UNIVERSITÁ DEGLI STUDI DI SALERNO

DIPARTIMENTO DI INFORMATICA

Exploration of Gene Fusions through Combinatorial Approach and Machine Learning

Studente

Eduardo Autore
Matr. 052250/1549

Professore

Rocco Zaccagnino

Anno Accademico 2023-2024

Abstract

Nel campo della biologia computazionale, l'associazione accurata delle letture di sequenze RNA (RNA-Seq) con i rispettivi geni di origine rappresenta un obiettivo fondamentale. Affrontare questa sfida richiede l'applicazione di avanzate tecniche di apprendimento automatico (ML). Questo studio si focalizza sull'analisi dell'efficacia delle rappresentazioni basate su **k-fingers**, esplorando un ibrido tra il contesto del **machine learning** e il **calcolo combinatorio**, con un'enfasi particolare sulla rilevazione di fusioni geniche e sul calcolo delle metriche di valutazione. Studiare la rilevazione di fusioni geniche risulta infine essere l'obiettivo finale ed anche quello più interessante poichè la fusione genica è un meccanismo di riarrangiamento cromosomico mediante il quale due (o più) geni si uniscono in un singolo gene (gene di fusione) e ciò è spesso associato al cancro. I riarrangiamenti cromosomici che determinano fusioni geniche sono particolarmente diffusi nei sarcomi e nelle neoplasie ematopoietiche; sono comuni anche nei tumori solidi. Il processo di splicing può anche dare origine a modelli di RNA più complessi nelle cellule. Le fusioni geniche influenzano frequentemente le tirosin chinasi, i regolatori della cromatina o i fattori di trascrizione e possono causare l'attivazione costitutiva, il miglioramento della segnalazione a valle e lo sviluppo del tumore, come principali fattori di oncogenesi. Conoscendo l'importanza delle fusioni geniche in relazione al tumore, vogliamo sogliare i dataset di trscritti così da ottenere un ottimo valore (soglia) tale da farci ottenere il maggior numero di fusioni geniche possibili con un ottimo compromesso in termini di metriche.

Sommario

1 Introduzione

Le fusioni geniche o i riarrangiamenti cromosomici rappresentano un'importante classe di alterazioni somatiche nel cancro e possono avere ruoli importanti nelle fasi iniziali della tumorigenesi.Il primo riarrangiamento cromosomico associato al cancro fu identificato nel 1960 come una traslocazione dei cromosomi 9 e 22.

Gli obiettivi principali di questa analisi sono valutare se le rappresentazioni basate su k-fingers possano:

1. Evidenziare regioni comuni tra le letture RNA-Seq e i trascritti genici.
2. Allenare modelli di ML per comprendere la complessa mappatura tra le letture e i geni.
3. Utilizzare calcoli combinatori al fine di controllare quanto i 2 geni siano fusi, ottenendo delle metriche di valutazione.
4. Valutare le metriche finali

Per affrontare questa sfida, è stato adottato un approccio a tre fasi: inizialmente, addestrando modelli ML per assegnare k-fingers a geni specifici, seguito dallo sviluppo di un classificatore specializzato che utilizza tali k-fingers per associare le letture ai geni corrispondenti ed infine l'utilizzo di varie funzioni di calcolo combinatorio per il calcolo di metriche necessarie per la valutazione del *grado di fusione genica*. Consapevoli dell'importanza e delle implicazioni connesse alla scoperta di una fusione genica rispetto a un singolo gene, andremo ad esplorare in parallelo il campo delle fusioni geniche. Utilizzando le rappresentazioni k-fingers, estendiamo la metodologia per identificare letture *chimeriche*, cioè quelle assegnate a due geni anziché uno solo e *non chimeriche*, cioè quelle assegnate ad un singolo gene (senza fusione). Questo approccio di machine learning senza allineamento è supportato da un classificatore basato su regole, utilizzando le conoscenze acquisite nel contesto della classificazione letture-geni.

2 Fattorizzazione e Kfinger

Nell'analisi delle sequenze, la fattorizzazione svolge un ruolo fondamentale nel comprendere la struttura e le relazioni all'interno delle sequenze di dati. La fattorizzazione di una sequenza consiste nella rappresentazione di essa come una sequenza di sottostringhe (fattori) che, combinate, ricreano la sequenza originale.

2.1 Preliminari: Fingerprint e k-fingers

Una fattorizzazione di una stringa s è una sequenza di fattori $F(s) = <f_1 f_2 ... f_n>$, dove

$$s = f_1 f_2 ... f_n$$

Il fingerprint di s rispetto a F(s) è la sequenza delle lunghezze dei fattori, cioè $L(s) = <|f1|, |f2|, ..., |fn|>$. I k-mers estratti da un fingerprint sono chiamati k-fingers.

2.2 Fattorizzazione di Lyndon

La fattorizzazione di Lyndon è un metodo di fattorizzazione particolarmente significativo in quanto garantisce proprietà uniche e utili per l'analisi di sequenze. Una parola di Lyndon è definita come una sequenza di caratteri che è lessicograficamente strettamente più piccola di ciascuno dei suoi suffissi non vuoti.

2.2.1 Fattorizzazione di Lyndon Diretta (CFL)

La Fattorizzazione di Lyndon diretta di una sequenza genera una sequenza di parole di Lyndon $<f_1, f_2, ..., f_n>$, dove ogni f_i è una parola di Lyndon. Questo processo è essenziale per scomporre una sequenza in elementi fondamentali che conservano le proprietà di Lyndon, fornendo una rappresentazione compatta e unica della sequenza.

Example: CFL(AATTTCGAATTGCTTAACTC) = ⟨AATTTCG, AATTGCTT, AACTC⟩.

2.2.2 Fattorizzazione Inversa di Lyndon (ICFL)

La fattorizzazione di Lyndon Inversa svolge un ruolo cruciale quando è necessario considerare l'ordine inverso delle sequenze. Producente la stessa sequenza di parole di Lyndon, ma in ordine inverso, questa fattorizzazione è di particolare importanza nell'analisi di sequenze biologiche a doppio filamento.

Example: ICFL(TACTATAGTTCTG) = ⟨TAC,TATAG,TTCTG⟩.

2.2.3 Importanza della Fattorizzazione di Lyndon

La fattorizzazione di Lyndon riveste un ruolo cruciale in vari contesti, grazie alle sue proprietà distintive.

1. **Unicità e Linearità**

   Ogni parola $w \in \Sigma^*$ ha una fattorizzazione unica in termini di parola di Lyndon $\langle <f_1, f_2, ..., f_n> \rangle$ e questo processo può essere eseguito in tempo lineare rispetto alla lunghezza della parola.Unicità e linearità forniscono una rappresentazione efficiente e senza ambiguità delle sequenze.

2. **Conservazione delle Relazioni di Similarità**

   La fattorizzazione di Lyndon conserva le relazioni di similarità tra le sequenze. Questo significa che sequenze simili mantengono una struttura simile anche nella loro rappresentazione di Lyndon. Tale conservazione delle relazioni è cruciale per l'analisi dettagliata delle sequenze, specialmente in contesti biologici.

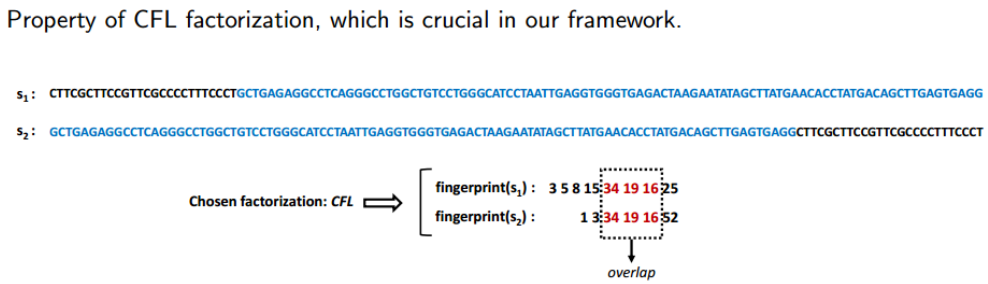

Figura 1: Proprietà di conservazione.

2.3 Fattorizzazioni a Due Livelli (CFL_ICFL)

Le fattorizzazioni a due livelli, ottenute attraverso l'unione di CFL (Fattorizzazione di Lyndon Diretta) e ICFL (Fattorizzazione Inversa di Lyndon), rappresentano un approccio avanzato nell'analisi delle sequenze. Questo paragrafo esplorerà il processo di ottenimento della fattorizzazione CFL_ICFL(w) da una parola w e spiegherà le ragioni per cui questa unione produce risultati superiori rispetto alle singole fattorizzazioni di CFL e ICFL.

- **Procedura di Fattorizzazione CFL ICFL(w)**

1. **CFL(w)** $= \langle f_1, ..., f_t \rangle$: La sequenza w viene scomposta in fattori mediante la Fattorizzazione di Lyndon Diretta.
2. **ICFL(f_i) Per ogni** f_i: Si applica la Fattorizzazione inversa di Lyndon a ciascun fattore f_i ottenuto nella prima fase, considerando solo quelli con lunghezza $|f_i|$ superiore a una soglia T.

- **Vantaggi Rispetto a CFL o ICFL Singolarmente**

1. **Arricchimento della Fattorizzazione**: L'unione di CFL e ICFL arricchisce la fattorizzazione in termini di numero di fattori, fornendo una visione più dettagliata della struttura della sequenza.
2. **Unicità e Computazione Lineare**: La combinazione di CFL e ICFL garantisce l'unicità della facttorizzazione e la capacità di eseguire il calcolo in tempo lineare rispetto alla lunghezza della parola, fornendo una rappresentazione efficiente e accurata delle sequenze.
3. **Conservazione delle Relazioni**: La proprietà di conservazione, cruciale nell'analisi dettagliata delle sequenze, persiste anche nell'unione di CFL e ICFL.

2.4 Lyn2vec tool

Lyn2vec è una metodologia innovativa presentata in conferenze di rilievo come AlCoB 2020 e 2021 (Algorithms for Computational Biology) che si pone l'obiettivo di risolvere la sfida della rappresentazione numerica delle sequenze biologiche. La sua metodologia, si basa sull'algoritmo di fattorizzazione noto come CFL, spiegato nel precedente capitolo e in questo contesto, esploreremo l'applicazione di lyn2vec su una sequenza specifica s1.

Consideriamo CFL come algoritmo di fattorizzazione e impostiamo k, la lunghezza dei k-finger, a 3. Applichiamo l'algoritmo CFL alla sequenza s1:

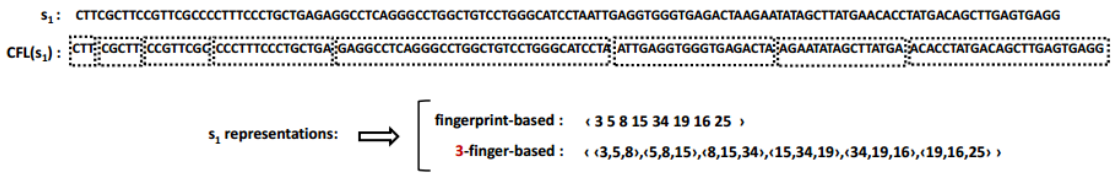

Figura 1: Lyn2Vec con K = 3.

Per comprendere il funzionamento, introduciamo il concetto di **superfingerprint** in modo semplice, quindi immaginiamo di avere due alfabeti ordinati, come A, C, G, T e T, G, C, A, una stringa s e un algoritmo di suddivisione chiamato F. Il superfingerprint di s è ottenuto **concatenando due impronte digitali**: Lp(s), $, Lp'(s). Qui, Lp(s) rappresenta l'impronta digitale ottenuta applicando l'algoritmo F all'alfabeto A, C, G, T sulla stringa s, mentre Lp'(s) è l'impronta digitale ottenuta applicando l'algoritmo F all'alfabeto T, G, C, A sulla stessa stringa s. Il simbolo $ è utilizzato per segnare la separazione tra le due impronte digitali.

Lyn2vec riceve in input un file di letture di sequenza in formato FASTA o FASTQ ed esegue un algoritmo di fattorizzazione (specificato in input) su ogni lettura in modo da calcolarne la rappresentazione. In particolare, lyn2vec produce i seguenti tipi di rappresentazioni:

1. L'impronta digitale (fingerprint).
2. La sequenza dei k-fingers estratti dall'impronta digitale della lettura (dato un valore di k specificato come parametro di input).
3. La sequenza dei k-fingers estratti dal superfingerprint, dopo aver scartato i k-fingers contenenti il separatore $.

Considerando la fattorizzazione a 2 livelli (CFL_ICFL), spiegata nel capitolo precedente, facendo un esempio considerando:

$$s = GCATCACCGCTCTACAG.$$

Utilizzando l'algoritmo CFL ICFL con una soglia $\mathbf{T = 30}$, la fattorizzazione e l'impronta digitale di s (rispetto all'**ordinamento "regolare"** A; C; G; T) saranno rispettivamente:

$$\text{CFL\_ICFL s} = \text{G, C, ATC, ACCGCTCT, ACAG e L(s)} = 1, 1, 3, 8, 4\ .$$

Si osserva che la fattorizzazione rispetto all'**ordinamento inverso** T; G; C; A, produce l'impronta digitale 1, 2, 7, 7. Quindi, il superfingerprint sarà dato da:

$$S = 1, 1, 3, 8, 4, \$, 1, 2, 7, 7$$

ovvero dalla concatenazione delle due impronte digitali intervallate dal separatore $. Supponendo k = 3, lyn2vec produrrà una delle seguenti tre rappresentazioni:

1. L'impronta digitale L(s) = 1, 1, 3, 8, 4 ottenuta da CFL ICFL s.
2. La sequenza 1, 1, 3, 3, 8, 8, 4 dei k-fingers estratti da L(s).
3. La sequenza 1, 1, 3, 3, 8, 8, 4, 1, 2, 7, 7 dei k-fingers estratti da S, dopo aver scartato quelli contenenti il separatore $.

Considerazioni finali

Le rappresentazioni risultanti per la sequenza s1 dimostrano l'efficacia di lyn2vec nella creazione di rappresentazioni numeriche significative per sequenze biologiche. La sequenza è trasformata in impronte digitali e 3-finger, evidenziando la capacità di lyn2vec di catturare informazioni rilevanti dalla sequenza originale. Inoltre, l'applicazione specifica di CFL e k = 3 fornisce una configurazione ottimale per catturare le caratteristiche distintive delle sequenze biologiche, dimostrando la flessibilità di lyn2vec nell'adattarsi a diversi contesti.

Lyn2vec ha dimostrato che la sequenza di k-fingers è una rappresentazione numerica adatta per sequenze biologiche che possono essere apprese in modo efficace da modelli di apprendimento automatico (ML). L'utilizzo di questa rappresentazione numerica apre nuove prospettive nella comprensione e nell'analisi delle sequenze biologiche, fornendo un ponte tra le informazioni biologiche e le capacità di apprendimento automatico.

Nel prossimo capitolo discuteremo dell'utilizzo di questo tool per l'estrazione di k-finger da sequenze relative a dei pannelli genici al fine di costruire un modello in grado di predire e riconoscere pattern ricorrenti di k-finger nelle sequenze e quindi in grado di capire a quale gene appartengano.

3 Scelta e costruzione del modello

In questo capitolo, esploreremo il problema della classificazione Read-Gene utilizzando lyn2vec, che, come descritto nella sezione 2.4, fornisce **tre tipi di rappresentazione delle read**. Di conseguenza, abbiamo affrontato il problema della classificazione Read-Gene in tre differenti contesti, ognuno considerando una rappresentazione fornita da lyn2vec. Abbiamo organizzato i nostri esperimenti in tre gruppi, chiamati task, rispondendo alle seguenti tre domande:

- **T1**. "Quanto è efficace l'impronta digitale prodotta da lyn2vec come rappresentazione di una read nel problema della classificazione Read-Gene?"

- **T2**. "Quanto è efficace la sequenza dei k-finger estratti dalle impronte digitali prodotte da lyn2vec come rappresentazione di una read nel problema della classificazione Read-Gene?"

- **T3**. "Quanto è efficace la sequenza dei k-finger estratti dalla superfingerprint prodotta da lyn2vec come rappresentazione di una read nel problema della classificazione Read-Gene?"

Ogni task è composto da sotto-task (o esperimenti), ciascuno dedicato a individuare e valutare un classificatore utilizzando una specifica rappresentazione.

- **Task 1: Classificazione basata su Impronte Digitali**
  Nel primo task (T1), esploriamo l'uso diretto delle impronte digitali lyn2vec come vettori di caratteristiche per la classificazione Read-Gene. Ogni impronta digitale corrisponde a un vettore di caratteristiche, e modelli di apprendimento automatico vengono addestrati su questi vettori per classificare le read nei rispettivi geni. Questo task fornisce approfondimenti sull'efficacia delle impronte digitali lyn2vec come rappresentazioni autonome per dati

privi di errori.

- **Task 2: Classificazione basata su Sequenza di k-finger**
  Passando al secondo task (T2), indaghiamo sull'uso delle sequenze di k-finger, estratte dalle impronte digitali lyn2vec, come vettori di caratteristiche per la classificazione Read-Gene. Qui, seguiamo un processo a due fasi: (i) addestramento di modelli per assegnare ogni k-finger a un gene e (ii) utilizzo di un classificatore speciale (Sezione 4.1.2) che sfrutta il risultato ottenuto per assegnare ogni read a una classe (o gene).

- **Task 3: Classificazione basata su Sequenza di k-finger dalla superfingerprint**
  Nel terzo task (T3), estendiamo l'esplorazione alle sequenze di k-finger estratte dalle superfingerprint lyn2vec. Similmente al Task 2, ciò comporta un processo di classificazione a due fasi. Inizialmente, modelli vengono addestrati per assegnare geni ai k-finger estratti dalle superfingerprint, e successivamente un classificatore specializzato viene impiegato per assegnare read a geni basandosi sugli assegnamenti dei k-finger. Questo task mira a valutare l'efficacia delle superfingerprint nella classificazione Read-Gene.

3.1 Dati presi in analisi

Per i nostri esperimenti, è stata utilizzata l'annotazione (havana ed ensembl_havana) dei 6040 geni umani dei cromosomi 1, 17 e 21, contenente un totale di 17,314 trascritti. Sono stati selezionati casualmente 100 geni tra di essi per ottenere un piccolo set di geni per valutare l'efficacia della nostra rappresentazione di incorporamento. Per ciascuno dei 100 geni, sono stati selezionati casualmente 4 trascritti (per un totale di **400 trascritti**). Successivamente, da ciascuno di questi 400 trascritti, sono stati estratti tutti i 100-mers, ottenendo così un totale di **797,407 sottostringhe lunghe 100**.

3.2 Estrazione delle Caratteristiche

I dataset di vettori di caratteristiche sono stati estratti dalla collezione di 797,407 read di input lunghe 100 del nostro dataset. Ogni impronta digitale corrisponde a un vettore di caratteristiche nel task T1, e ogni k-finger estratto da un'impronta digitale (risp. superfingerprint) corrisponde a un vettore di caratteristiche nei task T2 (risp. T3). In dettaglio, sono stati considerati un totale di 10 algoritmi di fattorizzazione: i quattro algoritmi CFL, ICFL, CFL^d e $ICFL^d$ più i due algoritmi CFL ICFL e CFL $ICFL^d$ applicati per i tre valori 10, 20, 30 del parametro T. Per ciascun algoritmo, è stato calcolato (utilizzando lyn2vec) l'impronta digitale e la superfingerprint di ciascuna read di input, ottenenendo un certo numero di dataset per ogni esperimento. Successivamente, sono stati estratti i k-finger per k da 3 a 8 (un totale di sei valori) dai dataset così da ottenere dei dataset di k-finger. Poiché la lunghezza di un vettore di caratteristiche deve essere costante e il numero di elementi (valori interi) che compongono i vettori di caratteristiche (impronte digitali o k-finger) è chiaramente variabile, è necessario eseguire un riempimento con 1 finali per ottenere una dimensione costante su tutti i vettori di caratteristiche. Osserviamo che un k-finger può anche essere riempito quando l'impronta digitale è più breve di k.

3.3 Valutazione classificatore migliore

Metodologia di Base : Per ciascun esperimento, sono stati eseguiti i seguenti passaggi per produrre un classificatore:

Etichettatura: Ogni vettore di caratteristiche (impronta digitale o k-finger) nel dataset è etichettato per creare un collegamento con la sua classe (cioè, con il gene di origine).

Apprendimento: Sono stati considerati i seguenti modelli di apprendimento automatico: **Random Forest (RF)**, **Logistic Regression (LR)** e **Multinomial Naive Bayes (MNB)**. Ciascun modello è stato addestrato utilizzando i campioni etichettati. Successivamente, la tecnica di k-fold cross-validation (ovvero, trovare i valori iperparametrici ottimali che producono una performance di generalizzazione soddisfacente) è stata utilizzata in combinazione con il metodo GridSearchCV al fine di selezionare il miglior modello e ottenere un classificatore. Il dataset è stato prima diviso in due sottoinsiemi utilizzando la stratificazione: un set di addestramento (80% dei campioni) e un set di test (20% dei campioni). La stratificazione garantisce che tutte le 100 classi (geni) considerate, siano rappresentate in entrambi i set, mantenendo le stesse proporzioni del dataset originale. Inoltre, per minimizzare la perdita di dati, è stata utilizzata una pratica comune di apprendimento automatico che consiste nell'applicare la normalizzazione all'interno dei fold della cross-validation, separatamente. In particolare, abbiamo normalizzato i dati utilizzando la tecnica MinMaxScaler. La normalizzazione è utilizzata per mappare i valori delle caratteristiche (valori interi nel nostro caso) su un intervallo fisso (di solito [0, 1]), fornendo risultati migliori rispetto al caso in cui le caratteristiche abbiano valori in range differenti. Successivamente, la k-fold cross-validation è stata eseguita sul set di addestramento (sono stati provati diversi valori di k e 5 dà buoni risultati), e, ad ogni passo k (ad ogni passo, k - 1 fold vengono utilizzati per addestrare i modelli, e un fold, chiamato test fold, viene utilizzato per la valutazione delle prestazioni), è stata applicata una ricerca esaustiva con parametri specificati (metodo GridSeachCV), e per ciascuna combinazione di tali valori, è stata calcolata la prestazione media del classificatore considerato raggiunta sul corrente test fold indipendente. Infine, dopo aver scoperto valori iperparametrici

soddisfacenti, è stata eseguita una ritraining del miglior modello (il modello che mostra la migliore performance) sul set di addestramento, producendo così un classificatore.

Valutazione: La capacità di generalizzazione del classificatore ottenuto, cioè la capacità di raggiungere alte prestazioni su campioni non visti, è stata valutata applicandolo per classificare gli elementi del set di test utilizzando i migliori parametri trovati nel passo precedente.

Miglior modello
Dai vari test effettuati è emerso in base alle metriche che il miglior modello è risultato essere quello basato su **Random Forest(RF)**, in particolare il modello RF addestrato sui k-fingers estratti dalle superfingerprints, calcolate con l'algoritmo CFL_ICFL^d e con T = 30, considerando k = 8, ha ottenuto una precisione media ponderata di 0,91, recall 0,77 e F1 score 0,82.(fase di valutazione).

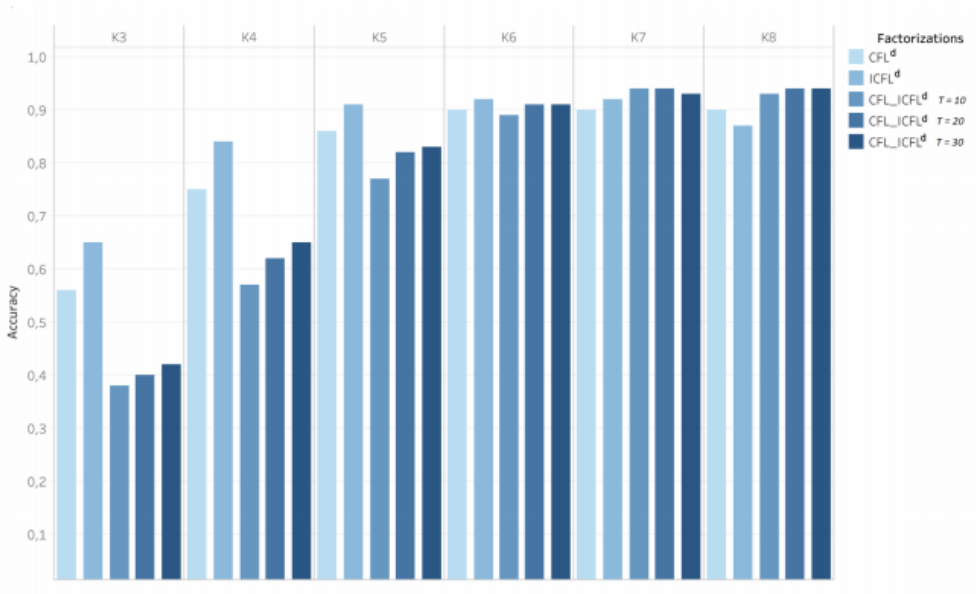

Figura 1: Accuratezza raggiunta con RF per ogni tipo di fattorizzazione a doppio filamento e k da 3 a 8.

Dopo aver costruito il modello, nel prossimo capitolo esploreremo la costruzione di un sistema di classificazione read-gene che prende i k-

finger predetti per ogni read ed effettua dei calcoli combinatori al fine di calcolare una percentuale di coverage per ciascun gene e determinare se la read è chimerica o non chimerica

4 Chimeric read-gene classification

Problema: Il problema di classificazione delle letture chimeriche consiste nell'individuare letture genetiche(reads e nello specifico sequenze nucleotidiche) che si appaiano a due geni anziché uno solo.

Metodologia Generale: La metodologia sviluppata per la classificazione delle reads è generalizzata per implementare un approccio di apprendimento automatico (ML) in modo ibrido con un approccio combinatorio. Viene utilizzato il classificatore di k-fingers, di cui abbiamo discusso nel capitolo 3, precedentemente addestrato per la classificazione delle reads geniche e successivamente viene definito un classificatore basato su regole specifiche per risolvere il problema della classificazione delle reads geniche chimeriche.

1. **Addestramento del Classificatore a K-fingers**:
   Per addestrare il classificatore a k-fingers, si considera un insieme di geni(pannello di geni) e i loro trascritti annotati. Per ciascun trascritto, si estraggono tutte le sottostringhe di dimensione pari alla finestra specificata. Successivamente, vengono estratti i k-fingers utilizzando l'algoritmo CFL_ICFL^d con parametri specifici. Il dataset di addestramento è costituito da tutti i k-fingers etichettati con il gene di origine.

2. **Adattamento della Fattorizzazione per Reads Lunghe**:
   Per calcolare i k-fingers per reads lunghe, viene sviluppata una **fattorizzazione adattativa** guidata dal set di fattori calcolato nella fase di addestramento. Ogni parola è divisa in sottostringhe in modo che esista un L-target-factor nel set di fattori che è prefisso di ciascuna sottostringa.

3. **Classificazione delle Letture Lunghe in Chimeriche/Non-chimeriche**:

Date le sottostringhe ottenute dalla fattorizzazione adattativa $w = w_1...w_n$, si applica CFL_$ICFL^d$ per ogni w_i (T = 30, k = 8).
Per ogni sottostringa w_i, vengono **estratti tutti i k-fingers** e ne viene ottenuta una lista ordinata $<K_1, K_2, ..., K_q>$.
Il classificatore a k-fingers viene applicato a questa lista, generando una **lista ordinata di geni assegnati ai k-fingers**:

$$ListG(w) = <g_1, g_2, ..., g_q>$$

Sulla base di questa lista, viene valutata la ripetitività di ciascun gene.

4. **Calcolo della misura di copertura dei geni e fusion score**:

   La ripetitività massima di un gene è utilizzata per calcolare uno score di fusione tra due geni, indicando quanto sia probabile che i due geni siano fusi in una lettura lunga.

   La ripetitività di un gene g in ListG(w) è definita come il numero di coppie di elementi consecutivi in ListG(w) che sono uguali a g.

4.1 Calcolo del coverage e fusion score

La copertura genica per il gene g, indicata come **g-coverage**, è rappresentata dalla sottolista massima $<g_i, g_{i+1}, ..., g_j>$ in ListG(w), soddisfacendo le seguenti condizioni:

- $g_i = g_j = g$,
- g è il gene con la massima ripetitività in ListG(w).

Nei vari algoritmi combinatori utilizzati per avere una valutazione del **grado di fusione** delle varie coppie di geni, abbiamo una porzione di codice in comune relativa al calcolo del punteggio di fusione, che si prefissa come obiettivo, quello di rispondere alla domanda: È possibile misurare quanto i geni g1 e g2 siano propensi a fondersi in w?

E per rispondere a questa domanda, il punteggio di fusione per due geni g1 e g2 in una lettura lunga w è stato definito nel seguente modo:

$$score_w(g1, g2) = \frac{(P1 + P2 - P\cap)}{|ListG(w)|}$$

Dove:

- **P1 (P2)**: rappresenta la lunghezza della lista g1-coverage (g2-coverage).
- $\boldsymbol{P\cap}$: è il numero di elementi comuni tra le liste g1-coverage e g2-coverage,
- $\boldsymbol{|ListG(w)|}$: è la lunghezza totale della lista ListG(w).

Procedura per ogni Pannello di geni

1. Generiamo dal pannello di geni un dataset chimerico e uno non chimerico.

2. Calcoliamo una **soglia adattiva** sui due dataset (chimerico e non chimerico) in base alle metriche ottenute e quindi in base al numero di *true positive*, *true negative*, *false positive* e *false negative*, riassunte con l'**F1-score**. Alla fine otteremo una soglia che ci porterà ad un buon compromesso tra le varie metriche, ottimizzando quindi il numero di letture chimeriche restituite.
3. Per ogni lettura lunga in input w (in entrambi i dataset) e per ogni coppia di geni g1 e g2, calcoliamo la copertura g1 e la copertura g2 rispetto a ListG(w).

4. Se la copertura g1 non è racchiusa nella copertura g2 (o viceversa), il punteggio di fusione (g1, g2) viene quindi calcolato.

5. Tra tutte le coppie di geni in input, selezioniamo la coppia che dà il punteggio più alto ed etichettiamo come chimerico se quel punteggio è maggiore o uguale della soglia adattiva precedentemente calcolata.

È importante notare che lo $score_w(g1, g2)$ è normalizzato e varia nell'intervallo [0, 1]:

- **valore 1**: indica una copertura perfetta della lista ListG(w) da parte dei due geni.

- **valore 0**: indica l'assenza di copertura da parte dei due geni.

Figura 1: Esempio di come potrebbe apparire la copertura g di due geni fusi (geni rappresentati da lettere).

4.2 Metodi combinatori presi in esame

Per calcolare la misura di copertura, abbiamo preso in esame tre algoritmi combinatori differenti che calcolano il fusion score per ogni dataset, in base ai range delle varie coppie di geni prese in esame.

4.2.1 statistical analysis with known genes check-range-majority

1. Viene caricato il file delle predizioni (fingerprint predette per ogni read).

2. viene effettuato un primo controllo per verificare che i due geni siano realmente presenti, altrimenti passa alla coppia di geni successiva.

3. Vengono analizzati **due casi** di sovrapposizione dei range dei due geni

   (a) Controllo che assicura che gli intervalli dei due geni siano ben definiti, non abbiano lunghezza negativa e che il **primo gene**

termini dopo la fine del secondo, evitando sovrapposizioni indesiderate o situazioni inconsistenti negli intervalli dei due geni.

(b) Controllo che assicura che gli intervalli dei due geni siano ben definiti, non abbiano lunghezza negativa e che il **secondo gene termini prima del primo**, evitando sovrapposizioni indesiderate o situazioni inconsistenti negli intervalli dei due geni. La differenza principale rispetto al controllo precedente è l'ordine delle condizioni, che cambia l'orientamento della relazione tra i due geni.

4. Qualsiasi caso si verifichi dei due precedenti, viene effettuato un controllo per verificare che il fusion score sia maggiore o uguale alla soglia adattiva.

5. Se tutte queste condizioni vengono rispettate, vengono calcolate le metriche in base al dataset di input (chimerico o non chimerico), generando true positive, true negative, false positive e false negative, riassunte dall'F1-score.

4.2.2 statistical analysis with known genes no-check-range-majority

1. Viene caricato il file delle predizioni (fingerprint predette per ogni read).

2. viene effettuato un primo controllo per verificare che i due geni siano realmente presenti, altrimenti passa alla coppia di geni successiva.

3. Va a cercare il **gene più presente (o più comune)**, ovvero quello che si manifesta il maggior numero di volte sia nel range del

gene 1, sia nel range del gene 2.

4. Viene effettuato un controllo per verificare che il fusion score sia maggiore o uguale alla soglia adattiva.

5. Se tutte queste condizioni vengono rispettate, vengono calcolate le metriche in base al dataset di input (chimerico o non chimerico), generando true positive, true negative, false positive e false negative, riassunte dall'F1-score.

4.2.3 statistical analysis with known-genes-consecutive-frequency

1. Viene caricato il file delle predizioni (fingerprint predette per ogni read).

2. viene effettuato un primo controllo per verificare che i due geni siano realmente presenti, altrimenti passa alla coppia di geni successiva.

3. Va a cercare il **gene che si ripresenta più volte consecutivamente**, ovvero quello che si manifesta il maggior numero di volte ma in modo consecutivo sia nel range del gene 1, sia nel range del gene 2.

4. Viene effettuato un controllo per verificare che il fusion score sia maggiore o uguale alla soglia adattiva.

5. Se tutte queste condizioni vengono rispettate, vengono calcolate le metriche in base al dataset di input (chimerico o non chimerico), generando true positive, true negative, false positive e false negative, riassunte dall'F1-score.

Questi metodi combinatori ci permettono di capire, per ogni pannello di geni, quale sia il migliore in base alle metriche restituite e quindi implicitamente anche in base al numero di letture chimeriche e non chimeriche restituite.

5 Test effettuati

Sono stati effettuati test su 2 pannelli genici, uno contenente tutti e 100 i geni, l'altro su un pannello di 10 geni presi a caso e sono state eseguite le seguenti procedure:
Prima sono stati generati i fingerprint dai trascritti relativi ad un dataset genico, poi con questi ultimi è stato addestrato il modello con Random Forest.

5.1 Generazione dataset fusioni con Fusim

Per prima cosa è stato eseguito il tool Fusim, che dato il **nome di due o più geni ed i loro trascritti collegati**, il genoma di riferimento (nel nostro caso **hg19.fa**) ed il modello genico **refFlat**, permette di generare un numero arbitrario di trascritti fusi o **chimerici** (ovvero con pattern di nucleotidi del gene 1 e pattern di nucleotidi del gene 2) oppure **non chimerici** se i due geni di partenza su cui viene applicato l'algoritmo sono uguali. Successivamente è stato utilizzato uno script per portare tutti i trascritti generati in un formato compatibile, generando un file **.FASTQ**.

Alla fine viene quindi dato in input al modello a Random forest il dataset di fusioni .FASTQ simulate con Fusim sul pannello genico di dimensione arbitraria, generando un file di predizioni denominato come **test_fusion_result_CFL_ICFL_COMB-30_K8_nome_dataset.txt** , dove le fattorizzazioni su cui abbiamo addestrato il modello e su cui abbiamo predetto i Fingerprint, sono basate su $CFL-ICFL^d$**, con T = 20 o T = 30, ovvero i valori che hanno fornito le prestazioni migliori**.

5.2 Esecuzione di metodi combinatori

Scegliamo se eseguire uno, due o tutti e tre i metodi combinatori in base alle esigenze di test. Dopo aver eseguito quindi il metodo opportuno, esso genererà due file di conteggio delle fusioni, ovvero con il numero di occorrenze per ogni gene nel dataset:

- **gene_fusion_count** : con il formato numero_gene-numero di occorrenze di quel gene

  - 98:35
  - 98:28
  - 98:12
  - 92:82

- **parsed_gene_fusion_count** : con il formato nome_gene-numero di occorrenze di quel gene

  - ENSG00000205129:35
  - ENSG00000205129:28
  - ENSG00000205129:12
  - ENSG00000197580:82

Successivamente il metodo caricherà le predizioni dal file **test_fusion_result_CFL_ICFL_COMB-30_K8_nome_dataset.txt** ed effettuerà dei calcoli sui range genici, calcolando infine un fusione score basato su soglia adattiva, come spiegato nel capitolo 4 e nello specifico il 4.2, generando un file chiamato **statistics_Method_dataset_name (chimerico o non chimerico)** con le informazioni sulle fusioni per ogni lettura.

5.2.1 Risultati

Dataset di 10 geni

I risultati di test per il dataset da 10 geni ha dato per i 3 metodi le seguenti metriche:

Metric	Value
Accuracy	0.94
Recall	0.94
Precision	1.00
F1-score	0.97
Specificity	0.00

(a) Check Range Majority.

Metric	Value
Accuracy	0.72
Recall	0.95
Precision	0.74
F1-score	0.83
Specificity	0.05

(b) Most Consecutive Frequency.

Metric	Value
Accuracy	0.71
Recall	0.95
Precision	0.73
F1-score	0.83
Specificity	0.03

(c) No Check Range Majority.

Figura 1: I tre metodi combinatori applicati al dataset da 10 geni.

E nello specifico andando ad approfondire il metodo Check range Majority, notiamo che esso ha dato una **soglia ottimale di 0.1** e come metriche: 165 True positive, 0 False positive, 0 True negative e 10 False negative, trovando **165 trascritti chimerici** sui **218** trascritti che hanno rispettato i controlli dei metodi combinatori e di soglia (partendo da 450 trascritti con label chimerica e 100 con label non chimerica), riducendo al minimo False positive e False Negative.

Questa soglia è ottimale in quanto andando ad aumentarla, il compromesso delle metriche diventa sempre meno vantaggioso, riducendo eccessivamente il numero di letture chimeriche. Gli altri metodi combinatori (Most consecutive frequency e No check range Majority) in questo caso danno risultati nettamente inferiori a Check Range Majority, sottolineando come la scelta del metodo per ogni pannello genico possa fare la differenza in termini di risultati e che eseguendo tutte e tre le metriche sul pannello genico, si possa avere un quadro generale del numero di letture chimeriche e non chimeriche.

Dataset di 100 geni

I risultati di test per il dataset da 100 geni ha dato per i 3 metodi le seguenti metriche:

Metric	Value
Accuracy	0.96
Recall	0.96
Precision	1.00
F1-score	0.98
Specificity	0.00

(a) Check Range Majority.

Metric	Value
Accuracy	0.94
Recall	0.97
Precision	0.98
F1-score	0.97
Specificity	0.05

(b) Most Consecutive Frequency.

Metric	Value
Accuracy	0.94
Recall	0.97
Precision	0.98
F1-score	0.97
Specificity	0.05

(c) No Check Range Majority.

Figura 2: I tre metodi combinatori applicati al dataset da 100 geni.

E nello specifico andando ad approfondire il metodo Check range Majority, notiamo che esso ha dato una **soglia ottimale di 0.1** e come metriche: 28424 True positive, 0 False positive, 0 True negative e 1097 False negative, trovando **28424 trascritti chimerici** sui **29521** trascritti che hanno rispettato i controlli dei metodi combinatori e di soglia (partendo da 44652 trascritti con label chimerica e 960 con label non chimerica), riducendo al minimo False positive e False Negative.

Questa soglia è ottimale in quanto, allo stesso modo dell'esperimento con 10 geni, andando ad aumentarla, il compromesso delle metriche diventa sempre meno vantaggioso, riducendo eccessivamente il numero di letture chimeriche. Rispetto all'esperimento con 10 geni, il metodo Check range Majority risulta migliore solo di poco rispetto agli altri metodi combinatori (Most consecutive frequency e No check range Majority), precisamente di un solo punto di F1-score. Quindi in questo caso la scelta del metodo check range Majority non risulta così incidente rispetto all'esperimento precedente, quindi utilizzare gli altri due metodi non peggiorerebbe eccessivamente i risultati.

6 Conclusione e sviluppi futuri

In conclusione possiamo affermare di essere riusciti a ottenere lo scopo desiderato, ovvero riuscire a sogliare dei dataset di trascritti (letture lunghe), così da trovare quel punto di essi (soglia), dal quale riusciamo a trovare il maggior numero di letture chimeriche con il un compromesso ottimale. Nonostante la scelta della soglia adattiva sia efficiente e permetta di trovare ottimi compromessi, essa può ancora essere migliorata poichè potremmo sceglierla non solo in base al solo valore di F1-score, ma anche in base ad un certo numero di letture chimeriche e non chimeriche oppure privilegiare maggiormente la metrica false positive rispetto alla false negative e viceversa in casi in cui le due singole metriche risultino poco bilanciate tanto da peggiorare i risultati.