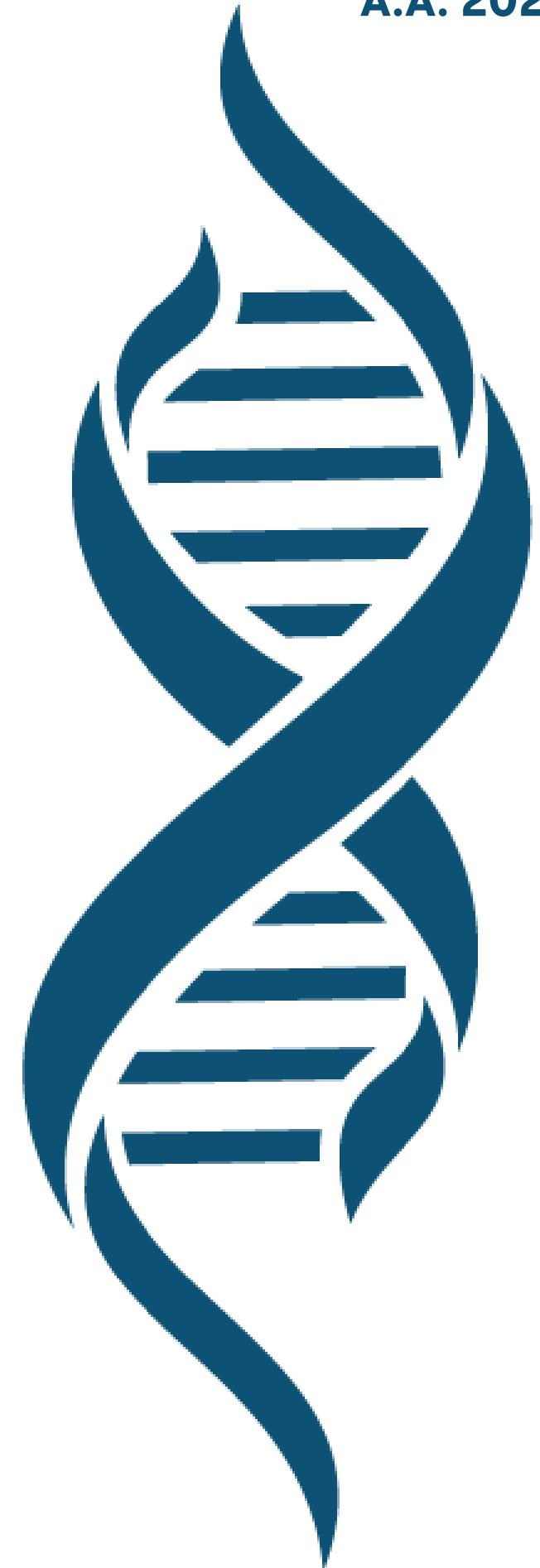




UNIVERSITÀ  
DEGLI STUDI  
DI SALERNO

A.A. 2023/2024

# LEVERAGING GENE EXPRESSION AND GENOMIC VARIATION FOR CANCER PREDICTION USING ONE-SHOT LEARNING



Strumenti formali per la Bioinformatica

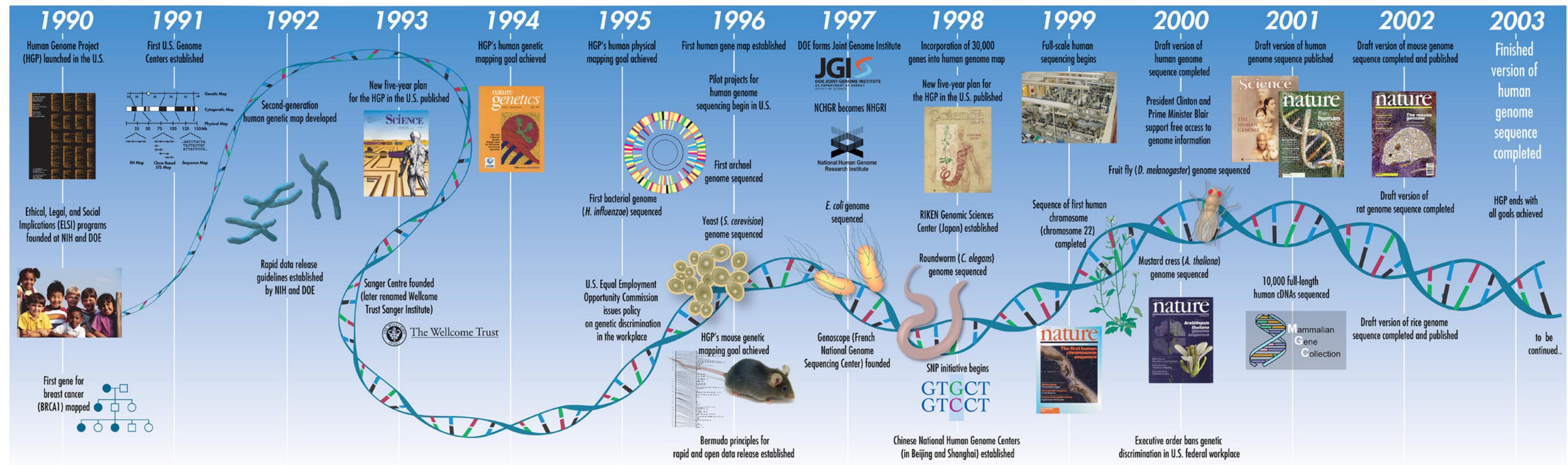
Alessandro Macaro 05225 01459  
Alberto Montefusco 05225 01498

# Introduzione

Sono trascorsi più di 20 anni dal completamento della bozza della sequenza del genoma umano. Questo traguardo ha portato alla generazione di una straordinaria quantità di dati genomici.

## Conseguenze:

- Il sequenziamento del DNA e altre tecniche biologiche continuano a aumentare la quantità e la complessità dei dati
- La crescente complessità richiede strumenti computazionali basati su AI/ML, essenziali per gestire, estrarre e interpretare informazioni dai dati genomici.

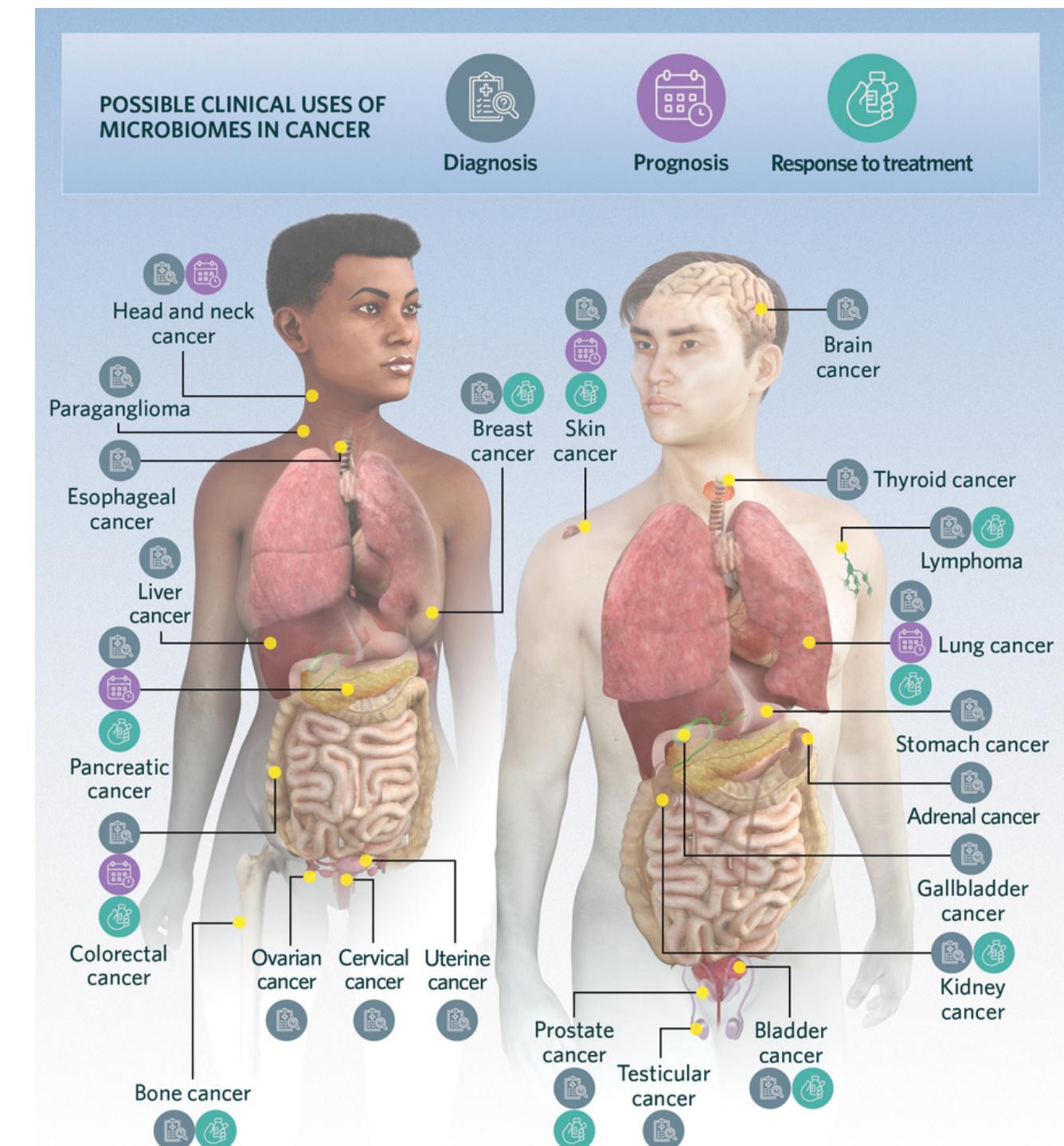


# Stato dell'arte

1

## A deep learning system accurately classifies primary and metastatic cancers using passenger mutation patterns

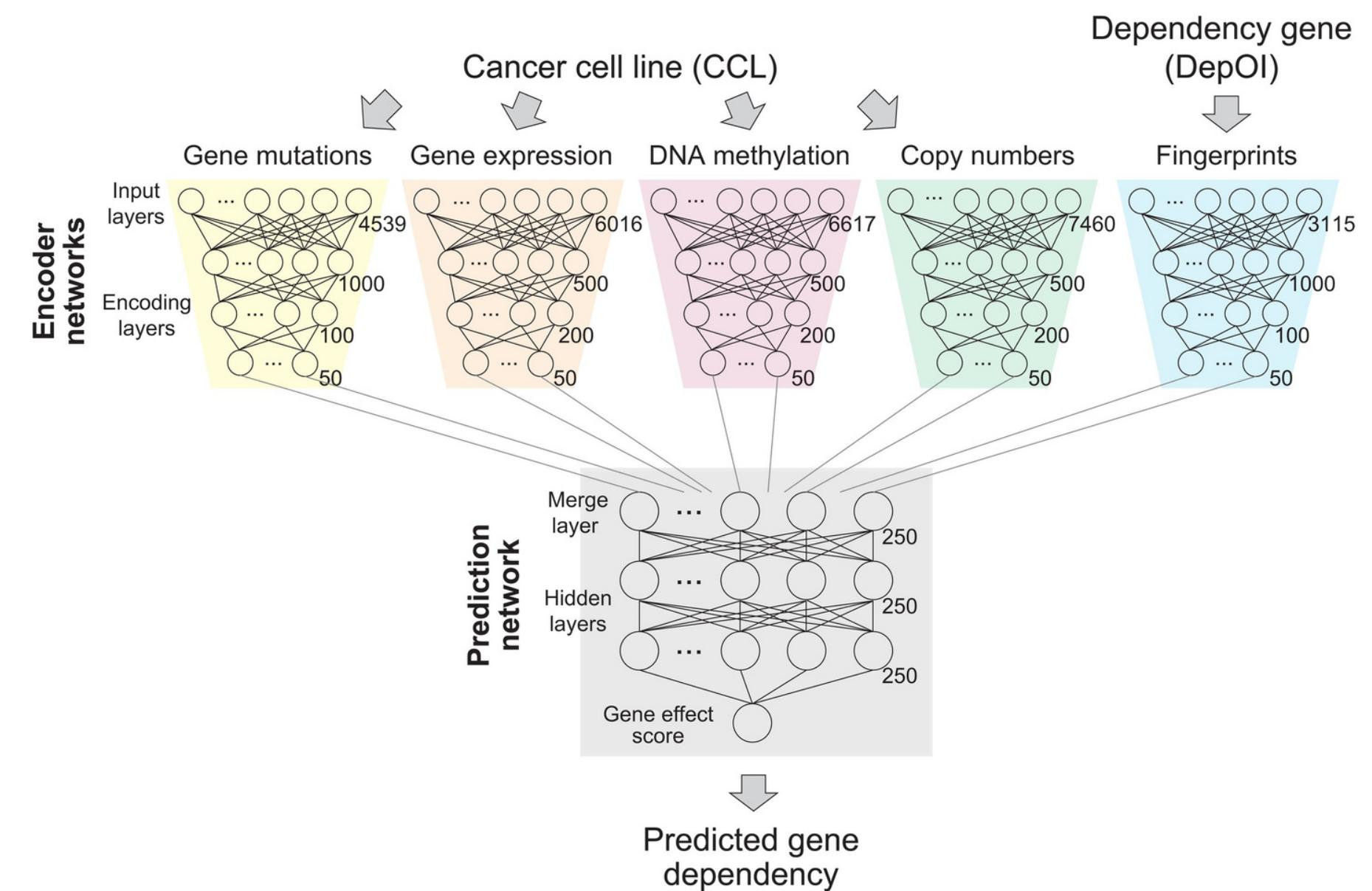
- Viene approfondita la fattibilità di uno strumento diagnostico basato sul sequenziamento di nuova generazione (NGS) per identificare il tipo di tumore
- Vengono utilizzate tecniche di deep-learning per valutare se un semplice protocollo di sequenziamento e analisi del DNA possa essere un complemento utile alle tecniche esistenti nella determinazione del tipo di tumore



# Stato dell'arte

## 2 Predicting and characterizing a cancer dependency map of tumors with deep learning

- Viene esaminata la sfida di collegare le dipendenze tumorali alle composizioni molecolari delle cellule tumorali
- Introduce DeepDEP, un modello di deep learning progettato per prevedere le dipendenze tumorali utilizzando profili genomici integrati

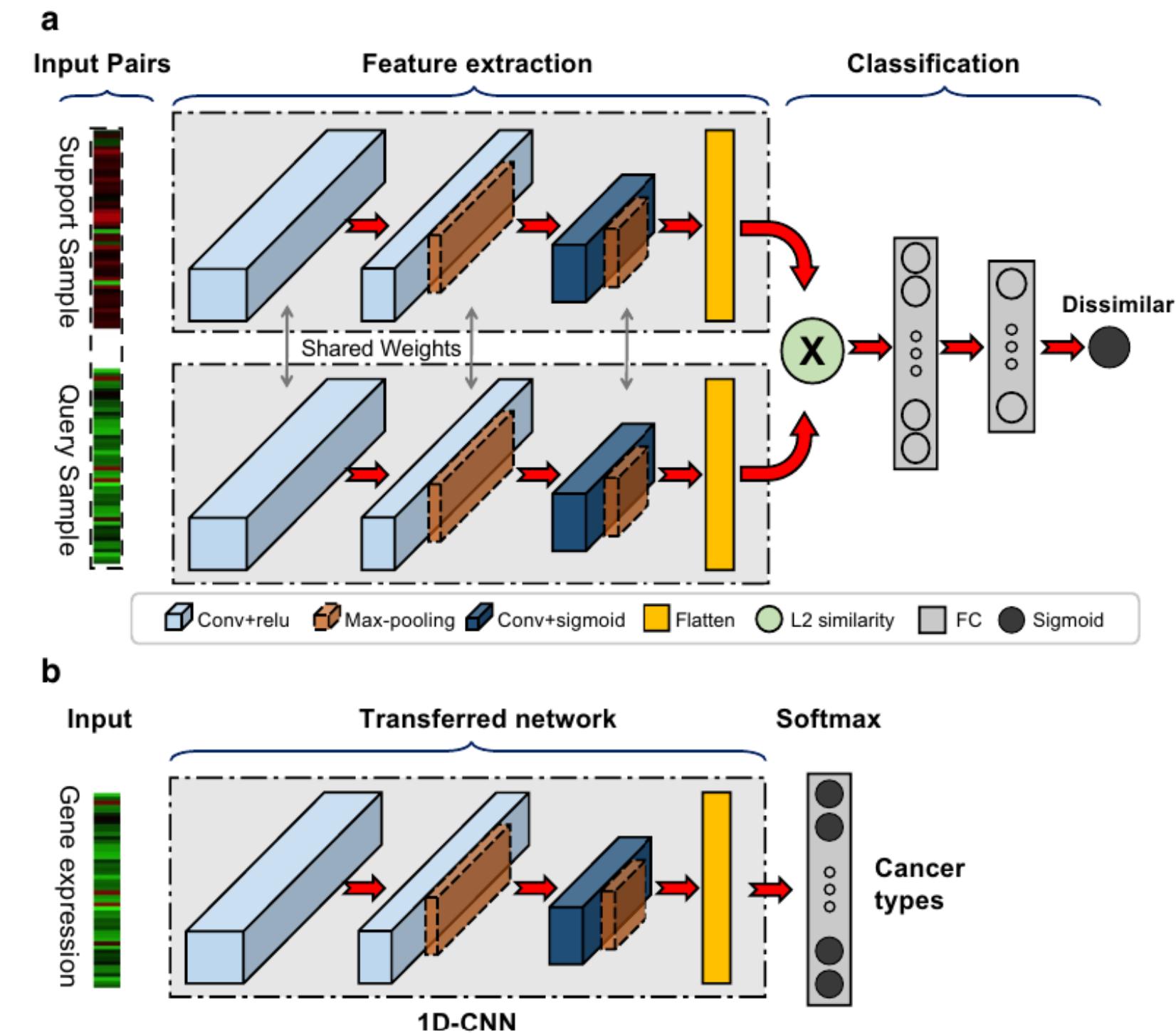


# Stato dell'arte

3

## CancerSiamese: one-shot learning for predicting primary and metastatic tumor types unseen during model training

- Viene proposto un modello SCNN, denominato CancerSiamese, che contiene due 1D-CNN identiche
- Per la metrica di similarità viene applicata una distanza L2 element-wise ai due vettori di feature generati dalle 1D-CNN
- Le reti CancerSiamese addestrate sono state testate per diverse N-way Predictions. Per una N-way Predictions la rete CancerSiamese ha confrontato un campione di query con un set di supporto di N campioni



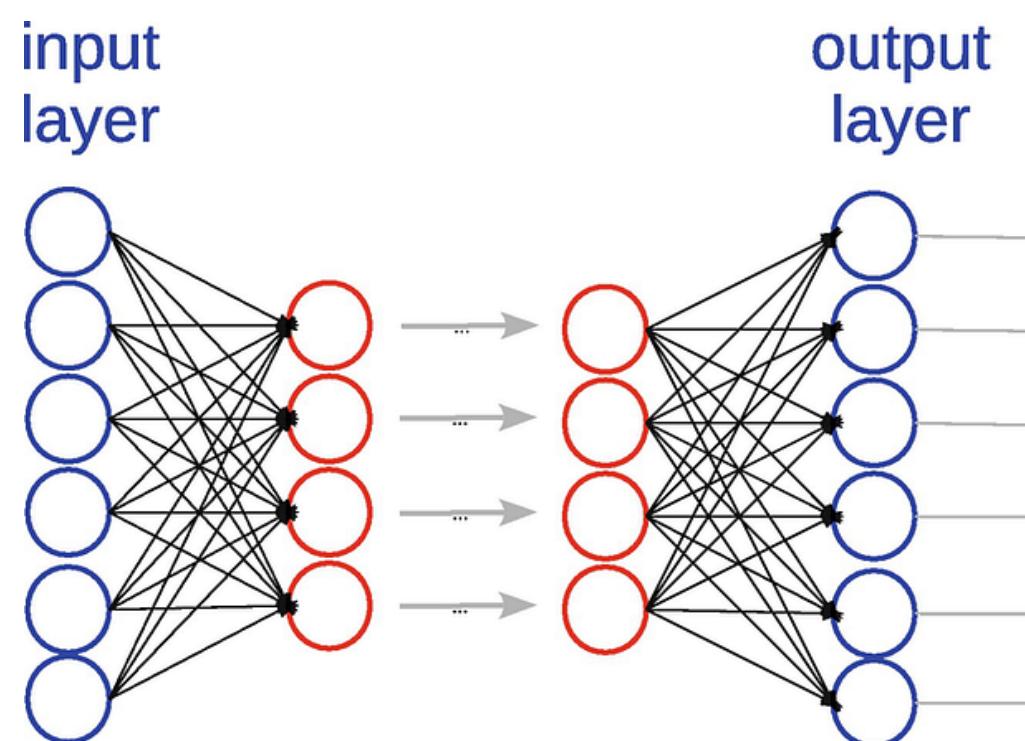
# Background

- **Feedforward neural network**

Una rete neurale artificiale è un modello di apprendimento automatico che comprende diversi strati di neuroni, unità di elaborazione ispirate a neuroni biologici.



Le reti neurali tipiche sono supervisionate e seguono il modello feedforward, basato sul percettrone.



In una rete neurale feedforward, ogni neurone nel primo strato riceve un valore reale in ingresso, lo moltiplica per un peso e trasmette il risultato a tutti i neuroni nello strato successivo.

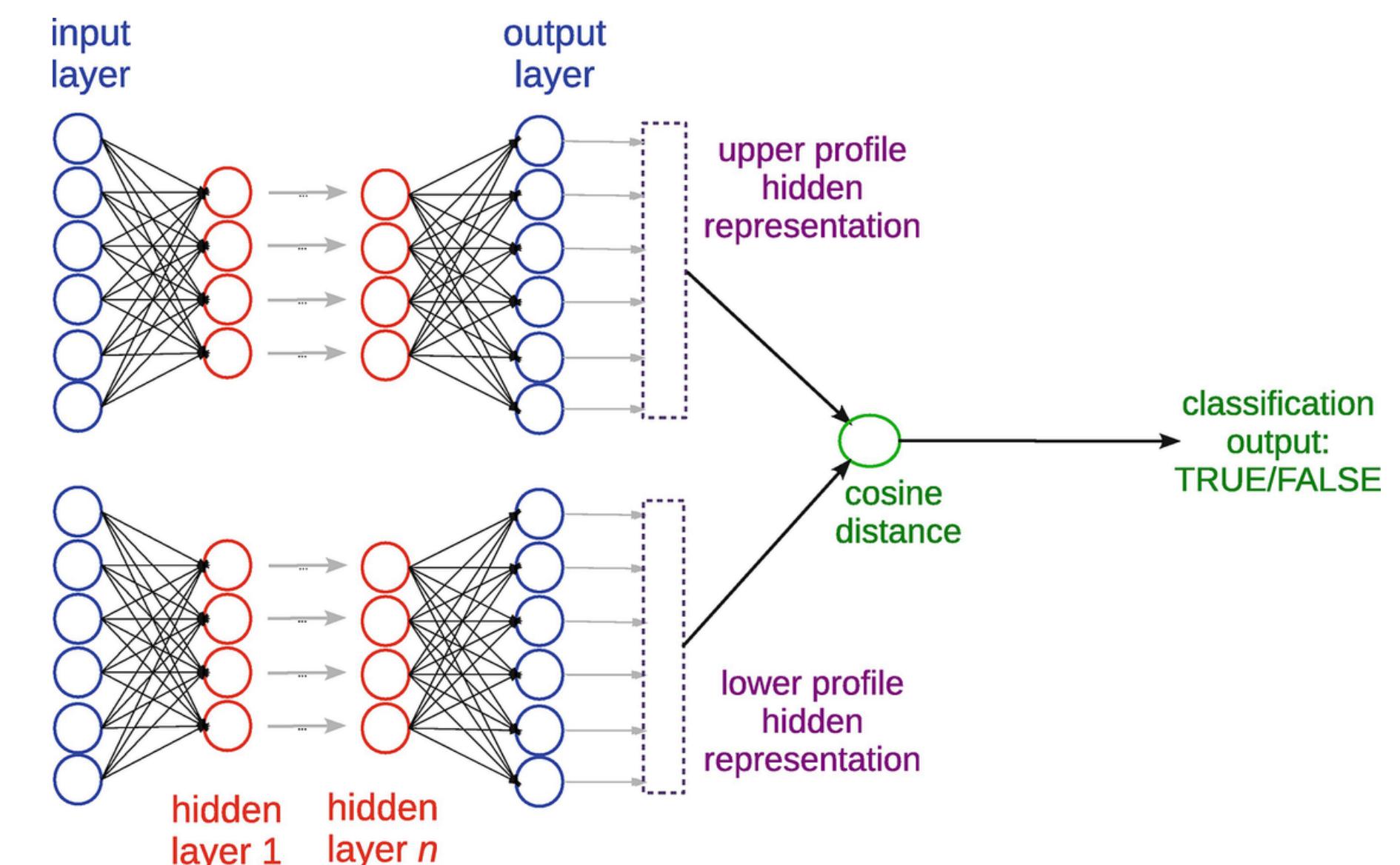


Durante il training, la rete neurale confronta i valori della rete neurale con il corrispondente valore e calcola l'errore statistico. In seguito, la rete neurale invia l'errore agli strati precedenti e aggiorna i pesi dei neuroni di conseguenza, attraverso la tecnica chiamata error-back propagation.

## • Siamese neural network

L'architettura di una rete neurale siamese consiste in due reti neurali feedforward identiche unite all'uscita.

Durante il training, ciascuna rete neurale elabora un profilo di valori reali attraverso i vari strati. La rete attiva neuroni in base a tali valori, aggiorna i pesi tramite la back propagation dell'errore e genera un profilo di uscita. L'output delle due reti neurali è confrontato attraverso una metrica di somiglianza.



TCGA BY THE NUMBERS

TCGA produced over  
**2.5 PETABYTES** of data

To put this into perspective, 1 petabyte of data is equal to

**212,000 DVDs**

TCGA data describes  
**33 DIFFERENT TUMOR TYPES** ...including  
**10 RARE CANCERS**

...based on paired tumor and normal tissue sets collected from  
**11,000 PATIENTS** ...using  
**7 DIFFERENT DATA TYPES**

TCGA RESULTS & FINDINGS

 MOLECULAR BASIS OF CANCER	Improved our understanding of the genomic underpinnings of cancer
 TUMOR SUBTYPES	Revolutionized how cancer is classified
 THERAPEUTIC TARGETS	Identified genomic characteristics of tumors that can be targeted with currently available therapies or used to help with drug development

THE TEAM



**20**

COLLABORATING INSTITUTIONS  
across the United States and Canada

## • The Cancer Genome Atlas (TCGA)

The Cancer Genome Atlas (TCGA) raccoglie molti tipi di dati per ciascuno degli oltre 20.000 campioni tumorali e normali. Ogni fase della pipeline di caratterizzazione del genoma ha generato numerosi dati, quali:

- Informazioni cliniche
- Metadati relativi all'analisi molecolare
- Dati di caratterizzazione molecolare

## • TCGA Barcode

Il codice a barre TCGA è l'identificatore principale dei dati biospecifici nell'ambito del progetto TCGA, poiché gli ID identificano in modo univoco una serie di risultati per un particolare campione prodotti da un particolare centro di generazione dei dati (ad esempio, GCC, GSC o GDAC).

WHAT'S NEXT?

The Genomic Data Commons (GDC) houses TCGA and other NCI-generated data sets for scientists to access from anywhere. The GDC also has many expanded capabilities that will allow researchers to answer more clinically relevant questions with increased ease.

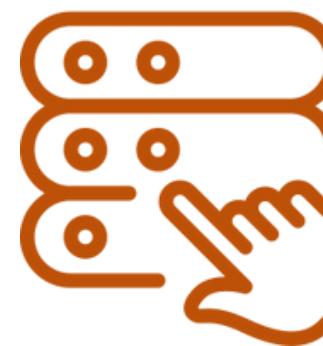


# Preprocessing dei dati



## Data Collection

Questa fase coinvolge la raccolta dei dati pertinenti al nostro contesto.



## Data Selection

In questa fase, si selezionano le caratteristiche rilevanti e significative per il problema in esame.



## Data Cleaning

Una volta raccolti i dati, è essenziale eseguire l'operazione di pulizia per eliminare errori, outliers o dati mancanti.



## Data Transformation

Questa fase include normalizzazione, standardizzazione e/o altre operazioni che modificano la distribuzione dei dati.



## Data Merge

Questa fase coinvolge l'unione di dati provenienti da diversi set di dati. Processo fondamentale per creare un dataset unico e informativo.

# Data Collection

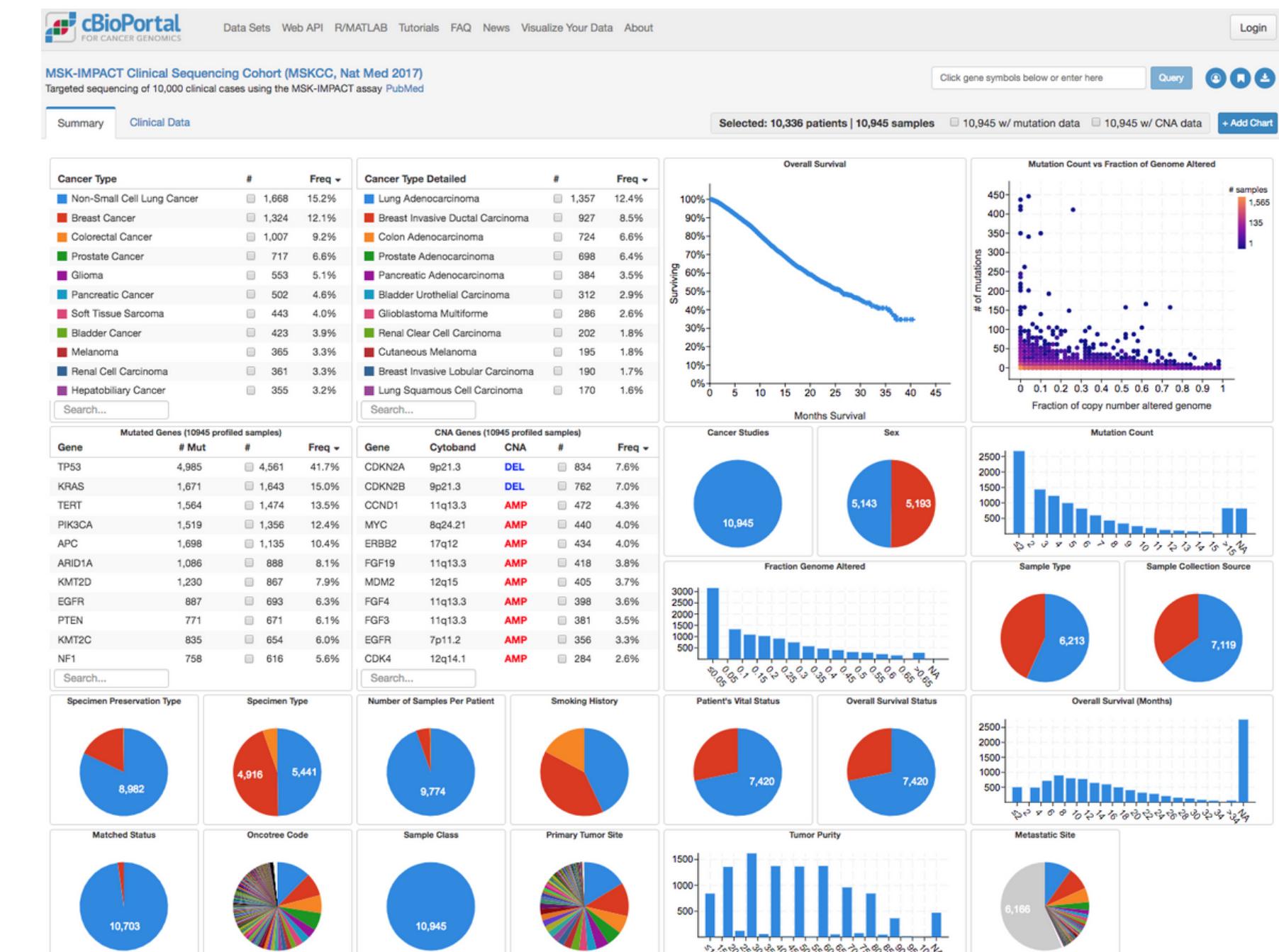


- **Fase 1**

**The Cancer Genome Atlas (TCGA)** rappresenta il dataset di maggiore rilevanza in questo ambito, ma l'accesso ai dati in suo possesso è particolarmente ostico.

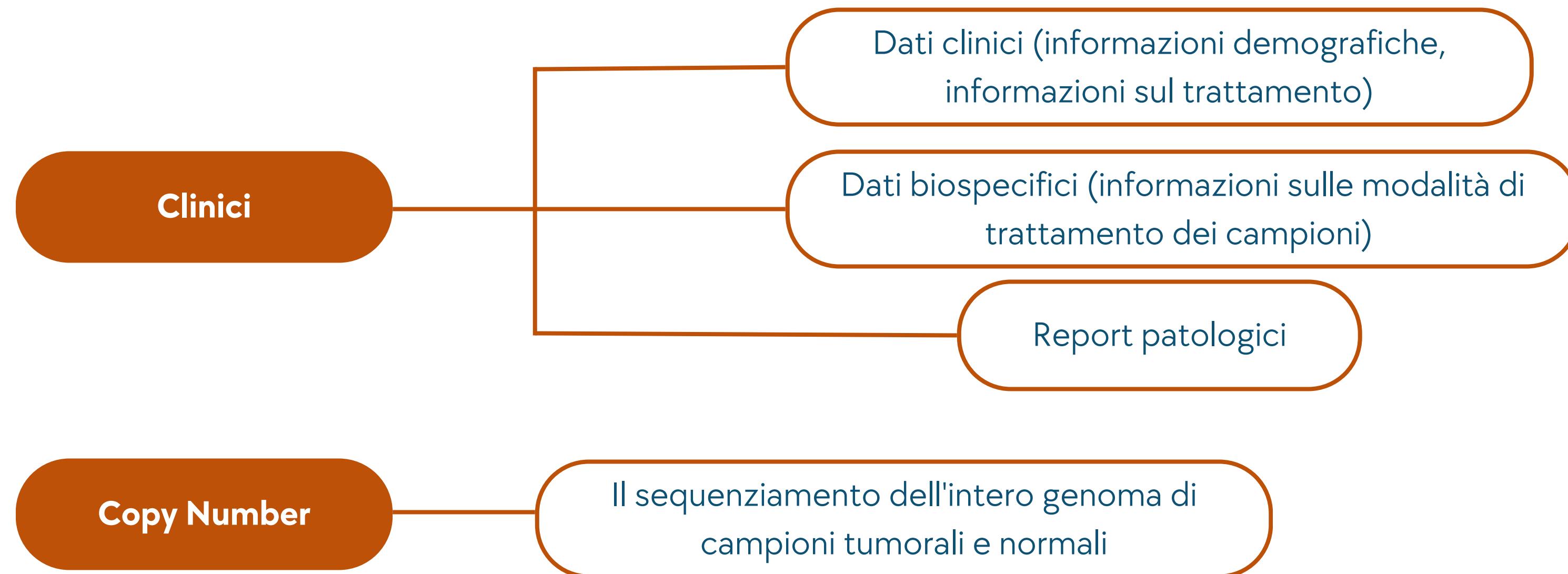
- **Fase 2**

Per colmare questa difficoltà, è disponibile **cBioPortal**, una risorsa open-access e open-source che facilita l'esplorazione interattiva di dati genomici sul cancro. In particolare, cBioPortal mette a disposizione un vasto assortimento di progetti che incorporano frammenti del dataset TCGA.



# Data Selection

Ciascun progetto comprende una varietà di file, suddivisi nelle seguenti categorie:



# Data Selection

**DNA**

Il sequenziamento dell'intero esoma di campioni tumorali e normali

**Metilazione**

Il sequenziamento dell'intero genoma eseguito dopo il trattamento con bisolfito dei campioni tumorali

**mRNA Expression**

Il sequenziamento di mRNA di campioni tumorali

**Protein Expression**

Permette di studiare l'espressione delle proteine coinvolte nei processi cellulari coinvolti nella formazione e progressione dei tumori

# Data Selection

Nel contesto di questo studio sono stati selezionati soltanto alcuni file tra tutti quelli disponibili, ovvero:

- **data\_clinical\_patient**, che contiene i dati clinici sul paziente (come Patient ID, sesso e stato del tumore)
- **data\_clinical\_sample**, che contiene i dati riguardanti i campioni tumorali
- **data\_cna**, che contiene informazioni sulle variazioni nel numero di copie di segmenti specifici del DNA
- **data\_methylation**, che contiene informazioni sulla metilazione del DNA
- **data\_mrna\_seq\_v2\_rsem**, che contiene il sequenziamento di mRNA di campioni tumorali
- **data\_mutations**, che contiene i dati sulle mutazioni ottenuti tramite sequenziamento dell'intero esoma
- **data\_rppa**, che contiene i dati sull'espressione delle proteine

Tipo di Cancro	Organo di riferimento	Pazienti
Adrenocortical Carcinoma	Ghiandola surrenale	79
Breast Cancer	Seno	1103
Cervical Cancer	Cervice uterina	306
Hepatobiliary Cancer	Fegato e vie biliari	409
Colorectal Cancer	Colon e retto	609
Bladder Cancer	Vescica urinaria	408
Esophagogastric Cancer	Esofago e stomaco	600
Prostate Cancer	Prostata	498
Head and Neck Cancer	Testa e collo	522
Renal Cell Carcinoma	Rene	891
Leukemia	Midollo osseo e sangue	173
Non-Hodgkin Lymphoma	Tessuto linfatico	48
Non-Small Cell Lung Cancer	Polmone	1014
Mesothelioma	Membrana sierosa	87
Ovarian Epithelial Tumor	Ovaie	300
Pancreatic Cancer	Pancreas	179
Pheochromocytoma	Midollare del surrene	147
Melanoma	Pelle e occhi	103
Thyroid Cancer	Tiroide	502
Thymic Tumor	Timo	120
Ocular Melanoma	Occhi	80
Glioma	Sistema nervoso centrale (Cervello e midollo spinale)	697
Miscellaneous Neuroepithelial Tumor	Varie localizzazioni nel sistema nervoso centrale	31
Soft Tissue Sarcoma	Tessuti molli (muscoli, tessuto connettivo, ecc.)	254
Endometrial Cancer	Rivestimento dell'utero (endometrio)	588
Nerve Sheath Tumor	Sistema nervoso periferico	9
		<b>9757</b>

# Data Cleaning

- **Data\_clinical\_patient**

1. **Age**, ovvero l'età alla quale è stato diagnosticato per la prima volta
2. **Sex**, ovvero il sesso del paziente. Alcuni tipi di cancro sono più comuni in un sesso rispetto all'altro
3. **OS status**, ovvero lo stato complessivo di sopravvivenza del paziente
4. **AJCC pathologic tumor stage**, ovvero l'estensione di un cancro, in particolare se la malattia si è diffusa dal sito originale ad altre parti del corpo

- **Data\_clinical\_sample**

1. **Sample ID**, ovvero l'identificatore del sample con riferimento al paziente
2. **Cancer type**, ovvero il tipo di cancro
3. **Sample type**, ovvero il tipo di campione (normale, primario, metastatico, recidiva)
4. **Somatic status**, viene utilizzato per descrivere le mutazioni genetiche
5. **Oncotree code**, ovvero un sistema di classificazione utilizzato per identificare e categorizzare i diversi tipi di tumori



# Data Cleaning

- **Data\_methylation**

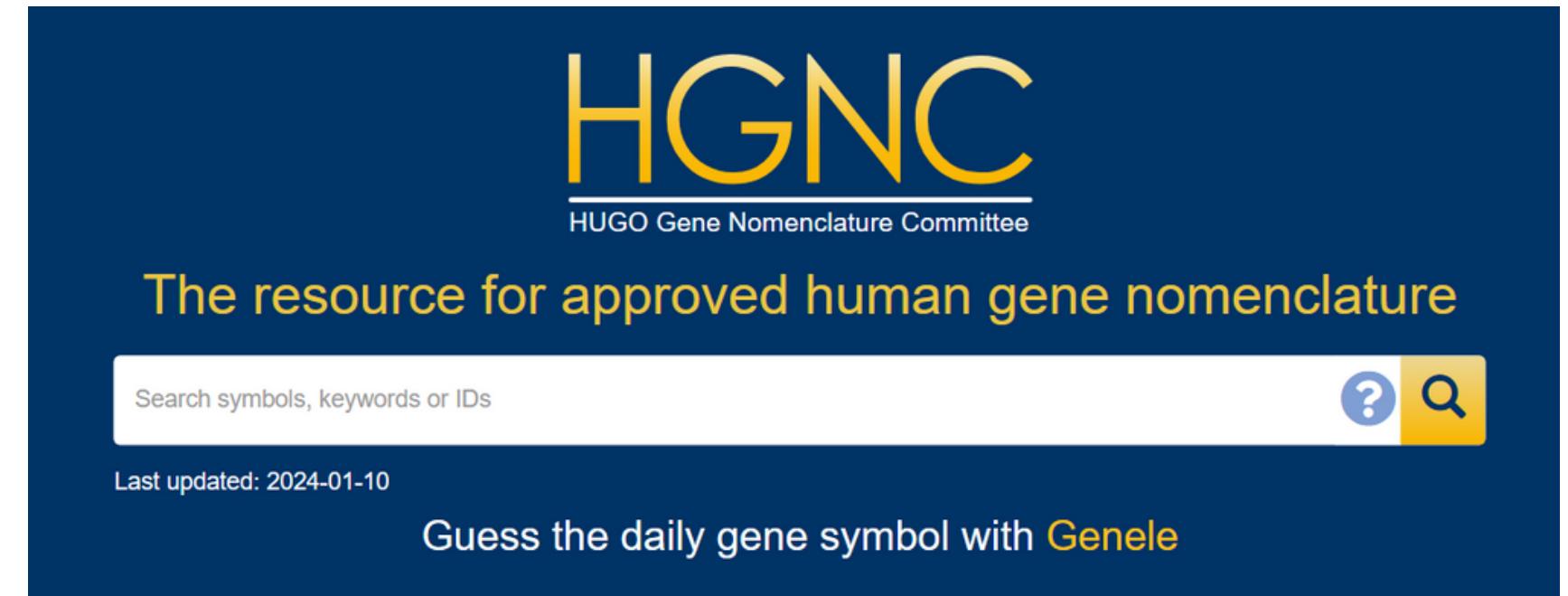
1. **Hugo Symbol**, ovvero un acronimo utilizzato per rappresentare un gene in modo univoco.
2. Le altre colonne rappresentano campioni diversi. Ogni colonna contiene valori che indicano l'espressione o i livelli di metilazione del gene corrispondente in ciascun campione

- **Data\_mrna**

Fornisce informazioni sull'espressione genica, ossia quanto RNA messaggero (mRNA) è stato trascritto dai geni in un campione biologico.

- **Data\_rppa**

Le misurazioni presenti nel file si riferiscono all'espressione delle proteine



# Data Cleaning

- **Data\_mutations**

1. **Hugo Symbol**, ovvero l'acronimo utilizzato per rappresentare un gene in modo univoco
2. **Chromosome**, specifica il cromosoma sul quale è stata rilevata la mutazione
3. **Start Position**, rappresenta la posizione di inizio della mutazione all'interno del cromosoma
4. **End Position**, rappresenta la posizione di fine della mutazione all'interno del cromosoma
5. **Strand**, indica il filamento del DNA coinvolto nella mutazione
6. **Consequence**, descrive l'effetto o la conseguenza della mutazione
7. **Variant Classification**, classifica il tipo di variante
8. **Variant Type**, specifica il tipo di variante
9. **Tumor Sample Barcode**, identifica i campioni di tessuto tumorale prelevati dai pazienti
10. **Matched Norm Sample Barcode**, identifica i campioni di tessuto normale (non tumorale) che sono stati prelevati dallo stesso paziente il cui tessuto tumorale è stato analizzato.



# Data Transformation

- **Conversione da TXT a CSV**

- **Normalizzazione**

Fondamentale perchè serve a garantire che le feature nel dataset abbiano una scala comune, in modo che il modello possa apprendere in modo efficace e convergere più rapidamente durante il training.

- **Split Methylation**

La maggior parte dei progetti include il file relativo alla metilazione con l'etichetta **hm450**, ma una piccola parte di essi contiene altri due tipi di file: la metilazione con l'etichetta **hm27** o la metilazione combinata **hm27\_hm450**.



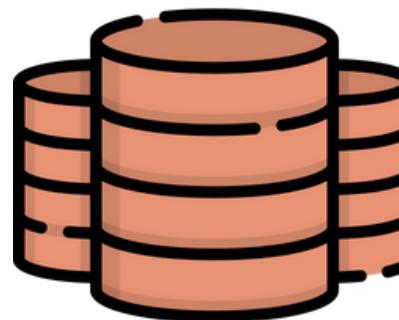
# Modello - Fasi preliminari



## Scelta dei geni

Per ciascun gene, viene calcolata la deviazione standard dei valori di espressione genica. Dopodichè vengono filtrati i geni con deviazione standard superiore ad una certa soglia:

- Soglia a **0.00005**, a cui corrispondono 12670 geni
- Soglia a **0.0001**, a cui corrispondono 7670 geni
- Soglia a **0.0002**, a cui corrispondono 3748 geni
- Soglia a **0.0005**, a cui corrispondono 1349 geni
- Soglia a **0.0010**, a cui corrispondono 610 geni
- Soglia a **0.0015**, a cui corrispondono 399 geni
- Soglia a **0.0020**, a cui corrispondono 275 geni
- Soglia a **0.0025**, a cui corrispondono 216 geni
- Soglia a **0.0030**, a cui corrispondono 166 geni



## Varianti del Dataset

Il dataset utilizzato per l'addestramento del modello è composto da un insieme di feature di tipo clinico e dalla sequenza di geni filtrata mediante la deviazione standard. Di questo dataset sono state costruite tre varianti:

- **Only Mutations:** le cui feature sono esclusivamente le espressioni geniche per ciascun paziente
- **Only Variants:** le cui feature sono date solamente dal conteggio del tipo di mutazioni per ogni gene (divise tra SNP, DEL, INS) e CNA
- **Mutations and Variants:** le cui feature includono sia le espressioni geniche che il conteggio del tipo di mutazioni

# Classificazione

La struttura del modello comprende:

- **Input Layer**
- **Strati di Convoluzione**, utilizzati per estrarre caratteristiche specifiche mediante filtri con dimensioni diversificate e funzione di attivazione ReLU
- **Strati di Pooling**, utilizzati per ridurre la dimensione spaziale dell'output dalla convoluzione, preservando le caratteristiche più significative
- **Flatten**, trasforma l'output tridimensionale dei layer precedenti in un vettore monodimensionale
- **Output Layer**, produce l'output finale del modello, con un numero di unità pari al numero di classi e attivazione softmax per ottenere le probabilità di appartenenza a ciascuna classe

Model: "classification"		
Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 332, 1)]	0
conv1d (Conv1D)	(None, 7, 256)	13056
conv1d_1 (Conv1D)	(None, 7, 128)	327808
max_pooling1d (MaxPooling1D)	(None, 3, 128)	0
conv1d_2 (Conv1D)	(None, 3, 128)	82048
max_pooling1d_1 (MaxPooling1D)	(None, 1, 128)	0
flatten (Flatten)	(None, 128)	0
dense (Dense)	(None, 26)	3354
Total params: 426266 (1.63 MB)		
Trainable params: 426266 (1.63 MB)		
Non-trainable params: 0 (0.00 Byte)		

# Rete Siamese - Funzioni principali

## 1 GET\_SIAMESE\_MODEL

Responsabile della creazione del modello della rete siamese.

Il modello è composto da due reti neurali identiche, ciascuna delle quali elabora uno dei due input.

Vengono definiti due input chiamati **left\_input** e **right\_input**, ciascuno con la forma `input_shape`.

Gli output delle due reti siamesi vengono calcolati separatamente per `left_input` e `right_input` e successivamente passati attraverso la funzione **last\_layer** per calcolare la distanza o somiglianza tra le rappresentazioni.

```
1 def get_siamese_model(input_shape, model):
2     # Define the tensors for the two input images
3     left_input = Input(input_shape)
4     right_input = Input(input_shape)
5
6     # Convolutional Neural Network
7     siamese_model = Sequential()
8
9     siamese_model.add(
10         Conv1D(filters=256, kernel_size=50, strides=50, activation='relu', weights=model.layers[1].get_weights(),
11                padding='same', input_shape=input_shape))
12     siamese_model.add(
13         Conv1D(filters=128, kernel_size=10, strides=1, activation='relu', weights=model.layers[2].get_weights(),
14                padding='same'))
15     siamese_model.add(MaxPooling1D(pool_size=2))
16     siamese_model.add(Conv1D(filters=128, kernel_size=5, strides=1, activation='sigmoid',
17                             weights=model.layers[4].get_weights(), padding='same'))
18     siamese_model.add(MaxPooling1D(pool_size=2))
19     siamese_model.add(Flatten())
20
21     encoded_l = siamese_model(left_input)
22     encoded_r = siamese_model(right_input)
23
24     # Connect the inputs with the outputs
25     siamese_net = Model(inputs=[left_input, right_input],
26                          outputs=last_layer(encoded_l, encoded_r, lyr_name='L2'), # prediction and cosine_similarity
27                          name="siamese-network")
28
29     return siamese_net
```

# 2 GET\_BATCH

Utilizzata per creare un batch di coppie di esempi per l'addestramento della rete.

La funzione genera due array vuoti **pairs** e **targets**, dove pairs conterrà le coppie di dati e targets le etichette (1 per coppie della stessa classe, 0 per coppie di classi diverse).

La metà del batch è riservata per coppie della stessa classe, mentre l'altra metà è per coppie di classi diverse.

Per ogni coppia, viene scelta casualmente una classe dal dataset di addestramento e

- Per **coppie della stessa classe**, viene selezionato casualmente un campione da quella classe
- Per **coppie di classi diverse**, viene selezionata casualmente una seconda classe diversa dalla prima, e viene scelto un campione da quella classe

Le coppie e le relative etichette vengono aggiunte agli array pairs e targets che vengono restituiti e rappresentano il batch di dati di addestramento.

```
1 def get_batch(batch_size, x_train, class_train_ind, genes_len, md='train'):  
2  
3     categories = random.sample(list(class_train_ind.keys()), len(class_train_ind.keys()))  
4     n_classes = len(class_train_ind.keys())  
5  
6     pairs = [np.zeros((batch_size, genes_len, 1)) for i in range(2)]  
7     targets = np.zeros((batch_size,))  
8  
9     targets[batch_size // 2:] = 1  
10    j = 0  
11    for i in range(batch_size):  
12        if j > n_classes - 1:  
13            categories = random.sample(list(class_train_ind.keys()), len(class_train_ind.keys()))  
14            j = 0  
15  
16        category = categories[j]  
17        idx_1 = random.sample(class_train_ind[category], 1)[0]  
18  
19        pairs[0][i, :, :] = x_train.values[idx_1].reshape(genes_len, 1)  
20        if i >= batch_size // 2:  
21            category_2 = category  
22            idx_2 = random.sample(class_train_ind[category_2], 1)[0]  
23            pairs[1][i, :, :] = x_train.values[idx_2].reshape(genes_len, 1)  
24  
25        else:  
26            ind_pop = list(categories).index(category)  
27            copy_list = categories.copy()  
28            copy_list.pop(ind_pop)  
29            category_2 = random.sample(copy_list, 1)[0]  
30            idx_2 = random.sample(class_train_ind[category_2], 1)[0]  
31            pairs[1][i, :, :] = x_train.values[idx_2].reshape(genes_len, 1)  
32  
33    j += 1  
34    return pairs, targets  
35
```

# 3 TEST\_ONESHOT

Utilizzata per valutare l'accuratezza del modello su compiti di apprendimento one-shot. La funzione esegue K compiti one-shot.

Per ciascun compito, utilizza la funzione `make_oneshot_task` per creare coppie di test e supporto.

Il modello predice quale coppia del test dovrebbe appartenere alla stessa classe del supporto e l'accuratezza viene calcolata come la percentuale di compiti one-shot correttamente classificati.

```
1 def test_oneshot(model, genes_len, x_test, class_test_ind, N, k, verbose=0):
2     """Test average N way oneshot learning accuracy of a siamese neural net over k one-shot tasks"""
3     n_correct = 0
4     if verbose:
5         print("Evaluating model on {} random {} way one-shot learning tasks ... \n".format(k, N))
6     for i in range(k):
7         inputs, targets, true_category, list_N_samples = make_oneshot_task(genes_len, x_test, class_test_ind, N)
8         probs = model.predict(inputs)
9         if np.argmax(probs) == np.argmax(targets):
10             n_correct += 1
11     percent_correct = (100.0 * n_correct / k)
12     if verbose:
13         print("Got an average of {}% {} way one-shot learning accuracy \n".format(percent_correct, N))
14     return percent_correct
```

# Struttura della Rete Siamese

La rete siamese è composta da due reti identiche, dove ogni rete è un modello sequenziale con vari layer convoluzionali (Conv1D) e di pooling (MaxPooling1D).

Dopo i layer convoluzionali e di pooling, i dati vengono appiattiti (Flatten) e passati all'ultimo layer che calcola la distanza tra le coppie di input.

Model: "siamese-network"			
Layer (type)	Output Shape	Param #	Connected to
input_2 (InputLayer)	[None, 166, 1]	0	[]
input_3 (InputLayer)	[None, 166, 1]	0	[]
sequential (Sequential)	(None, 128)	422912	['input_2[0][0]', 'input_3[0][0]']
lambda (Lambda)	(None, 128)	0	['sequential[0][0]', 'sequential[1][0]']
dense_1 (Dense)	(None, 512)	66048	['lambda[0][0]']
dropout (Dropout)	(None, 512)	0	['dense_1[0][0]']
dense_2 (Dense)	(None, 1)	513	['dropout[0][0]']
=====			
Total params: 489473 (1.87 MB)			
Trainable params: 489473 (1.87 MB)			
Non-trainable params: 0 (0.00 Byte)			
=====			

# Analisi dei risultati

Sono state eseguite diverse sperimentazioni impiegando principalmente due dataset, ovvero quello generato con una soglia di 0.0030 e quello generato con una soglia di 0.0005.

Questi esperimenti possono essere suddivisi in due approcci distinti:

- Nella prima modalità, è stata utilizzata la medesima architettura di rete proposta nel paper CancerSiamese, conducendo diversi test esclusivamente basati sul dataset **Only Mutations** ed altri con il dataset **Mutations and Variants**
- Nella seconda modalità è stata utilizzata una rete più profonda, caratterizzata dall'aggiunta di due layer supplementari, ed è stata eseguita unicamente sul dataset **Mutations and Variants**

Dataset	Tipo di Dataset	batch_size	#Classi	#Pazienti	#Features	evalaute_every	n_iterazioni	n_val	Accuracy
0.0030	Only Mutations	26	26	9757	166	200	100000	1000	86.1 %
		128							86.6 %
		256							89.8 %
		512							91.1 %
		26			1349				91.1 %
0.0005									

Dataset	Tipo di Dataset	batch_size	#Classi	#Pazienti	#Features	evalaute_every	n_iterazioni	n_val	Accuracy
0.0030	Mutations and Variants SNP DEL INS + CNA	26	23	5394	691	200	100000	1000	36.7 %
		128							36.4 %
		256							32.7 %
		512							29.7 %
0.0005		26	25	8796	5719				22.4 %

Dataset	Tipo di Dataset	batch_size	#Classi	#Pazienti	#Features	evalaute_every	n_iterazioni	n_val	Accuracy
0.0030	Mutations and Variants SNP DEL INS + CNA	26	23	5394	691	200	100000	1000	20.8 %
		128							74.5 %
		256							75.4 %
		26	25	8796	5719				13.5 %
0.0005		32						85.3 %	

Dalle sperimentazioni effettuate si può notare che un incremento della batch size porta ad un miglioramento dell'accuracy in quanto consente di elaborare simultaneamente un maggior numero di blocchi di dati durante la fase di addestramento.

# Conclusioni

Questo lavoro ha dimostrato la possibilità di prevedere tipi di cancro anche quando si dispone di un numero limitato di campioni.

Ciò potrebbe ispirare nuove e innovative applicazioni di soluzioni di apprendimento one-shot e few-shot per migliorare la diagnosi del cancro, la prognosi e la nostra comprensione dei meccanismi alla base del cancro.

**Grazie per  
l'attenzione!**

