793	**0.3390**

Tabella 3.5

Prestazioni su un dataset metagenomico reale dell'intestino umano

In seguito i metodi sono stati valutati utilizzando un dataset metagenomico reale dell'intestino umano contenente sequenze di diversa lunghezza, incluse quelle più corte di 500 bp. Dopo aver costruito i grafici delle curve ROC dei quattro metodi (Figura 3.12), è stato osservato che quasi tutte le parti della curva di Virtifier si trovano al di sopra di quelle degli altri tre metodi.

Nel dettaglio Virtifier ha identificato più sequenze virali rispetto a VirFinder e DeepVirFinder per sequenze inferiori a 500 bp dimostrando una miglior precisione. Inoltre, per sequenze virali con lunghezza compresa tra 100 e 299 bp PPR-Meta ha ottenuto, seppur di poco, punteggi AUROC superiori rispetto a Virtifier, che presenta, tuttavia, prestazioni molto migliori per lunghezze tra 300 e 500 bp come osservabile nel dettaglio in Figura 3.13.

Inoltre i punteggi di richiamo, precisione e F1 di Virtifier sono migliori rispetto agli altri tre metodi, come si vede nella Tabella 3.6.

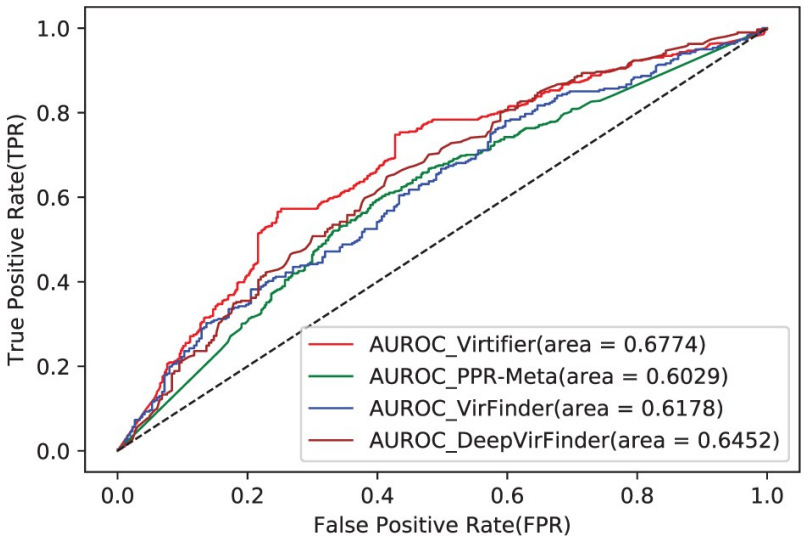


Figura 3.12

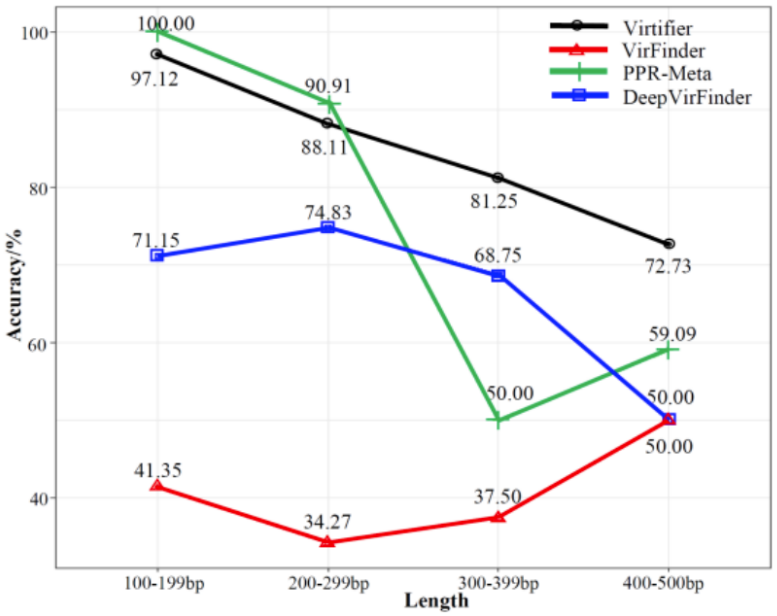


Figura 3.13

Prestazioni per sequenze più lunghe di 5.000 bp

Lo stesso dataset metagenomico dell'intestino umano è stato inoltre utilizzato per l'identificazione di sequenze virali più lunghe di 5.000 bp.

I risultati della valutazione mostrano che Virtifier ha dimostrato prestazioni superiori rispetto a VirFinder e DeepVirFinder nell'identificazione di queste sequenze virali lunghe.

In questo caso, tuttavia, tra i quattro metodi, PPR-Meta ha ottenuto il maggior numero di sequenze virali correttamente identificate, ovvero 1.096 contro le 1.035 di Virtifier, presentando, allo stesso tempo, anche i punteggi più alti per le metriche di valutazione come il TPR, la precisione e il punteggio F1, come si vede in Tabella 3.7.

Il motivo per cui Virtifier ha dimostrato di avere un'efficacia leggermente inferiore nella

Criteria	VirFinder	DeepVirFinder	PPR-Meta	Virtifier
Recall	0.6179	0.7110	0.8738	**0.8937**
Precision	0.6480	0.7589	0.9164	**0.9276**
F1 Score	0.6326	0.7342	0.8946	**0.9103**

Tabella 3.6

Criteria	VirFinder	DeepVirFinder	PPR-Meta	Virtifier
Num.	981	1,019	**1,096**	1,035
Recall	0.7724	0.8024	**0.8630**	0.8150
Precision	0.7437	0.7814	**0.8424**	0.8333
F1 Score	0.7578	0.7949	**0.8526**	0.8240

Tabella 3.7

classificazione di sequenze lunghe, è probabilmente attribuito al processo di segmentazione di queste sequenze, durante il quale alcune regioni utili che contribuiscono in modo significativo alle caratteristiche dell'intera sequenza potrebbero essere separate, portando alla perdita di informazioni importanti per l'identificazione virale.

3.3.3 Confronto Detire - DeepVirFinder - PPR Meta

Nello studio [16] sono stati confrontati i risultati ottenuti dai metodi Detire, DeepVirFinder e PPR-Meta. Anche in questo caso sono stati valutati i risultati su diversi set di dati, tra cui un dataset di test, il metagenoma marino CAMI e un metagenoma umano reale del tratto intestinale, lo stesso utilizzato da Virtifier.

Si osservi che nelle figure e tabelle di questo paragrafo è incluso anche un metodo non trattato in questo documento (CHEER) che non verrà quindi analizzato nel dettaglio.

Prestazioni sul set di dati di test

Nel set di test, Detire ha dimostrato ottime prestazioni nell'identificazione di sequenze virali corte con una lunghezza inferiore a 500 bp superando gli altri sia in termini di accuratezza, TPR, precisione che punteggio F1 (Tabella 3.8).

Prestazioni sul metagenoma marino CAMI

Anche nel metagenoma marino CAMI, Detire ha ottenuto risultati migliori rispetto a DeepVirFinder e PPR-Meta per l'identificazione di sequenze virali di tutte le lunghezze (Tabella 3.9), mostrando inoltre prestazioni comparabili al metodo CHEER.

Criteria	Deep-VirFinder	PPR-Meta	CHEER	DETIRE
Accuracy	0.8126	0.8403	0.8744	**0.8772**
Recall	0.8159	0.8396	0.8782	**0.8812**
Precision	0.8105	0.8408	0.8715	**0.8741**
F1 score	0.8132	0.8402	0.8748	**0.8776**

Bold values represent the best performance.

Tabella 3.8

Length	Criteria	Deep-VirFinder	PPR-Meta	CHEER	DETIRE
<500 bp	Accuracy	0.7840	0.7982	0.7982	**0.8061**
	Recall	0.7860	0.8009	0.8041	**0.8109**
	Precision	0.7828	0.7967	0.7948	**0.8032**
	F1 score	0.7844	0.7988	0.7994	**0.8071**
500–1,000 bp	Accuracy	0.8130	0.8435	0.9007	**0.9030**
	Recall	0.8074	0.8369	0.9001	**0.9040**
	Precision	0.8165	0.8481	0.9012	**0.9021**
	F1 score	0.8119	0.8425	0.9007	**0.9031**
1,000–3,000 bp	Accuracy	0.8269	0.8400	0.8956	**0.8964**
	Recall	0.8221	0.8358	0.8964	**0.8973**
	Precision	0.8301	0.8429	0.8947	**0.8957**
	F1 score	0.8261	0.8393	0.8956	**0.8965**
>3,000 bp	Accuracy	0.8409	0.8519	**0.8708**	0.8699
	Recall	0.8434	0.8485	**0.8724**	0.8691
	Precision	0.8393	0.8544	0.8696	**0.8706**
	F1 score	0.8413	0.8514	**0.8710**	0.8698
Overall	Accuracy	0.8194	0.8351	0.8652	**0.8673**
	Recall	0.8190	0.8322	0.8672	**0.8685**
	Precision	0.8196	0.8370	0.8637	**0.8664**
	F1 score	0.8193	0.8346	0.8654	**0.8675**

Bold values represent the best performance.

Tabella 3.9

Prestazioni sul metagenoma reale del tratto intestinale umano

Nel metagenoma umano del tratto intestinale, Detire ha nuovamente dimostrato la sua superiorità rispetto a PPR-Meta e DeepVirFinder per l'identificazione di tutte le sequenze in esame, indipendentemente dalla loro lunghezza.

Inoltre, sebbene abbia mostrato una leggera diminuzione di precisione rispetto a CHEER per sequenze con lunghezza compresa tra 500 e 3.000 bp, Detire ha ottenuto risultati superiori in termini di accuratezza, richiamo e punteggio F1.

Nonostante all'interno di questo studio non venga effettuato un confronto con Virtifier, poiché il dataset dell'intestino reale umano utilizzato è lo stesso, possiamo eseguire un confronto approssimato delle prestazioni dei due metodi: confrontando la Tabella 3.6 di Virtifier e la sezione "Overall" della Tabella 3.10, si può notare che in questa specifica situazione Virtifier ottiene punteggio più elevati per quanto riguarda i valori di TPR,

precisione e punteggio F1.

Length	Criteria	Deep-VirFinder	PPR-Meta	CHEER	DETIRE
<500 bp	Accuracy	0.8299	0.8052	0.8831	**0.8905**
	Recall	0.8234	0.8026	0.8857	**0.8962**
	Precision	0.8342	0.8068	0.8811	**0.8828**
	F1 score	0.8288	0.8047	0.8834	**0.8894**
500–1,000 bp	Accuracy	0.8154	0.8154	0.8863	**0.8924**
	Recall	0.8093	0.8044	0.8875	**0.8998**
	Precision	0.8193	0.8225	**0.8954**	0.8867
	F1 score	0.8143	0.8133	0.8864	**0.8932**
1,000–3,000 bp	Accuracy	0.8592	0.8482	0.8860	**0.8874**
	Recall	0.8722	0.8448	0.8777	**0.8832**
	Precision	0.8501	0.8506	**0.8925**	0.8906
	F1 score	0.8610	0.8477	0.8850	**0.8869**
>3,000 bp	Accuracy	0.8317	0.8404	**0.8650**	0.8596
	Recall	0.8393	0.8481	**0.8732**	0.8634
	Precision	0.8267	0.8353	**0.8591**	0.8568
	F1 score	0.8330	0.8416	**0.8661**	0.8601
Overall	Accuracy	0.8369	0.8330	0.8777	**0.8818**
	Recall	0.8416	0.8326	0.8789	**0.8842**
	Precision	0.8337	0.8333	0.8768	**0.8800**
	F1 score	0.8377	0.8329	0.8779	**0.8821**

Bold values represent the best performance.

Tabella 3.10

Tempo di esecuzione dei test

Un altro aspetto preso in considerazione nello studio è il tempo di esecuzione, una caratteristica particolarmente importante quando si lavora con grandi quantità di dati. Nel caso di Detire, questo tempo risulta essere più breve rispetto agli altri metodi su tutti e tre i set di dati esaminati, come si vede nella Tabella 3.11.

L'attrezzatura utilizzata per l'analisi comprende due processori Intel Xeon Gold 6226R con 256 Gb di memoria.

Datasets	Deep-VirFinder	PPR-Meta	CHEER	DETIRE
Testing dataset(s)	762	812	891	**716**
CAMI Marine(s)	362	321	332	**304**
Real human gut(s)	130	114	121	**112**

Bold values represent the best performance.

Tabella 3.11

Capacità di assemblare un genoma virale intero

Infine, Detire è stato sottoposto a un test per verificare se le sequenze virali identificate possono essere assemblate per ottenere il genoma completo del virus. Anche in questo caso,

Detire ha dimostrato una buona capacità di assemblaggio delle sequenze virali, superando gli altri tre metodi nella precisione e nella quantità di sequenze virali ottenute.

Il numero di sequenze virali identificate e l'accuratezza corrispondente sono mostrati nella Tabella 3.12.

Criteria	Deep-VirFinder	PPR-Meta	CHEER	DETIRE
Number of identified reads	1,352	1,402	2,126	**2,203**
Accuracy	0.3909	0.4053	0.6146	**0.6369**
Number of assembled contigs	561	568	873	**890**
Number of hits	212	201	265	**289**

Bold values represent the best performance.

Tabella 3.12

3.4 Future work

Per migliorare l'efficacia dei metodi di identificazione delle sequenze virali, è essenziale esplorare nuove prospettive e direzioni di ricerca. Di seguito vengono presentati alcuni suggerimenti e direzioni future basate sui risultati del confronto tra DeepVirFinder, Virtifier, DETIRE e PPR-Meta.

3.4.1 Architetture neurali per sequenze di diverse lunghezze

Dai confronti effettuati, l'approccio di analizzare separatamente le sequenze virali brevi, ad esempio inferiori a 500 bp, e lunghe potrebbe offrire vantaggi significativi.

Come evidenziato nei metodi analizzati, la lunghezza delle sequenze di input può influenzare notevolmente le prestazioni del modello e per questa ragione, nei prossimi sviluppi, potrebbe essere opportuno implementare strategie di adattamento che consentano ai modelli di gestire in modo ottimale sequenze di varie lunghezze.

Ad esempio, come si è rilevato nell'analisi specifica di Virtifier e PPR-Meta, alcune reti neurali dimostrano maggiore efficacia nell'identificazione di sequenze corte, mentre altre si distinguono per la loro abilità nella gestione di sequenze più estese.

In particolare, le reti neurali ricorrenti (RNN) hanno dimostrato un'eccellente capacità nell'identificazione di sequenze brevi, come è emerso dai risultati di Virtifier, in cui l'impiego del LSTM è risultato particolarmente efficace nel catturare relazioni a lungo termine nelle sequenze virali. Per future implementazioni, potrebbe essere vantaggioso

ottimizzare ulteriormente le RNN al fine di acquisire relazioni a lungo termine ancora più sofisticate nelle sequenze virali più corte. Ciò potrebbe essere conseguito mediante l'utilizzo di meccanismi di attenzione mirati a concentrarsi sulle regioni critiche delle sequenze.

D'altra parte, i risultati indicano che le reti neurali convoluzionali (CNN) si sono dimostrate altamente performanti nell'identificazione di sequenze lunghe. Una prospettiva interessante per il futuro potrebbe essere lo sviluppo di CNN altamente specializzate nell'identificazione di sequenze virali estese. Queste CNN potrebbero incorporare filtri di diverse dimensioni per catturare pattern di dimensione variabile e sfruttare modelli ibridi che combinano più architetture neurali, ciascuna specializzata in un intervallo specifico di lunghezze di sequenza, come accade in parte in PPR-Meta.

3.4.2 Lunghezza dei k-mer e delle sequenze

L'analisi delle prestazioni dei metodi ha inoltre fornito alcune informazioni sulla lunghezza dei k-mer utilizzati nel processo di identificazione. In particolare, è emerso che il punteggio AUROC e le prestazioni generali migliorano all'aumentare della lunghezza dei k-mer. Tuttavia, è stato osservato che una lunghezza di k-mer di 8 ha prodotto buoni risultati per VirFinder, mentre per Detire e Virtifier la prestazioni migliori si sono osservate con valori di k pari a 3. Questo suggerisce che una lunghezza di k-mer relativamente breve potrebbe essere sufficiente per alcune applicazioni di identificazione.

Infatti, oltre a richiede più tempo di calcolo, un aumento eccessivo della lunghezza di k può portare a una complessità del modello eccessiva, con un rischio che il modello possa adattarsi troppo bene ai dati di addestramento avendo di conseguenza prestazioni peggiori su nuovi dati.

È tuttavia fondamentale considerare che la scelta della lunghezza dei k-mer dovrebbe essere adattata al tipo di sequenze virali che si sta cercando di classificare. Pertanto, per i futuri lavori, è consigliabile esplorare diverse lunghezze di k-mer e determinare la lunghezza ottimale in base al contesto specifico dell'analisi, regolando ed esempio la lunghezza dei k-mer in modo dinamico, utilizzando k-mer più lunghi per sequenze più lunghe e k-mer più corti per sequenze più brevi.

3.4.3 Dimensione dell'Embedding

È stato infine osservato che, in generale, un aumento della dimensione della matrice di embedding può portare a una migliore capacità del modello di catturare informazioni complesse e dettagliate nelle sequenze virali. Questo può tradursi in una maggiore precisione nell'identificazione e nella classificazione delle sequenze. Tuttavia, l'effetto esatto dell'aumento delle dimensioni può variare da un metodo all'altro e dipendere dalle caratteristiche specifiche del dataset di addestramento.

Aumentare la dimensione della matrice di embedding può richiedere infatti più risorse computazionali, compreso un maggior quantitativo di memoria e di capacità di calcolo. Inoltre, l'addestramento di modelli con dimensioni di embedding molto grandi può richiedere più tempo, risultando in un fattore limitante per chi lavora con risorse computazionali limitate.

La dimensione della matrice di embedding dovrebbe quindi essere considerata come un compromesso tra prestazioni e risorse computazionali disponibili.

Ne risulta quindi che il futuro lavoro nell'identificazione di sequenze virali dovrebbe concentrarsi sull'ottimizzazione delle architetture neurali in base alle caratteristiche delle sequenze e delle lunghezze coinvolte, con l'obiettivo di sviluppare modelli altamente specializzati che possano riconoscere sequenze virali con alta precisione.

Conclusioni

L'obiettivo di questo studio era quello di esaminare e confrontare le principali strategie di deep learning impiegate per individuare virus all'interno di sequenze genomiche.

Dopo un breve resoconto delle principali tecniche di individuazione adottate nel corso degli anni, sono state introdotte le principali categorie di reti neurali impiegate nei processi di riconoscimento all'interno dei contesti di deep learning. La presentazione ha proseguito con un'analisi dettagliata di ciascuna metodologia, che comprende una valutazione approfondita delle performance dei singoli metodi e una comparazione diretta tra le diverse strategie analizzate, prendendo in considerazione anche la lunghezza delle sequenze prese in esame.

Le analisi condotte sui vari metodi di identificazione virale hanno portato ad alcune considerazioni rilevanti. Per quanto riguarda l'analisi dei risultati ottenuti con DeepVirFinder, sebbene dimostri maggiore efficacia rispetto a VirFinder, specialmente con sequenze più corte, presenta risultati nettamente inferiori rispetto a PPR-META, Virtifier e Detire. Particolarmente degne di nota sono le ottime prestazioni di Virtifier nell'individuazione di sequenze corte, anche se si è osservata una diminuzione delle performance rispetto a PPR-META e Detire quando si è trattato di sequenze oltre i 5000 bp. La comparazione tra questi metodi ha messo in luce l'importanza di considerare la lunghezza delle sequenze in ingresso. Mentre Virtifier si è dimostrato ideale per le sequenze virali brevi, Detire, e in misura maggiore PPR-Meta, hanno mostrato performance superiori con sequenze molto estese. Oltre alle prestazioni, è stato valutato anche il tempo di esecuzione. In questo contesto, Detire si è distinto per un'esecuzione più veloce, rendendolo una scelta interessante per analizzare grandi volumi di dati metagenomici. Le analisi sono state eseguite su diversi tipi di dataset, compresi test, dataset metagenomici marini e un campione reale di metagenoma intestinale umano. In tutti questi scenari, Virtifier e Detire hanno dimostrato prestazioni superiori rispetto agli altri approcci.

Un'identificazione accurata dei virus nei campioni di metagenoma riveste un'importanza cruciale per la comprensione della complessità e della diversità virale che influenzano gli ecosistemi. Il costante perfezionamento delle tecniche di deep learning e la loro adattabilità a nuove situazioni costituiscono aspetti fondamentali per il futuro avanzamento in questo campo di ricerca.

Bibliografia

[1] Zhencheng F. et al. *PPR-Meta: a tool for identifying phages and plasmids from metagenomic fragments using deep learning. GigaScience, 8, 1–14.* 2019. URL: https://doi.org/10.1093/gigascience/giz066 (cit. alle pp. 5, 8, 14).

[2] Miao Y. Liu F. Hou T. e Liu Y. *Virtifier: A deep learning-based identifier for viral sequences from metagenomes. Bioinformatics 38, 1216–1222*. 2022. URL: https://doi.org/10.1093/bioinformatics/btab845 (cit. alle pp. 5, 6, 9, 17, 33).

[3] Buchfink B. et al. *Fast and sensitive protein alignment using DIAMOND. Nat. Methods, 12, 59–60.* 2015. URL: https://doi.org/10.1038/nmeth.3176 (cit. a p. 5).

[4] Tarini S.G. et al. *ProViDE: a software tool for accurate estimation of viral diversity in metagenomic samples. Bioinformation, 6, 91–94.* 2011. URL: https://doi.org/10.6026/97320630006091 (cit. a p. 5).

[5] Almeida J. Zielezinski A. Vinga S. et al. *Alignment-free sequence comparison: benefits, applications, and tools. Genome Biol 18, 186*. 2017. URL: https://doi.org/10.1186/s13059-017-1319-7 (cit. a p. 5).

[6] Eddy S. *What is dynamic programming?. Nat Biotechnol 22, 909–910*. 2004. URL: https://doi.org/10.1038/nbt0704-909 (cit. a p. 6).

[7] Holmes EC. Duffy S Shackelton LA. *Rates of evolutionary change in viruses: patterns and determinants. Nat Rev Genet.* 2008. URL: https://doi.org/10.1038/nrg2323 (cit. a p. 6).

[8] P.F. Hyatt D. Chen GL. LoCascio et al. *Prodigal: prokaryotic gene recognition and translation initiation site identification.* 2009. URL: https://doi.org/10.1186/1471-2105-11-119 (cit. a p. 6).

[9] Salzberg SL. Wood DE. *Kraken: ultrafast metagenomic sequence classification using exact alignments. Genome Biol.* 2014. URL: https://doi.org/10.1186/gb-2014-15-3-r46 (cit. a p. 7).

[10] Wanamaker S. Ounit R Close TJ Lonardi S. *CLARK: fast and accurate classification of metagenomic and genomic sequences using discriminative k-mers.* 2015. URL: https://doi.org/10.1186/s12864-015-1419-2 (cit. a p. 7).

[11] Steven Flygare et al. *Taxonomer: an interactive metagenomics analysis portal for universal pathogen detection and host mRNA expression profiling. Genome Biol.* 2016. URL: https://doi.org/10.1186/s13059-016-0969-1 (cit. a p. 7).

[12] Lu Y.Y. Ren J. Ahlgren N.A. et al. *VirFinder: a novel k-mer based tool for identifying viral sequences from assembled metagenomic data. Microbiome 5, 69.* 2017. URL: https://doi.org/10.1186/s40168-017-0283-5 (cit. alle pp. 7, 10).

[13] Xiaoxi Shen Chang Jiang Yalu Wen et al. *A Brief Review on Deep Learning Applications in Genomic Studies.* 2022. URL: https://doi.org/10.3389/fsysb.2022.877717 (cit. alle pp. 7–9).

[14] Yann Chevaleyre Thanh Hai Nguyen et al. *Deep Learning for Metagenomic Data: using 2D Embeddings and Convolutional Neural Networks*. 2017. URL: https://doi.org/10.48550/arXiv.1712.00244 (cit. a p. 7).

[15] Jie Ren Kai Song Chao Deng et al. *Identifying viruses from metagenomic data using deep learning.* 2020. URL: https://doi.org/10.1007/s40484-019-0187-4 (cit. alle pp. 8, 11).

[16] Jilong Bian Yan Miao et al. *DETIRE: a hybrid deep learning model for identifying viral sequences from metagenomes*. 2023. URL: https://doi.org/10.3389/fmicb.2023.1169791 (cit. alle pp. 9, 21, 37).

[17] Ako-Adjei D Brister JR et al. *NCBI viral genomes resource*. 2014. URL: https://doi.org/10.1093/nar/gku1207 (cit. a p. 10).

[18] Tae Kyun Kim. *T test as a parametric statistic*. 2015. URL: https://doi.org/10.4097/kjae.2015.68.6.540 (cit. a p. 11).

[19] Alex Krizhevsky Nitish Srivastava Geoffrey Hinton et al. *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. 2014. URL: https://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf (cit. a p. 11).

[20] Chengsheng M. Liang Y. Y. L. *Interpreting TF-IDF term weights as making relevance decisions. ACM Trans. Inform. Syst. 26*. 2008. URL: https://doi.org/10.1145/1361684.1361686 (cit. a p. 22).

[21] Hanks P. Church K. W. *Word association norms, mutual information, and lexicography," in Proceedings of the 27th annual meeting on Association for Computational Linguistics (Vancouver: Association for Computational Linguistics;), 76–83.* 1989. URL: https://scholar.google.com/scholar_lookup?title=Proceedings+of+the+27th+annual+meeting+on+Association+for+Computational+Linguistics&author=K.+W.+Church&author=P.+Hanks&publication_year=1989& (cit. a p. 22).