POLITECNICO DI TORINO

Master degree course in Computer Engineering

Master Degree Thesis

A Data Analytics integrative approach for multi-omics clustering in Leukemia samples

Supervisors
Prof. Elisa Ficarra
Ph.D. Marta Lovino

Candidates
Stefano Nardella
matricola: S243846

Academic Year 2020-2021

A noi,
noi piccoli
e grandi guerrieri

Acknowledgements

Per un attimo chiudete gli occhi e immaginate.
Immaginate di essere in spiaggia, la sabbia umida e quella temperatura mite, quasi fredda che vi spinge a rimanere sull'asciugamano e non entrare in acqua, ma sapete che tutto quello che volete, è esattamente su quell'isoletta chiamata felicità.
Ho scelto questo posto per raccontarvi al meglio quello che in questi anni ho vissuto con sensazioni che tutti abbiamo provato.
Provate a entrare in acqua così di getto, chi mi conosce sa che non è da me.
La temperatura, la sensazione di infinito, di ignoto, spesso mi frenano.
Eppure, ogni volta che mi butto non voglio più uscire!
Ho immaginato il mio percorso esattamente così.
Io sulla spiaggia. Il mare. E quella temperatura fredda dell'acqua che mi spinge a non entrare.
Quella temperatura per me ha rappresentato una specifica paura, la paura del ritorno, di quei sogni e della difficoltà a risvegliarmi.
Quel catetere nel petto e quelle parole.
Quelle limitazioni, la privazione nel poter vedere gli amici, e la prova di chi, nonostante la paura, mi è rimasto vicino per quasi 25 anni.
E in questi anni voi, voi, che mi siete stati vicini, avete rappresentato il coraggio dii buttarmi,
Di non aver paura del freddo che poi tanto passa. Di non aver paura dell'infinito che poi alla fine è coraggio. E di amare, amare sempre, anche quando si è in mezzo alla tempesta, che poi forse, davvero, l'amore salva sempre.
E non importa averlo accanto, l'importante è portarlo con sé, perché l'amore vero non si perde in quelle onde.
E non importa avere la forza da soli, che qualcuno ti aiuterà sempre, ti darà un braccio che ti aiuterà a fare leva.
E non importa avere il coraggio perché qualcuno ti dimostrerà che... La nostra felicità è esattamente poco dopo le nostre paure.

La forza il coraggio e l'amore sono le tre linee che disegnano la rotta in questo immenso mare, come se fossero le indicazioni su una mappa.
E la temperatura di quell'acqua fredda, quella paura, è inulte che ve lo spieghi.
Ho imparato a crederci.
A credere nelle sconfitte, negli errori, nei difetti, nelle fragilità, nelle paure.
Ho imparato a credere cicatrici, nella paura di mostrarle, di raccontare la loro storia, nei lividi, nei silenzi e nel loro suono assordante.
Ho imparato a credere negli altri, nelle storie che non conosco, negli occhi di un bambino che si chiede "perché non posso giocare con i miei amici?".

Scrissi questi ringraziamenti tanto tempo fa, ma mi sento di volerli cambiare, esattamente come voi avete cambiato me.

Il primo ringraziamento vorrei farlo alla mia relatrice, la Professoressa Elisa Ficarra e a Marta, per la loro professionalità, l'umanità e la comprensione che hanno dimostrato verso di me in questo anno di tesi.
In particolare, a Marta vorrei dire grazie per le mille volte in cui mi ha recuperato dai miei momenti no, per le pettinate e per le motivazioni che mi ha dato per crescere durante questo percorso. Non è stato facile per me parlare di quello che fino a qualche anno fa era il mio incubo ricorrente e tu mi hai aiutato a mantenere sempre la lucidità nel farlo.

Un grazie ai miei genitori, che nonostante i mille litigi e le mille discussioni, anche se non direttamente, hanno dimostrato il loro orgoglio per i miei risultati. Li ho sempre saputi e di questo ve ne sono grato. Grazie per essere orgogliosi di me, per esserci ogni qualvolta che cado.

Un grazie ai miei nipoti, Francesco, Cristina ed Eleonora per essere la mia risata quando una lacrima marca il mio viso, spero tanto che un giorno vi possa davvero raccontare cosa siete per me, una vita che cresce, una speranza e l'amore incondizionato di uno zio. Mi rendo conto che siete stati voi a dare tanto a me e probabilmente poco io a voi, ma vi prometto che recupererò. Rappresentate per me il mio futuro. E io, nel futuro ripongo tutte le mie energie.

Un grazie a mia sorella Roberta per i mille scontri, per avermi insegnato che il bene permane sempre, anche dopo i mille conflitti.

Un grazie particolarmente grande a Francesca, Amica da sempre e sorella, per essermi stata vicina quando tutti mi hanno voltato le spalle per paura. Per avere avuto spesso più forza di me, per essermi vicina da quasi 24 anni e per avermi insegnato cosa vuol dire esserci sempre.

Un grazie alla mia famiglia, i miei zii e i miei nonni, di cuore.
Grazie a mia nonna Nona per avermi sempre coccolato, per esserti sempre preoccupata che mangiassi, che stessi bene. Per quelle fettine di vitello speziato che mi facevi arrivare in ospedale nonostante tutto.

Un grazie a miei angeli che oggi sono qua.

A nonna Emanuela per i suoi fantastici piatti di bucatini con le cime di rapa. Tu e nonno vi siete sempre prodigati affinché ci fosse il mio piatto preferito sul tavolo.
A nonno Antonio per aver creduto in me più di quanto lo facessi io, per la frase che mi hai detto la notte di Natale, per i mille insegnamenti, e per i mille messaggi che ogni giorno mi dai.
Per esserti commosso insieme a me quando ti dissi che mi sarei laureato per la triennale, per essere stato il mio primo abbraccio dopo quel mio piccolo traguardo.
A nonno Pino, angelo che ha lasciato una data particolare nella mia vita.
Mi venivi a prendere davanti alla scuola con il tuo ducato bianco, o con la tua punto con cui litigavi perché il navigatore ti indicava la strada che voleva lui. Grazie per essere entrato nei miei sogni quando più ne avevo bisogno, per avermi stretto le mani e avermi detto "andrà tutto bene, stai tranquillo".
I miei nonni sono stati coloro che interrompevano tutto pur di portarmi a scuola in ospedale, alle 7 del mattino erano sotto casa con la faccia ancora stropicciata ma con gli occhi pieni di amore.
Le mie nonne sono sempre state coloro che avevano il piatto pronto in tavola, segno di amore incondizionato.
L'amore che sono un nonno può dare ai nipoti, e che oggi rivedo negli occhi dei miei genitori per Francesco, Cristina ed Eleonora.

Un grazie ai miei zii.
A zio Pasquale e zia Raffaela, per essere sempre presenti. Una spalla su cui potersi confidare, per essere la mia seconda casa quando voglio staccare quel benedetto interruttore, per esserci sempre, per aver asciugato tante mie lacrime e per essere tutto quello che c'è di più in un rapporto zii-nipote.
Un grazie a zio Mauro per essermi sempre stato accanto, come amico, come

fratello maggiore e poi come zio, nonostante i mille confronti, il bene di fondo c'è sempre stato. Per essere così ansioso, e testardo da mandare in tilt pure me, ma per essere esempio di coraggio. Per aver avuto fiducia in me e nei miei sentimenti, per avermi reso partecipe delle tue più grandi scelte e per tutto l'affetto che mi dimostri.
Un grazie a zio Flavio per essere stato un confidente, per avermi consolato, per essere sempre pronto ad ascoltarmi. Per essere stato in primis un amico a cui poter dire tutto, un confidente a cui poter rivelare un segreto e uno zio per avermi fatto ragionare ogni volta che ero in crisi.
Grazie a mia zia Rosalba, per esserci anche se non ci sentiamo spesso. Per pensarmi anche se non mi faccio sentire, grazie.

Un grazie ai miei cugini Francesco e Mirko.
Negli ultimi anni abbiamo recuperato e restaurato un rapporto stupendo.
Un grazie lo devo a voi con cui sono cresciuto, si dice che i cugini siamo la metà strada tra un amico e un fratello, cugini di sangue fratelli per scelti.
Voi che siete stati un supporto per me più di quanto voi possiate immaginare, per essere stati quei fratelli di sangue, per essere stati amici prima che cugini.

Robertina, beh, un grazie mi sento di farlo anche a te... Tutto ciò che ti dico è qualcosa che già sai.
Abbiamo parlato tanto, tanto ed è come se ci fossimo capiti fin da subito, sai quanto mi sei stata vicina e te ne sono grato.

Un grazie alla mia Madrina e al mio Padrino, per essere stati le colonne della mia infanzia, per esserci anche oggi e per avermi coccolato quando ero più piccolo.

Un grazie alla mia seconda famiglia, UGI.
Voi per me avete sempre rappresentato qualcosa di più di semplici volontari.
Mi avete insegnato a vivere la malattia con normalità, ad affrontarla.
Siete il ponte perfetto tra due realtà così distanti, eppure siete stati capaci di costruirlo, pietra dopo pietra.
Avete dato valore a tutto ciò che mi faceva paura, semplicemente trasformandolo.
Un grazie in particolare a Manuela e Michela, mi ricordo ancora quando passavo dal piccolo ufficio del Regina Margherita per una cioccolata calda e poi finivo per mettermi dentro l'ovetto per girare come un pazzo, quando giocavo con il letto del centro trapianti mentre eravate in veranda.

Grazie per accogliermi sempre con un sorriso, per avermi visto crescere, per avermi supportato e anche sopportato.
Un grazie a Domenico per le mille ore al telefono, per la tua pacatezza, per esserci anche in piena notte quando ho un'idea, per ascoltarmi.
Un grazie a Pier per essere un supporto morale nei momenti in cui crollo e impazzisco.

E poi i miei compagni di viaggio, voi siete proprio l'esempio del "qualcosa di più di semplici volontari".
Voi siete quelli che per me sono gli Amici, siete il supporto, i pilastri, alcuni dei colori di quel quadro che chiamo vita.
Vedo i vostri sacrifici e vedo l'impegno che ci mettete in ogni vostra attività.
E sento che tutti questi sacrifici li fate anche per me.
Grazie di cuore a Fabio, Claudia, Federica, Giada, Paola, Daniele, Cecilia, Alice.
Grazie per essere sempre al mio fianco, grazie per le interminabili chiamate, grazie a chi durante il lockdown mi ha sopportato andando a correre, con un bicchiere di vino o
semplicemente con una chiamata.
Grazie per avermi compreso, per volermi così bene e per insegnarmi ogni giorno qualcosa di nuovo di quel mondo.

Un grazie ai miei volontari storici con cui sono cresciuto, coloro che mi portavano un sorriso in camera e oggi, lo portano nella mia vita: Silvia, Pino, Sandra, Mauro, Carla

Voi tutti rappresentate tutto ciò che di bello mi ha lasciato quel periodo, siete parte del mio cuore.
Siete la mia leggerezza per vivere quel periodo, siete il mio cuore Azzuro e si sa, senza cuore non si vive.

Grazie a Giovanna, per gli attimi di leggerezza estiva, per le cantate a squarciagola sui gradini e per accogliermi sempre con un immenso abbraccio.

Un grazie ai miei compagni di viaggio del Poli, per un tratto o per tutto il percorso.
In particolare, grazie a Gianna per essere la mia razionalità, per esserti arrabbiata tutte le volte che dicevo "non ce la faccio, ora mollo", per avermi accolto in casa sua, per gli abbracci e l'affetto.

A Yle per aver alleggerito le giornate di studio con una chitarra, per essere stata mia sostenitrice e avermi bacchettato insieme a Gianna in ogni mio momento no e per aver condiviso con lei l'affetto per me.
A Carlo, per aver condiviso con me gran parte di questo percorso, per essere stato insieme a Yle e Gianna una spalla in questi anni.
A Giuseppe, per essere stato prima compagno di laboratorio, Amico di bevute e di vacanze, per essere tremendamente testardo come me e per esserci per ogni follia.
A Laura per essermi sempre vicina, anche a migliaia di chilometri di distanza, per essere stata compagna fissa di lezione e di studio a Torino, per le risate e per gli immensi esami che abbiamo preparato insieme.

Grazie a Ste, gemello separato alla nascita con cui ho condiviso ansie, ore di studio e tante lezioni di informatica.
Grazie a Luca, per le poche ma intense chiacchierate e per ogni aperitivo finito a parlare di vacanze.

Grazie ad Arianna, per esserci sempre ogni sera in cui mi sento di fare due immense chiacchiere sul tuo balcone, per le serate in cui mi sono autoinvitato a casa tua, per tutti i supporti che ci siamo dati durante il primo lockdown.
Grazie a Dario, per il tuo istinto protettivo, per avere sempre una parola di conforto, un abbraccio o anche solo una battuta.
Grazie a Roberta, Davide e Francesca, Amici freschi freschi che hanno saputo prendersi un posticino tra i miei affetti più cari fin da subito.

Un grazie alle mie maestre delle elementari, Carmen e Maria, per essere state al mio fianco sempre durante la tempesta.
Un grazie alla Maestra Francesca, per essere stata la scintilla di tutto ciò, per avermi donato quel draghetto che oggi sta prendendo il volo, proprio tra queste righe. Te ne sarò per sempre riconoscente.
Credo che questo sia l'insegnamento più grande che tu mi abbia dato, il classico insegnamento che inizialmente non comprendi ma con gli anni diventa sempre più chiaro.

Un grazie a tutte le mie guide delle scuole superiori.
In particolare, vorrei ringraziare la mia Prof di matematica del biennio, Lorena.
Mi hai accolto in casa come un figlio, anche se dopo più di 10 anni riesci a chiamarmi ancora per cognome.

Grazie per le interminabili chiamate, per esserti sempre interessata di come andassero gli esami, di quanto mi mancasse e di come stessi.
Grazie a te, Luigi, Eugenio e Riccardo per avere sempre la porta di casa vostra aperta per me.
Grazie a Riccardo e Chiara per essere sempre un conforto, per essermi vicini in ogni momento, per le serate passate a giocare davanti a una birra e per avermi dimostrato una particolare vicinanza in alcuni momenti importanti della mia vita.

Un grazie vorrei farlo a Peppe, per avermi compreso, apprezzato e asciugato alcune lacrime di cui nessuno potrà avere il fazzoletto, per avermi insegnato che ogni orsetto può apprezzare il suo miele, che non importa come, l'importante è che "Fai rumore" e per avermi fatto capire che forse è meglio essere ancora qui di tutto il resto.

Grazie a Michele, per le passeggiate, le chiacchierate, le mangiate di sushi. Grazie per ogni nostra chiacchierata in cui mi hai fatto riflettere.
Grazie a Daniele, per esserci stato la sera prima di quell'intervista che ha significato tanto per me, per avermi aiutato a esternare al meglio quello che avevo dentro, per quel bigliettino, e per quei piccoli dettagli a cui anche tu fai caso.

Grazie a Nick, per avermi insegnato l'importanza della sincerità, del voler bene e per aiutarmi in quello che sarà l'epilogo di questo traguardo.

Grazie a Sandy, per avermi ascoltato le mie paure, per essersi confidata, per ricordarmi una persona molto importante della mia vita, per essere così semplicemente travolgente nelle emozioni, per tutti quei tasselli in comune.

Un grazie a tutti i miei colleghi, a tutta la Direzione ICT della Rai.
Grazie a Marco V., Monica P., Monica R., Egidio e Marco R. che mi avete dato la possibilità di raggiungere questo traguardo a me tanto caro, anche facendomi un po' di spazio per studiare e caricandovi voi.
Grazie a Michela, collega e Amica, grazie per aiutarmi nei miei momenti no a lavoro e non. Per le nostre call infinite, per le nostre cene e i nostri aperitivi.
Grazie anche a chi mi ha sopportato per ogni mio "permesso esame", per ogni mia ansia per l'esito, Donatella, Micaela e Luciana.
Vorrei tanto promettervi che non lo farò più, ma credo che dovremo rimandare la promessa.

Grazie a Roberto, so che ci sei anche tu oggi.

Un grazie alla professoressa Franca Fagioli, a tutta la sua equipe, a tutto il Reparto di Oncoematologia Pediatrica dell'Ospedale Regina Margherita di Torino, alle infermiere, agli infermieri, a tutto il personale Oss e tutti coloro che si sono presi cura di me.
Grazie per accogliermi sempre con un sorriso, anche quando correvo come un pazzo nel corridoio del quinto piano.

Un grazie alla mia Psicologa, la dottoressa Peirolo per avermi aiutato a lavorare su di me. Per avermi dimostrato che è vero quello che lessi anni fa, "dallo psicologo ci vai quando hai il coraggio di affrontare i propri limiti".
Per avermi accompagnato in questo percorso con una professionalità e un'umanità inequivocabile. Per avermi fatto riflettere, per avermi aiutato a lavorare su di me senza paura di conoscermi.

Un grazie alla Fondazione Umberto Veronesi e a tutte le persone che mi hanno seguito, per avermi dato la possibilità di esprimermi, di raccontarmi e di essere portavoce di un'esperienza di vita.

Ho lasciato voi due al fondo, perché credo che il motivo per cui io vi debba ringraziare sia impossibile da spiegare e lo sapete anche voi.
Grazie a Giacomino e Giulia, per essere così simili a me.
Voi siete quei fratelli di avventura a cui non devo davvero spiegare nulla perché basta un semplice sguardo.
Grazie perché mi avete fatto capire che non essere capiti degli altri è la nostra più grande forma d'Amore verso di loro.
Grazie di essere così speciali.

Un paio di anni fa feci una promessa a una persona, promisi che sarei arrivato fin qua senza fermarmi. Ora so di aver mantenuto questa promessa anche se questa volta non potrà essere fisicamente vicino a me. Questo è anche per te Nonno.

Un ultimo grazie vorrei farlo allo Stefano di 17 anni fa.
A quel bambino che cercava qualcuno che gli spiegasse che un futuro ci sarebbe stato.
Vedi? Eccolo qua, è tra le tue mani!
Quel bambino che oggi ha capito che non c'è cosa più bella di emozionarsi.
Che anche il silenzio è una risposta, e alcune volte è anche la più saggia.
Che sensibilità non è sinonimo di fragilità.
Che non è vero che bisogna dire tutto ciò che si prova, alcune volte esserci è già una dimostrazione di ciò che si prova.
Quel bambino che oggi ha capito che tutto ciò che di bello si prova, bisogna provarlo Immensamente. Oggi quel bambino, ha trasformato la sua più grande paura nel suo sogno, e ci è riuscito anche grazie a voi.

Mi sento di dedicare questa tesi a tutti i piccoli e grandi guerrieri.
In particolare, la voglio dedicare a
Giacomino, Giulia, Irene, Andrea, Ivan, Francesca, Arens, Christian e Stefanino.

Abbiate sempre il Coraggio,la Forza e l'Amore per affrontare ogni paura.
Solo in quel momento, costruirete il vostro futuro.

Immensamente,
Stefano

Summary

In the last decades, the decrease in the cost of next-generation sequencing (NGS) technologies has allowed the widespread of many omics data (e.g., transcriptomics, proteomics, genomics).
This thesis focuses on a multi-omics approach to cluster patients so that similar ones are assigned to the same cluster, simultaneously considering all data sources.
In detail, the proposed method has been evaluated on patients affected by myeloid and lymphoid leukemias (AML, ALL).
The method considers two types of transcriptomics data, miRNA and mRNA expression.
In detail, the expression measures the quantity of the molecule (mRNA or miRNA in this case) in the sample, which is crucial in regulating transcriptional and post-transcriptional processes.
In the literature, many techniques based on multi-omics clustering of samples are presented. Among them, tools based on joint dimensionality reduction techniques -jDR- (e.g., JIVE and GCCA) should be mentioned.
The main issue of jDR techniques is that they are based on a direct computation of the distances between all the samples in the original input space.
In this thesis, the proposed technique overcomes the limit of computing the distances between samples, exploiting a neural network model.
The proposed method computes distances using pseudo-samples (also called centroids) generated by the neural network to identify the two classes, AML, and ALL diseases.
Indeed, the network's output is a matrix of centroids generated from the data distribution of the input omics.
The method is based on a Multi-Layer Perceptor (MLP) architecture which takes as independent inputs the miRNA and mRNA expression matrices. In this sense, the method can be defined as a multi-input approach.
The network is made up of 2 hidden layers for each input omic.
Finally, the last hidden layers of each omic are concatenated and sent to the

final output layer.
A custom loss function is implemented to minimize the error between the output value and the actual value.
Different custom loss functions have been considered. In the end, the final loss is based on the Mean Squared Errors (MSE) computed on both input omics, which are combined through the sum divided by the product of the mse.
In addition, it is not necessary to have a Y label given as input in the training phase. Indeed, the proposed method computes an 'artificial' Y label from the expression values of mRNA and miRNA input matrices. This contribution is beneficial since the Y label is not always known for this kind of problem.
For each patient, the artificial label is computed as the average expression values of all its features.
This choice is consistent with what is mentioned in the literature.
The output of the neural network is a matrix that for each omics outputs the centroids. Since this matrix is in the same dimensional space as the input features, computing distances between patients and centroids, I assigned all the samples to the closest centroid.
Unlike jDR methods, the proposed approach does not compute the distances between all the patients but between patients and centroids.
This computation generates a dataset that contains the patient-centroid association.
In the end, a similarity matrix is computed. This matrix is squared and binary. In detail, the value is 1 if the two samples belong to the same centroid, 0 vice-versa. Then, I applied KMeans, Spectral, Gaussian Mixture and Hierachical clustering techniques both on the similarity matrix and the original data, with and without a PCA dimensionality reduction.
In the end, a custom evaluation function was designed to evaluate the performance of the clustering techniques. In detail, it verifies if the clustering technique has correctly matched the cluster label, counts the correct matches, and returns a compatibility percentage.
This metric increases about 20% in the clustering applied on the input omics and those with the PCA on the similarity matrix.

Contents

List of Tables

List of Figures

Chapter 1

Background

1.1 Leukemia

Leukemias are blood cancers comprised of rapid and uncontrolled growth of immature and atypical cells that infiltrate the bone marrow, where all blood cells are produced.
Leukemias make up 33-35% of childhood cancers; in children, about 80% of cases are acute lymphoblastic leukemia.
In particular, in recent years, innovative therapies have allowed an increase in the survival rate, leading to significant progress in healing, going from minus 10% -20% in the 90s to 80% in the current years.
These results are the result of preclinical and clinical research and the ability to work in a network.
In Italy, the network is represented at the Italian Association of Pediatric Hematology and Oncology (AIEOP), a body that collaborates with the leading European centers.
The model conceived to work a network is represented by the "Diagnosis and treatment" model, a model which in Italy is supported by the Umberto Veronesi Foundation.
The research for new cutting-edge therapies is focused on precision medicine, science that evolves in the search for a drug that manages to go directly against an anomaly present only in the cancer cell or mainly in that cancer cell, causing the cell's death.
Precision medicine was born as a term in the 1950s and became a reality with the genomic analysis of cancer cells.
A particular result from genomic analysis is the treatment of chronic myeloid leukemia through oral therapy. Indeed, this disease was once only treated

with bone marrow transplantation.

Child and adolescent acute lymphoblastic leukemia is the most common pediatric cancer of children and adolescents and the most common pediatric cancer.
Every year in Italy, from 350 to 400 children and adolescents get sick from this disease. Acute lymphoblastic leukemia represents one example of the extraordinary success held in recent years precisely because today, more than 85% of adolescent children who fall ill with this disease become adults. It is the most excellent sign of success related to this disease.
The success of the therapy is linked first to the enrollment of adolescent children in diagnosis and treatment protocols. In Italy, it is guaranteed by the connection between the Italian Association of Pediatric Hematology and Oncology, AIEOP. It is so in Italy as in all over the world. Therefore, the first essential element of success is that all children are enrolled in diagnosis and treatment protocols. It has made it possible over the years to continuously improve both from a diagnostic point of view, recognition of groups with different prognoses, and in offering innovative therapeutic strategies that have been the reason for this success.
The therapies are chemotherapy, rarely radiotherapy, only in cases of recurrence of the disease the availability of bone marrow transplantation, but today indeed the expectations we have in the further improvement of treatments lie in the hope that immunotherapy, that is the manipulation and use of biological drugs, may also have in this disease.
The challenges of acute lymphoblastic leukemia today are on two fronts, on the one hand, that of being able to guarantee the children we care for a future of complete health, free of long-term side effects, which today precisely because more and more adolescent children are recovering, we discover, we identify, and we must try to prevent.
On the other hand, we have to deal with the 15% of adolescent children who do not survive today, and to whom we must try to offer innovative strategies both in the early definition of which are those cases of those adolescent children who are most at risk of presenting, a recovery of the disease.
In this perspective, genomics offers us unique perspectives to describe every single disease of every child, on the other hand, that of being able to have innovative therapeutic strategies.
In this context, immunotherapy, which today means monoclonal antibodies biological drugs, antibodies that interfere with the immune response but

above all Car T cells, genetically modified cells to be able to attack the disease selectively represent today a great hope for the future, also for these children who do not recover today.

1.2 Multi-Omics Approach

Multiomics is a new approach in which data sets from different omic groups are combined during analysis.
The different omics strategies employed during multi-omics are genome, proteome, transcriptome, epigenome, and microbiome.

- *Genomics*: Genomics is a field that encompasses the identification of genes and genetic variants associated with a disease or in response to certain drugs and medications. In this approach, large GWAS or genome association studies are used to identify genetic variants in a whole-genome associated with a disease.
  Genotyping is performed for thousands of people for nearly a million markers to identify significant differences in genetic markers between healthy and sick individuals. In addition to GWAS, genotype arrays, next-generation sequencing (NGS), and exome sequencing are also utilized in this approach.

- *Epigenomics*: Epigenomics refers to or identifies DNA-associated protein DNA modifications. These include acetylation/deacetylation and DNA methylation. The fate and functions of cells can be changed through changes in DNA and histones, in addition to genetic changes. These changes can be passed on to the offspring. Epigenetic changes in the genome can also act as markers for metabolic syndromes, cardiovascular disease, and metabolic disorders. These changes can be cell-and tissue-specific. Hence, it is critical to identify epigenetic changes during indigenous and sick people. Sequencing of the next generation is also used to underestimate DNA modifications.

- *Transcriptomics*: This approach is used to identify qualitative and quantitative levels of RNA throughout the genome. It includes which transcripts are present and the levels of their expression. Although only 2% of the DNA is translated into protein, nearly 80% of the genome is transcribed, including coding RNA, short RNA, including microRNA, small nuclear RNA. In addition to acting as an intermediate between DNA and

protein, RNA also has structural and regulatory functions during native and altered states. They have been shown to have a role in myocardial infarction, adipose differentiation, diabetes, endocrine regulation, neuron development, and others.
Hence, it is crucial to understand which transcripts are expressed at a time. Therefore, in addition to next-generation sequencing (NGS), a probe-following assay and aRNA are also used in this approach.

- *Proteomics* This field is involved in identifying protein levels, modifications, and registrations at the genome level. protein-protein interactions can be investigated with phage visualization, two classic yeast hybrid, affinity purification, and Chip-Sequence.
  Most proteins are regulated with post-translational modifications, such as phosphorylation, acetylation, ubiquitination, nitrosylation, and glycosylation.
  These changes are involved in maintaining cell structure and function. In addition, mass spectroscopy-based techniques are being used for global proteomic changes and the measurement of post-translation changes.

- *Metabolomics* Metabolome includes all metabolites present in a cell, tissue, or organism, including small molecules, carbohydrates, peptides, lipids, nucleosides, and catabolic products. It represents the finished product of gene transcription and consists of both signaling and structural molecules. The metabolome size is much smaller than the proteome size and is thus easier to study.

- *Microbiomics* Microbiomics consists of all the microorganisms of a community. Microbes have been found in human skin, mucous surfaces, and the intestine. The microbiome found in humans is very complex, where the gut consists of 100 trillion bacteria. The microbiota is involved in diabetes, obesity, cancer, colitis, heart disease, and autism.

Source: www.news-medical.net

With progress in all the different omics fields, it increasingly recognizes that an omics module cannot answer a research question.
The microbiome influences the expression of the protein and the gene, which in turn influences the metabolome and all these processes by interference and regulation.
Therefore, studying these treatments in their entirety to find strategies for treating diseases is of crucial importance.

This is where the field of multi-omics is coming in. This field encompasses all fields of omics and ranks to understand the native and altered state of an organism from the analysis of data from differential omics experiments.

In this thesis, we will deal with transcriptomic data: in particular mRNA and miRNA.

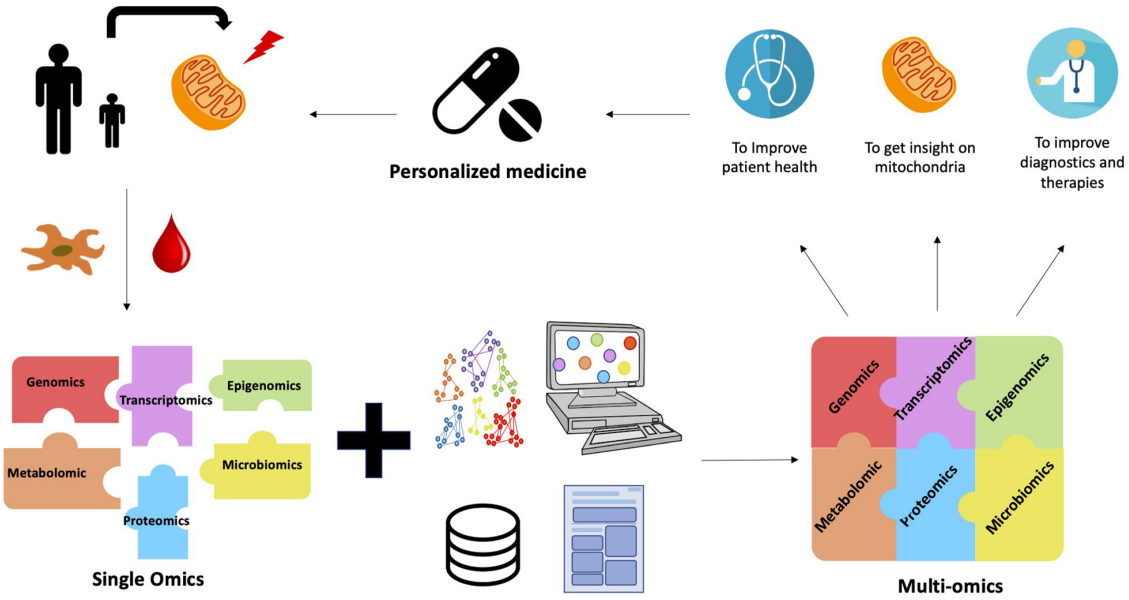


Figure 1.1. Multi-Omics Approach

Source: www.frontiersin.org

1.3 Neural Networks

The networks attempt to implement the functioning of the brain through a series of mathematical models, a continuation of the well-known neural machine of Turing, father of English mathematical intelligence and artificial intelligence. They coined with his thesis the computer model that everyone today we all know, the so-called Turing machine.
The idea of neural networks was born from a neurophysiologist and a mathematician who got together and had the idea of introducing a so-called neural network model, that is, the possibility of implementing a Turing machine through a model inspired by the human brain.

The neural network itself consists of many small units called "neurons".

These neurons are grouped into several layers. Units of one layer interact with all neurons of the next layer through "weighted connections," which are just connections with a real-valued number attached.
A neuron takes the value of a connected neuron and multiplies it with its connection's weight. The sum of all connected neurons and the neuron's bias value is then put into a so-called "activation function", which simply mathematically transforms the value before it finally can be passed on to the next neuron.
This way, the inputs are propagated through the whole network. That is pretty much all the network does, but the real deal behind neural networks is finding the correct weights to get the right results. This aspect can be done through a wide range of techniques such as machine learning.

Neural networks can be used in different fields of application, from speech recognition to economic credit risk, passing through image recognition.
In each of these areas, a machine cannot recognize what data represents, therefore relevant characteristics are generated that allow the computer to recognize the type of data processed thanks to the implemented algorithm, or it can discover it by itself through the data analysis.

A neural network is composed of three types of layers:

- Input Layer
- Hidden Layer
- Output Layer

Each of these layers is composed of a finite number of nodes. In detail, each node is connected with all the nodes of the next layer. These connections are "weighted" by multiplying factors in the algorithm, which represent the "strength" of the connection itself.

Initially, the results will be relatively error-prone. If the neural network receives feedback from a human trainer and can modify the algorithm, it is called machine learning. In deep learning, human training can be omitted. In this case, the system learns from its own experience and becomes better the more material it has available. The final result is an algorithm capable of identifying the type of data with minimal error, regardless of the value it contains

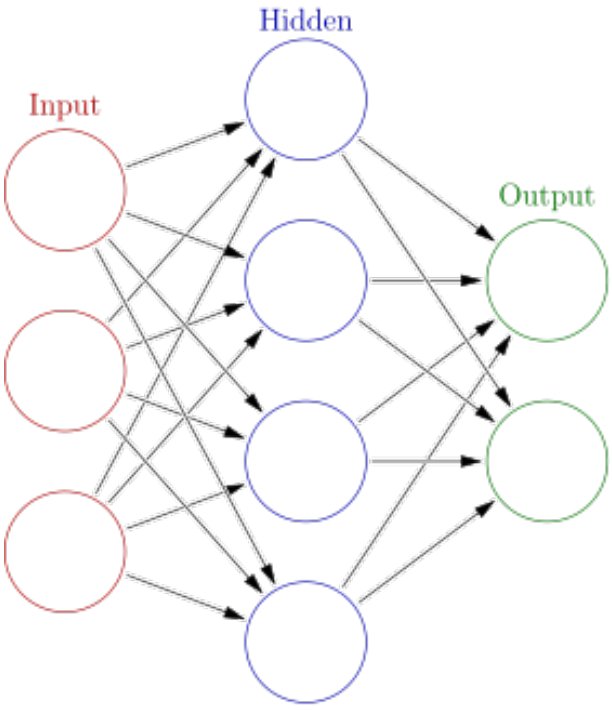


Figure 1.2. Example of Neural Network

Source: www.kdnuggets.com

1.4 Clustering

Clustering is the search for groups of objects such that objects belonging to one group are "similar" to each other and differentiate from objects in other groups. In particular, it consists of a set of statistical techniques aimed at identifying groups according to elements with characteristics that make them similar concerning a set of characters taken into consideration and a specific criterion. The objective is to group heterogeneous elements into several subsets that tend to be homogeneous and mutually exhaustive. In other words, the statistical units are divided into a certain number of groups according to their level of "similarity" evaluated, starting from the values that a series of selected variables assumes in each unit.

In this thesis, some clustering techniques are used. In particular, 4 clustering techniques are used:

- KMeans

- Spectral
- Gaussian Mixture
- Hierarchical

Below we briefly describe how these clustering techniques work.

1.4.1 KMeans Clustering

KMeans is a partition group analysis algorithm that allows you to divide a set of objects into k groups based on their attributes. The main functioning of this algorithm is as follows:

- Define the breadth of the data set and the k centroids randomly arranged
- Use each element and assign it to the nearest centroid
- Calculate the Euclid distance between each element and the centroid
  - The centroid with the minimum distance is taken from the element

$$argmin\ dist(C_i, x)^2 \tag{1.1}$$

- Updating of the centroids by averaging all elements that have been assigned to the new cluster

As long as there are no changes to the cluster centers.
The algorithm ends only in these cases:

- No data changes clusters
- The sum of the distances is at a minimum
- Number of iterations reached

This technique has the advantage of being fast but has some disadvantages such as assigning the initial centroids in a casual way and different results for each execution.

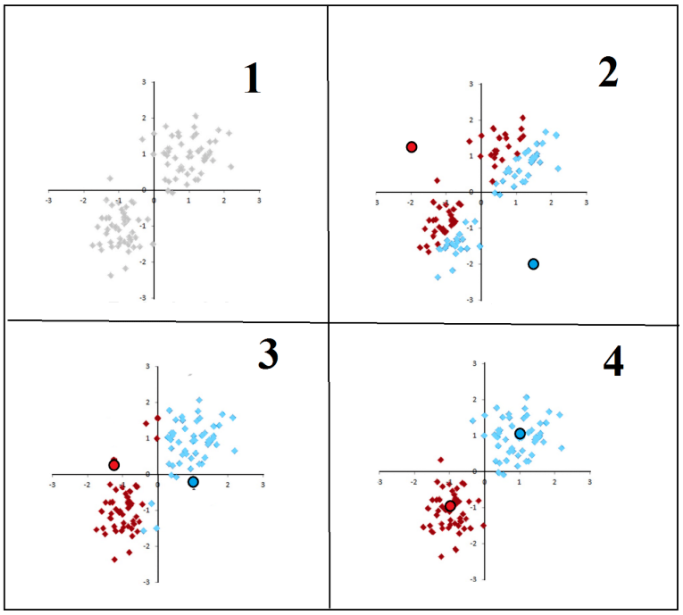


Figure 1.3. Example of Step in KMeans

Source: www.researchgate.net

1.4.2 Spectral Clustering

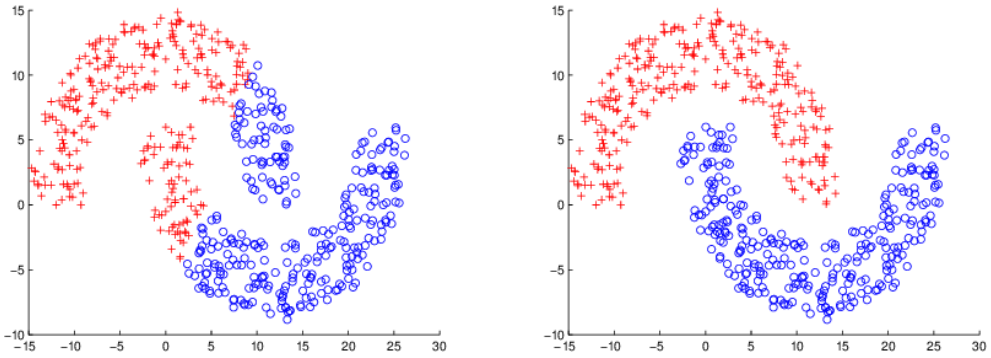
The critical principle of Spectral Clustering techniques is to consider i given as if they were the vertices of a graph and weight the connections based on the similarity between two vertices. This interpretation leads to the framework of the "Spectral Theory of Graphs", a theory in which the data of the training set can be considered as the approximation of a topological space (a manifold) whose properties can be studied through the spectral properties of a matrix called Laplacian. These properties, hence the name "Spectral", are used to characterize the graphs to proceed with an appropriate partitioning. Source: Alcuni metodi matriciali per lo Spectral Clustering - AMS Tesi ...http://amslaurea.unibo.it

The main steps of Spectral Clustering are

- Calculation of the Laplacian Matrix
- Calculation of the first K eigenvectors (and I k minor eigenvalues)
- Consider the matrix composed of the first k eigenvectors

- Clusters on graph nodes that use these features

As shown in the figure below, spectral clustering also works for data that cannot be linearly separated. In particular, seeing the application on a real dataset, it is visible how spectral clustering, in this case, can better group the data belonging to the same cluster.



(a) K-means (b) Spectral clustering

Figure 1.4. KMeans vs. Spectral Clustering

Source: www.researchgate.net

1.4.3 Gaussian Mixture Clustering

It is a clustering technique used primarily for unlabeled data.
It is important to remember that the KMeans method does not take into account the variance of the data distribution. Indeed, KMeans creates circles around the data, this methodology generates problems in the case of data distributed linearly as shown in the figure.

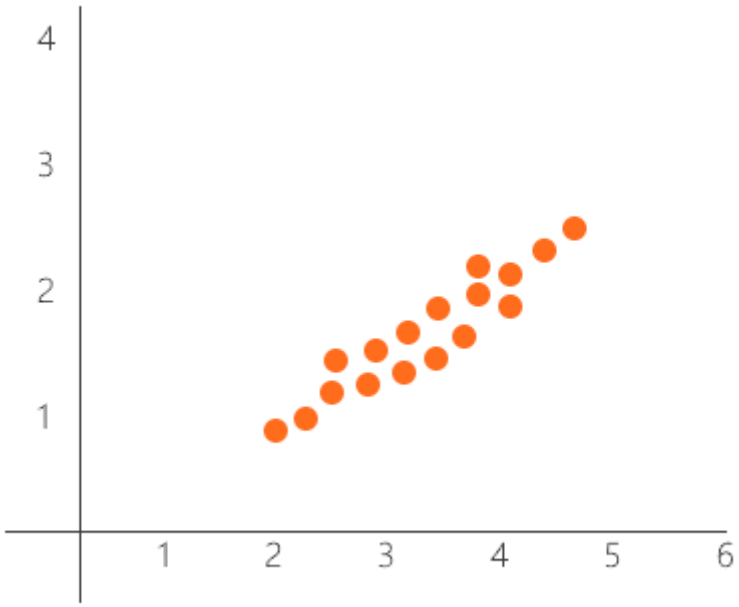


Figure 1.5. Linear distribution

In particular, KMeans tells us which data point belongs to which cluster but does not give us the probability that a given point belongs to each of the possible clusters.

The Gaussian Mixture instead of creating circles create ovals around the points. In detail, the Gaussian Mixture assumes that there are several Gaussian distributions and each of these is a cluster. This technique tends to group data belonging to a single distribution together, as shown in the figures.

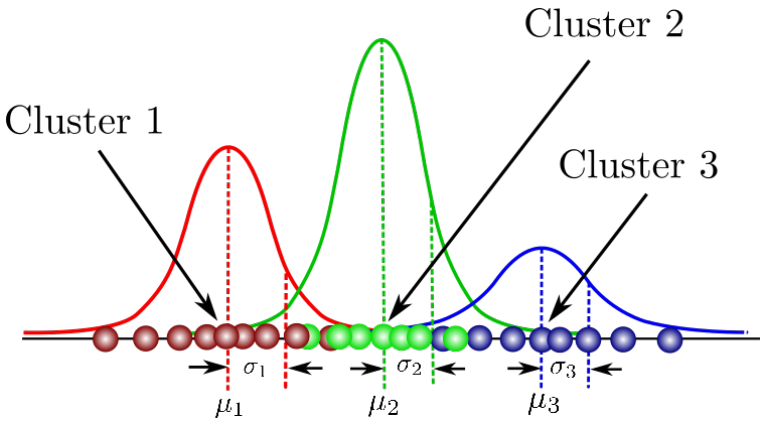


Figure 1.6. Gaussian Distributions

Source: www.towardsdatascience.com

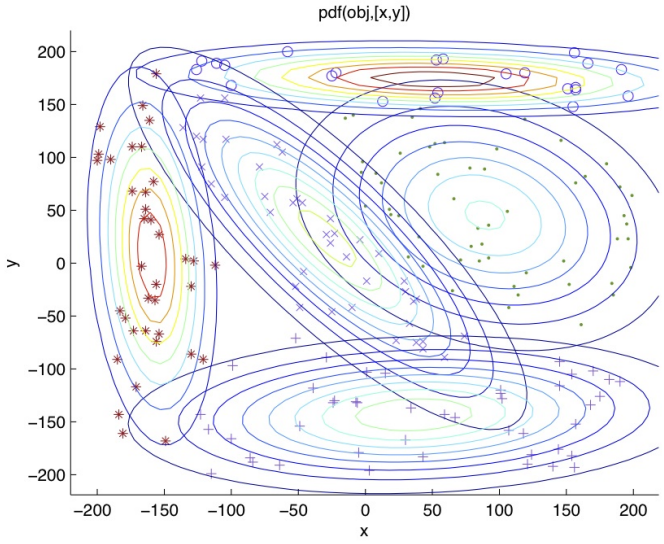


Figure 1.7. Clustering with Gaussian Mixture

Source: Stack Overflow

1.4.4 Hierarchical Clustering

This clustering technique assumes that each point is its own cluster. At each step, the closest pair of points are searched for in one of the following ways:

- minimum distance between points
- the maximum distance between clusters
- the average distance between the points of the cluster
- the average distance between the points of the cluster

and merges into a single cluster. you decompose objects into different levels of nested partitions In the end, clustering is performed by cutting the dendrogram to the desired level.

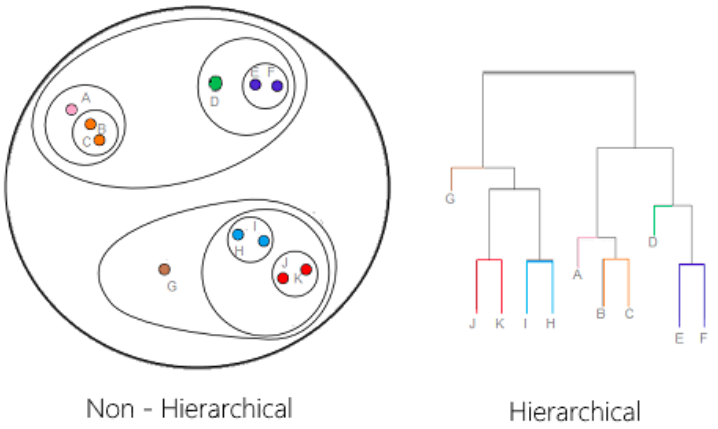


Figure 1.8. Hierarchical Clustering Explained

Source: www.vitalflux.com

Chapter 2

Data

2.1 Data Source

The data used in this research concern patients with myeloid and lymphoid leukemia.
All data files were downloaded from the GDC Portal Cancer. repository using the following filters:

- Primary Case: hematopoietic and reticuloendothelial systems
- Disease type: myeloid leukemias and lymphoid leukemias

To download the files related to each omic, the following filters were used:

- Copy Number:
  - Data Category: Copy number variation
  - Data type: Gene level copy number
- mRNA:
  - Data Category: Transcriptome Profiling
  - Data type: Gene Expression Quantification
- miRNA;
  - Data Category: Transcriptome Profiling
  - Data type: miRNA Expression Quantification
- Methylation:

– DNA methylation

A JSON file has been downloaded for each of these omics. It contains all the references to the archives necessary for downloading the data concerning the specific analysis.

These archives are downloaded in many folders, a script *ExtractionGene.py* has been developed to allows the extraction of only the necessary text file.

Each of these document files has the following structure:

GeneId	Value
ENSG00000000003.13	0.0089
ENSG00000000419.11	35.9391
ENSG00000000460.15	1.5156

Table 2.1. Structure of omic file

2.1.1 Creation of a single patient table for each omic of both leukemias

Once all the files have been extracted, a script was created which, for the omics taken into consideration:

- Filename is selected
- Search in the MANIFEST file of that omic for the corresponding CASEID
- Create a column within a table with column index the caseid and row index the gene. The intersection between the row and column index will contain the value of that gene for that patient.

This operation is done for all the files in the folder for each omic of that pathology to get to have a table described as follows:

The first line "ToL" identifies the type of leukemia and therefore has M values for the myeloid and L for the Lymphoid.

At the end of the execution of the scripts we had the following files for a total of 6 tables distributed as follows:

- Myeloid:

	CaseId1	CaseId2	CaseId3
ToL			
Gene1			
Gene2			

Table 2.2. Structure of generated omic file

  - CN
  - miRNA
  - mRNA
- Lymphoid:
  - CN
  - miRNA
  - mRNA

all of them with the following structure:

	CaseId1	CaseId2	CaseId3
ToL			
Gene1			
Gene2			

Table 2.3. Structure of generated omic file

2.1.2 Merge between the files of the same omics

In detail, these six tables (contained in 6 different files) have been merged to allow the model to be trained one file for each type of omics, regardless of the type of leukemia.
It was therefore necessary to merge the tables by omics to which they belong, using and generating 3 files, one for each omic containing the values of the patients of both pathologies.
The 6 starting files were structured as follows:

- Myeloid:

  - CN
  - miRNA
  - mRNA
- Lymphoid:
  - CN
  - miRNA
  - mRNA

The tables are then merged by type of omics, then

- Myeloid Copy Number with Lymphoid Copy Number
- Myeloid miRNA with Lymphoid miRNA
- Myeloid mRNA with Lymphoid mRNA

Therefore, having 3 new tables structured as follows:

	CaseId1M	CaseId2M	CaseId3M	CaseId1L	CaseId2L	CaseId3L
ToL	M	M	M	L	L	L
Gene1						
Gene2						

Table 2.4. Structure of final omic file

These files, one for each omic, containing the data of all patients for both pathologies, will be the inputs of our model.

To do this, a script has been created for each type of omic:

- *Copy Number: ALL_ReadCN_ML.py*
- *miRNA: ALL_ReadmiRNA_ML.py*
- *mRNA: ALL_ReadmRNA_ML.py*

In particular, these scripts perform different operations based on the formatting of the files that describe the omics in question.
In general, they retrieve the GeneId, its value, and the SubjectId from the omics file for that pathology and insert it into a new table.
The same operation is done for the other type of leukemia by writing in the table used previously.

2.1.3 GRCh38: Decoding geneid-genename

The 3 generated files contain information regarding all the sequencing of the omics under analysis. In detail, there are several "gene_biotype" that identifies the type of gene being analyzed. For this type of analysis, the type of gene taken into consideration is "proteing_coding", this information is provided by the sequencing of the reference human genome, HomoSapiens GRCh38.

This file, in * .gtf format, has been filtered with the following parameters:

- gene_biotype: "protein_coding"
- feature: "gene"

Furthermore, all columns not helpful in extracting the necessary information have been removed.
In the end, a file called hg38_onlygene_dropcolumns.csv was generated. This file is necessary because it contains the name of only the genes useful for the analysis.
As mentioned at the beginning of this section, the files containing the values of the omics contain genetic information that is not useful for this analysis, the newly generated file is necessary to filter only the genes used.
This operation is performed by a custom script named *ExtractGeneFromHG38.py*
Another necessary operation is the conversion of the geneid with its respective biological name.
Indeed, a script has been created to search for the geneid in the file containing the human genome (GRCh38). It finds the corresponding GeneName, and inserts it as the value of the last column of the omics being used.
It is done by the *ExtractGeneFromHG38.py* script.

2.2 Preprocessing

2.2.1 Evaluation of omics

This section illustrates the methodology for choosing the data to be used and the preprocessing activities applied to these.
In particular, we remind you that the available data are the following:

- Myeloid: CN, mRNA, and miRNA
- Lymphoid: CN, mRNA, and miRNA

In addition, the grouped data by type of omics are also available:

- Copy number: containing the values for ALL and AML
- mRNA: containing the values for ALL and AML
- miRNA: containing the values for ALL and AML

The first analysis was made on the omics data available for each patient. Indeed, an analysis was conducted to see how many patients have all three omics, just two, or just one.
This operation was done for both myeloid and lymphoid leukemias.
In particular, an analysis is made with Venn diagrams to identify the number of patients with major omic analyzes.
As can be seen from the Venn digraphs below, the number of the dataset containing the information regarding myeloid leukemia is greater than that of lymphoid leukemia.

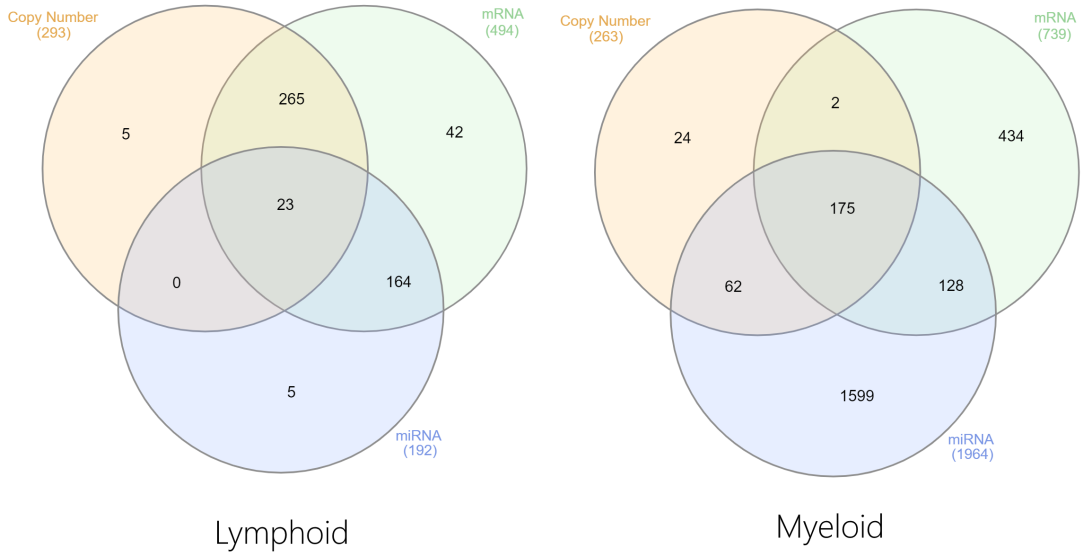


Figure 2.1. Data Venn diagrams

In particular, the analysis of all three omics would lead to a dataset composed of a too-small number of patients.
For this reason, it was chosen to use only two reference omics for this thesis,

miRNA, and mRNA, although the method allows the insertion and use of multiple omics as inputs.

The data relating to the analysis of copy numbers have been kept in the project directory.

2.2.2 Removal of outliers and null values

Another essential operation of the data preprocessing phase is removing outliers and feature values that retain NaN or null values.
To remove the records with null values, do all the rows containing values equal "NaN" or "?" Have been removed.
To do this, all the values of "null" and "?" a "NaN". Then, I was deleting all the lines that contained the "NaN" value. Below is the example of the command used for the procedure just available.

$$XOm[i] = XOm[i].replace(to\_replace =?, value = NaN) \tag{2.1}$$

It was done for both omics.
As regards the removal of the outliers, checks were made regarding the distribution of data for each omic.
The analysis technique used refers to using the Principal Component Analysis (PCA), applied to all the omics present in the input.
In particular, since the PCA is used in several points of this method, a function has been created which, given as input the dataset on which to apply the PCA, returns the data frame processed with two Principal Component This function is defined in the *function.py* file:

$$defPCAf(df) \tag{2.2}$$

Since this technique is applied to all the omics in input, a list of data frames has been used to contain all the PCAs of the corresponding omics.
Indeed, since the XOmScaled list contains all the omics with an application of the StandardScaler (detailed in the next paragraph), a for loop has been created that calls the PCAf function for each omic adds the returned value to the PCA list.

In order to perform these calculations, it is necessary to view the PCA data at a graphical level.
A function called "VisualizePCA" has been created to compute the PCA, which, given as input the data frame of the PCA under consideration, generates a distribution plot.

$$def visualizePca(df) \tag{2.3}$$

With this function on all the elements contained in the PCA list of the single omics, it was possible to generate the following graphs:

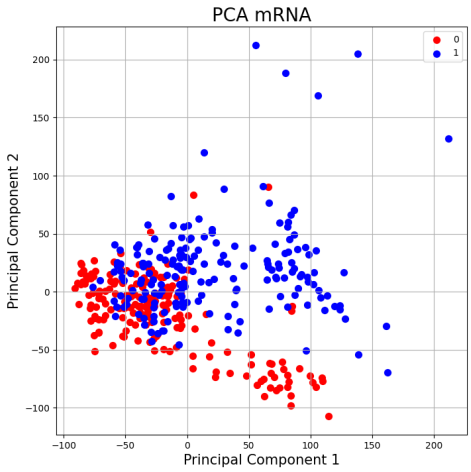


Figure 2.2. PCA on mRNA omic

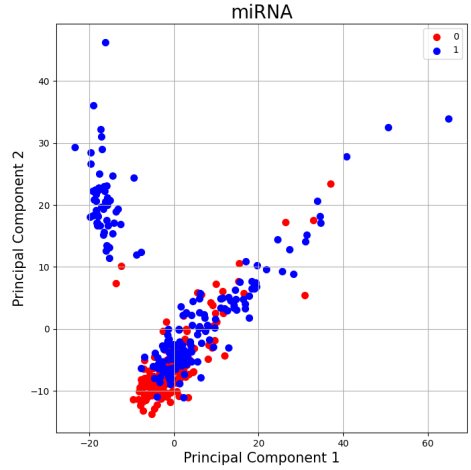


Figure 2.3. PCA on miRNA omic

Analyzing the distribution graph of the PCA data on the miRNA omics,

a cluster of data can be seen in the left part of the graph.
The first hypothesis was that the dataset contained data from differentiated analysis projects.
After a detailed analysis, it was discovered that the analysis data, at the same research project and containing values, were kept as elements of the analysis dataset.

2.2.3 Standardization

It is a step performed in the preprocessing pipelines of any machine learning or deep learning algorithm; it allows the elimination of the differences in scale between the various indicators, reporting the values of all indicators within a specific range (typically [0,1] or [-1,1]).
Doing so prevents gene values with a larger scale from dominating distance metrics compared to gene values with smaller values. At the same time, restricting the range of values to standard ranges around zero makes the training phase more stable with variations in the weights of the limited model, which translates into a more incredible speed of convergence of the network in the training phase.
In this work, the application to the dataset of both standardization and through the use of the scikit-learn library was analyzed.

The standardization assumes that the data is at a Gaussian distribution for which it expects that they will be rescaled to have mean = 0 and standard deviation = 1:

$$y = \frac{(x - \mu)}{\sigma} \tag{2.4}$$

This technique modifies the shape of the distributions leading them to assume a standard Gaussian distribution.

In order to make the algorithm dynamic, lists have been used. A list called XOmScaled is created, which, reading each element of the list containing the omics, standardizes them and adds them to the list. To summarize, make a copy of the omics by applying a standardization process This operation is contained in the *Tesi_v3.py* file

```
XOmscaledlist = list()
for i, scalerOm in enumerate(scalerOmlist):
XOmscaledlist.append(scalerOm.transform(XOmlist[i]))
print(i)
```

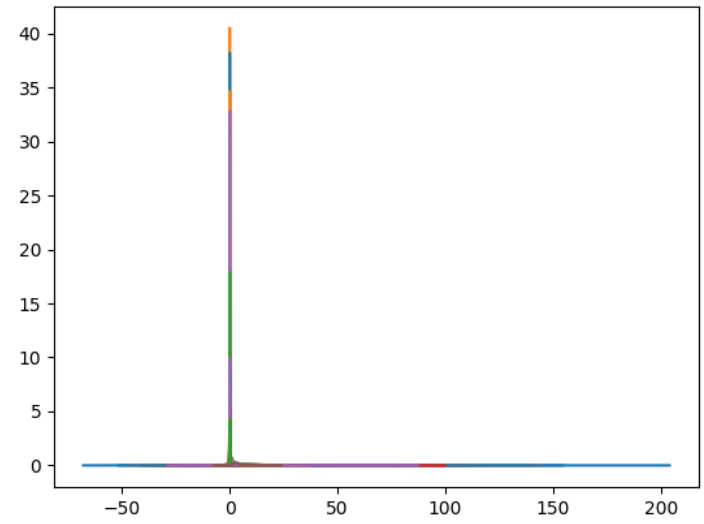


Figure 2.4. Distribution of mRNA data without Standardization

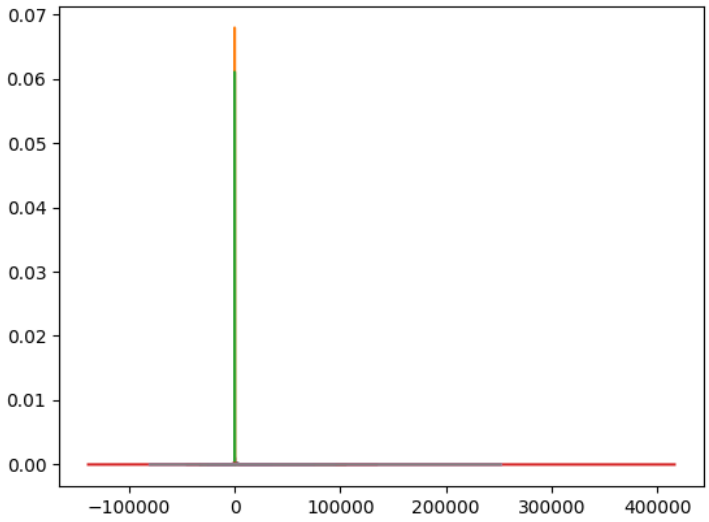


Figure 2.5. Distribution of miRNA data without Standardization

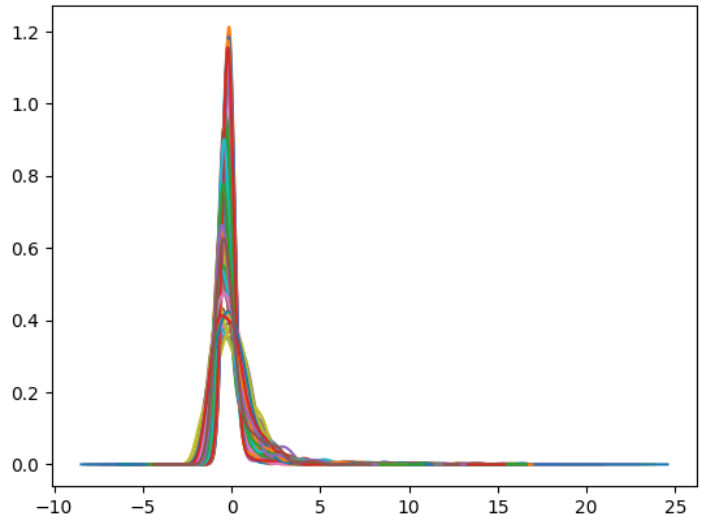


Figure 2.6. Distribution of mRNA data with Standardization

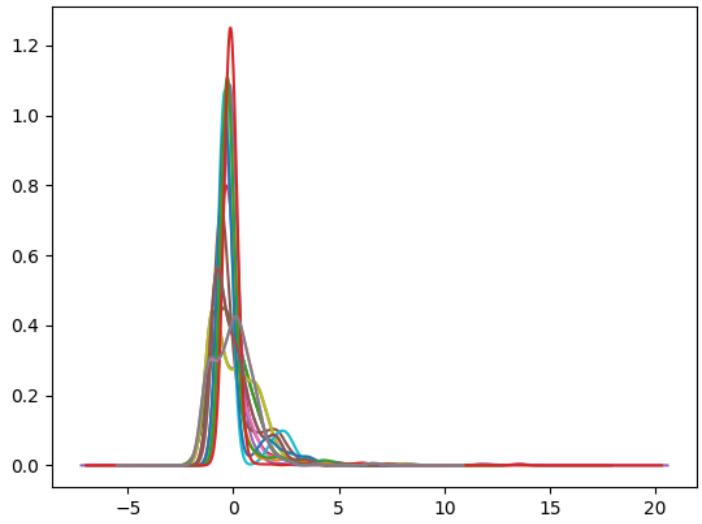


Figure 2.7. Distribution of miRNA data with Standardization

Chapter 3

Method

The purpose of this chapter is to discuss the design and implementation details of the proposed neural solution.

3.1 Assumption

This method aims to arrive at the definition of a model that inputs given two omics, manages to cluster the subject according to the disease from which they are suffering. The goal is to use an unsupervised model that does not need the input labels.
All files used were created as described in chapter 3. The project is structured as follows:

- config.py
- clustering.py
- function.py
- model.py
- thesis_v3.py
- data

The *config.py* file is where the model paths and parameters are described and configured. In particular, it is structured as follows:

- path1: contains the path of the first omic, it refers to the *.csv file contained in the "data" folder
- path2: contains the path of the second omic, it refers to the *.csv file contained in the "data" folder.
- pathIdTOL: contains the file path where the patient-type correspondence of leukemia is stored, therefore myeloid (M) or lymphoid (L).
- output_size: indicates the number of output nodes of the neural network. In this case, it is set to 30. Therefore, we will know that as output, we will have a matrix of n rows and output_size columns.
- type_of_loss_function: indicates which loss function to use. The possible choices are indicated in the "Loss Function" paragraph.

The *function.py* file contains the main data processing functions. Among the functions implemented in this file, we have the functions of importing omics (path1 and path2), the leukemia patient-type dictionary (pathIdTOL), and formatting these files, which are the first functions used when starting the method.
The other functions in this file will come later when we talk about them in the following paragraphs.

The *clustering.py* file contains the clustering functions that will be used in this method.

In particular, these functions receive as input the data frame on which to apply the clustering algorithm and the number of clustering. The returned value is a prediction vector of the membership clusters.

0	0	0	1	1	0	1

the element of index 2 belongs to cluster 0.
Here is an example of how functions are called:

def clustering_spectral(df, ncluster)

where is it:

- *df* is the dataframe on which to cluster
- *ncluster* is the number of clusters you want to use

The *model.py* file contains the definition of the model that will be trained and used for predictions. The creation of the model is invoked in the following way:

Subsequently, the content of this function will be deepened.

def model(C1, C2,lr, output_size, lf)

where is it:

- *C1* is the first omics, in this case, mRNA in the format rows-genes, columns-subjects
- *C2* it is the second omics, in this case, miRNA in the format rows-genes, columns-subjects
- *lr* acronym for *learning rate*, it indicates the step size to modify the weights (weights) of a deep neural network and is one of the most delicate and important hyperparameters to adjust to obtain excellent performance on our problem.
- *output_size* number of network output nodes (previously configured in the config.py file)
- *lf* acronym for function loss which will be explained in section 4.3

The *tesi_v3.py* file contains the body of the model, the parts in which the import, standardization, model, clustering, and evaluation functions are called.

The files used in this model are as follows

- ML_mRNA_CaseID_GeneName.csv
- ML_miRNA_CaseID.csv
- Dictionary_CaseId_TOL_B.csv

For all omics, in this case, mRNA and miRNA, the first row of both data frames has been removed as it contains the data on the type of leukemia, the same data present in the file indicated in pathIdTOL.

We remind you that the IdTOL file contains the correspondence between each patient and the type of leukemia from which he is affected.
The values contained in IdTOL were used in the evaluation function explicitly created to evaluate the model's effectiveness and subsequently described.
The mRNA and miRNA files are taken into account to consider the analyzes of all affected by those two types of leukemia.
As described in chapter 3, we have chosen only a few patients, more specified of which both mRNA and mRNA (see Venn diagram).
It was, therefore, necessary to filter these two omics to take only the interesting subjects.
The patients are contained in the IdTOL file. Therefore the Ids of the subjects of this file have been used as a filter on the omics. This dataset was called *sujectfilter*.

The functions contained in the *function.py* file were used to import the patient-type of leukemia dictionary, mRNA, and miRNA.

importCaseIdTOL(pathIdTOL)
reformatmRNA(pathOm, subjectfilter)

Omics paths and omics data frames are stored in lists. In the evolutionary phase, this technique allows us to insert more omics avoiding the modification of the code already written.

To get the final model that provides for the input of two omics without belonging labels, we started from simpler models, and with each new evolution, a characteristic of the final result was inserted.
The model is structured as follows:

- 1 part: neural network to perform a features reduction
- 2nd part: analysis and processing for clustering

The first part of the model, the neural network, attempts to reduce the dimensionality of the input data.

As we have seen, the input process is characterized by multiple files. In this case, the neural network input is the set of all the omics considered and analyzed, in this case, mRNA and miRNA.

This model, which we will call the "Final Model", has the following characteristics

- There is only one neural network model that receives both omics as input
- The omics in input can have a different dimensionality as regards the number of lines
- A Y label in input is not required
- A custom loss function has been defined that uses the mean squared error (mse) of both omics
- The output of the network is a matrix of output_size columns and number of rows equal to the number of rows of the largest omic
- The model performs a feature reduction of the subjects

In particular, first part is defined as follow:

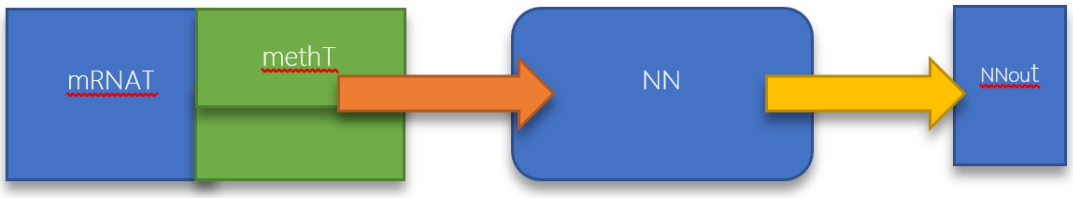


Figure 3.1. Final Model - First Part

The basic idea of this model is to perform a feature reduction of the subjects. Since the input files have a structure as shown here:

	Gene_1	...	Gene_n
Subject_1			
...			
Subject_n			

it was necessary to transpose this dataset. It is because the reduction occurs on the columns.

If we had left the initial file structure, we would have reduced genes and not in subjects as desired.

It is also important to underline that this operation no longer allows having the row-patient correspondence with the diagnosis, the theoretical Y label.

So, to allow the feature reduction of the subjects it was necessary to create the transpose of the input data and then have them in this format:

	Subject_1	...	Subject_n
Gene_1			
...			
Gene_n			

Another observation to make concerns the dimensionality of the two omics, mRNA and miRNA. Recall that the dimensionality of these two omics is distinctly different. One is about a tenth of the other. This feature was an observation that was fundamentally taken into consideration for this model.

Since the neural network with multiple inputs requires that the datasets have the same number of rows, the problem described above was a point of analysis in the development of the thesis.

After a detailed analysis of the possible solutions, it was decided to proceed as follows: given 2 omics, a function will find which of the two has the lower dimensionality, after which it will duplicate its lines until it reaches the number of lines of the larger omic.

The function that performs this operation is *duplOmics* and is defined in *function.py* as follow:

duplOmics(XOm, leng)

This function receives two parameters:

- *XOm* the omic on which to expand the dimensionality of the lines.
- *leng* the number of lines to be reached.

It is possible to check before calling it to recognize the omic to duplicate the rows, thus defining a generic function.
If the number of omics is greater than 2, it is necessary to identify the largest omic and call the duplOmcis function on all the others.

In the end, we will therefore have all the data frames with the same number of rows and the same number of columns.

In this way, the two omics are supplied as input to the neural network with the same number of genes(rows) and the same number of subjects (columns).

The neural network is structured to reduce the number of subjects to a predefined and configurable number within the config.py configuration file in the "output_size" field.

Modeling is invoked using the function:

model(C1, C2,lr, output_size, lf)

which is defined in the model.py file.
This model is textured with 4 dense layers. More precisely, the first two are independent for each omic, the third is the concatenation of the two, and the fourth is the output of our network, the output_size.

In this thesis, the value of output_size is 30.

The neural network is structured as follows:

- Dense Layer of 100 nodes
- Dense layer of output_size nodes
- Concatenation with layer generated at the same point on the other omic
- Dense layer of output_size nodes

For each dense layer, the ReLu function was used as the activation function. The trend of this activation function is shown in the figure

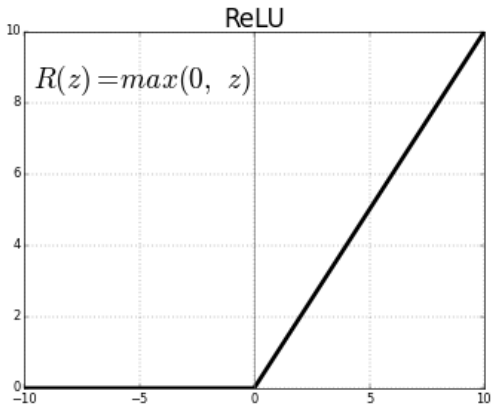


Figure 3.2. ReLu Function Graph

Below is a drawing of the structure of the model just described:

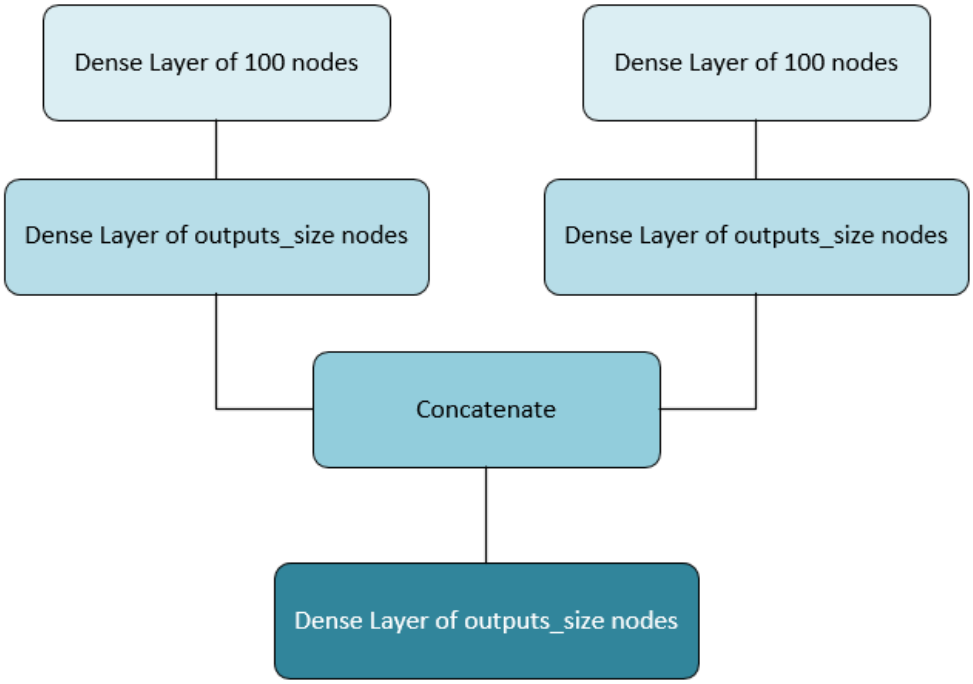


Figure 3.3. Internal Model

3.2 Artificial Label

This method aims to make the model independent from a mandatory Y label provided in the input.

In the literature, it has been seen that in the medical and financial fields (see ref3), arithmetic metrics are used on the Xs to define an artificial Y that is supplied to the model.

It is therefore mandatory to use a Y label, but, with this technique, it is calculated from the values of the Xs provided in the input.

In particular, the arithmetic mean of the values of each gene of all patients is used in this method.

In the function.py file is defined the function that the operation just described is performed. The function is:

createyOmics(OmList)

the list of data frames containing the values of all the omics is passed as parameters.

Then the arithmetic average of all the values of each row of both omics is made.

So for each row:

$$MOmic_j = \frac{1}{n}\sum_{i=1}^{n} a_i = \frac{a_1 + a_2 + \cdots + a_n}{n} \tag{3.1}$$

At the end of the execution of the function *createyOmics(OmList)* there is a vector containing for each row the average of the values of the duplicate rows of all the omics.

3.3 Loss Function Custom

Measuring model performance is the crux of any machine learning algorithm, and this is done through the use of loss functions or cost functions.
Choosing these functions can help the model learn better; on the contrary, choosing the wrong one could lead the model not to learn anything of significance.
This function is called the loss function.

To assess the loss function with improved performance has been the model runs completed considering using the appropriate evaluation function d and the match (see "Rating function of the match"), of the 'performance algorithm with loss function choice. The results of the 4 tests were reported below :

	$mean\left(\frac{mse_1 + mse_2}{mse_1 * mse_2}\right)$	$mean\left(\frac{mse_1 + mse_2}{mse_1 * mse_2} * mse_3\right)$	$\frac{mse_1 + mse_2}{mse_1 * mse_2}$	$\frac{mse_1 + mse_2}{mse_1 * mse_2} * mse_3$
K-Means	-Match with 0: 0 -Match with 1: 923 For class 1 -Match with 1: 77 -Match with 0: 1000	For class 0 -Match with 0: 534 -Match with 1: 380 For class 1 -Match with 1: 620 -Match with 0: 466	For class 0 -Match with 0: 943 -Match with 1: 0 For class 1 -Match with 1: 1000 -Match with 0: 57	For class 0 -Match with 0: 0 -Match with 1: 877 For class 1 -Match with 1: 123 -Match with 0: 1000
Spectral Clustering	For class 0 -Match with 0: 560 -Match with 1: 1000 For class 1 -Match with 1: 0 -Match with 0: 440	For class 0 -Match with 0: 495 -Match with 1: 807 For class 1 -Match with 1: 193 -Match with 0: 505	For class 0 -Match with 0: 0 -Match with 1: 1000 For class 1 -Match with 1: 0 -Match with 0: 1000	For class 0 -Match with 0: 998 -Match with 1: 24 For class 1 -Match with 1: 976 -Match with 0: 2
Gaussian Mixture	For class 0 -Match with 0: 552 -Match with 1: 997 For class 1 -Match with 1: 3 -Match with 0: 448	For class 0 -Match with 0: 466 -Match with 1: 620 For class 1 -Match with 1: 380 -Match with 0: 534	For class 0 -Match with 0: 943 -Match with 1: 0 For class 1 -Match with 1: 1000 -Match with 0: 57	For class 0 -Match with 0: 991 -Match with 1: 1 For class 1 -Match with 1: 999 -Match with 0: 9

As seen from the table, the two most performing and most reliable loss functions are reported in the last two columns.

In particular, the formula reported in the third column was chosen, the harmonic average between the mse of omics.

The algorithm has been designed to be executed with the preferred loss function among those defined in the functions. It is, therefore, possible to choose the type of loss function by entering the number corresponding to the chosen function in the *config.py*

- Mse1
- mse2

In this case, the following value is reported in the config.py: loss_function=3

3.4 Output of Neural Network, Centroid and Squared Matrix

The first part of the model deals with the feature reduction of patients. The output of the neural network is a matrix equal to as many rows as the patients and as many columns as the output_size indicated in the *config.py*.

This matrix results from a feature reduction on the subjects of both omics that we generically call y_pred.

The columns of the y_pred matrix output to the neural network are the centroids we used to analyze the results in the second part of the model. The rows of this matrix are the genes.

The second part of the model aims to produce a similarity matrix from the patients.

The centroids were used to calculate the subjects closest to each of them and assign them.

Subjects belonging to the same centroid are considered similar. In particular, the Euclidean distance from each centroid was calculated for each subject.

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \tag{3.2}$$

Having then calculated for the subject-i, output_size distances, we find the smaller distance value. The centroid corresponding to the shortest distance value is considered the centroid to which that subject belongs. Once the centroid of all the subjects has been calculated, we produced a matrix in which the number of the centroid to which it belongs. This last value is between 1 and output_size .

Subject	Centroid

Table 3.1. Subject - Centroid Table.

Since the goal is to generate a similarity matrix and therefore defined as follows:

	Subject1	Subject2	Subject3	Subjectn
Subject1	1	0	1	0
Subject2	0	1	1	0
Subject3	1	0	0	1
Subjectn	0	0	1	1

Table 3.2. Example of Similarity Matrix.

A function has been implemented which, given the subject-centroid matrix as input , generates a similarity matrix

squarematrix(subjcentroid)

In this matrix, given the subject-i and the subject-j, if both belong to the same centroid, they will have value 1, otherwise 0.

Being a similarity matrix, it has values of 1 on the diagonal and is symmetrical.

This similarity matrix is referred to as the "Squared matrix". To reach this model, which we will call "Final Model", we went from 3 simpler models to modify a part with a characteristic of those listed at the beginning of this chapter.

The 3 models that led us to the "Final Model" are summarized below.

3.4.1 Multiple Inputs Model

The first starting model aims to create a neural network that gives two omics as input (in this case mRNA and miRNA) and the relative label for each patient, creating a neural network based on training on these data.

It is therefore an MLP standard with multiple inputs as can be seen in the figure below.
In particular, this model requires that the inputs are not transposed as in the final model but are in the row-patient and column-gene format as shown in the figure:

	Gene1	Gene2	Gene3	Genen
Subject1				
Subject2				
Subject3				
Subjectn				

Table 3.3. Input file for Multiple Inputs Model Model

This model involves a reduction of the features representing the genes.
This approach, combined with clustering techniques, would lead to a clustering of genes to differentiate the type of leukemia.
Therefore, it would group the genes that most characterize myeloid leukemia on the one hand and the genes that characterize lymphoid leukemia on the other.

This technique has already been used and implemented to identify the most expressive genes of the two pathologies.
Our goal is to verify the similarity between patients who have a similar genetic expression.

The neural network used in this model is structured as follows, for each omic:

- Dense Layer of 100 nodes
- Dense layer of 80 nodes
- Dense layer of output_size nodes
- Concatenation with layer generated at the same point on the other omic
- Dense layer of output_size nodes

This network structure was maintained even for the model Final. Furthermore, as a first model, standard configurations were used for the compilation of the model:

model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])

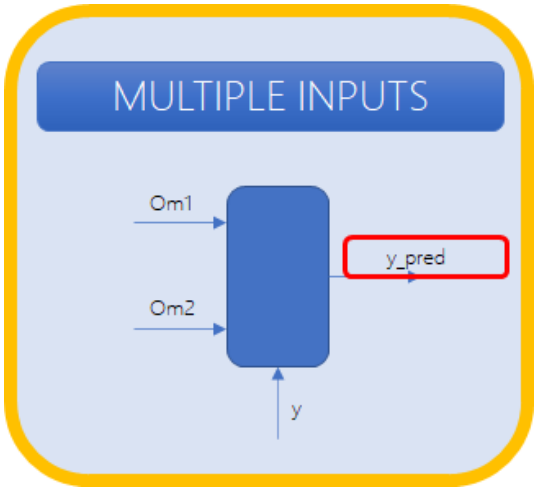


Figure 3.4. Multiple Inputs Model Model

This model has some potential regarding the input of multiple omics (multi-omics approach), while it has some limitations regarding:

- clustering on genes
- use of a Y data label
- Standard loss function

The evolutions of the models that we are going to see are exactly focused on overcoming these limits.

3.4.2 Indipendent Inputs model

This model has two objectives.

- The *first* is to prepare data for clustering based on patients and not on genes as seen in the first model.
- The *second* objective is to make the neural network independent from a Y provided in input and then proceed with the generation of an artificial Y label.

To solve the limit described in the first point, that is to prepare the model to obtain a reduction of patients, serve as input the table transposed with the correspondence row-gene column-subject.
Having a matrix composed as follows:

	Subject1	Subject2	Subject3	Subjectn
Gene1				
Gene2				
Gene3				
Genen				

Table 3.4. Input file for Multiple Inputs Model

For the second objective of this model, that of making it independent of a Y label data, we resorted to research carried out in the machine learning literature.

After various evaluations and research, the arithmetic mean of all patients' values for each gene of all patients was used.

This choice was kept for the final model.

Furthermore, this model aims to perform a feature reduction of patients, therefore, have as input the table transposed with the correspondence row-gene column-subject.

Having such a composed matrix, it is, therefore, easier to calculate the Y as first defined. It is therefore

	Subject1	Subject2	Subject3	Y_Artificial_Label
Gene1				
Gene2				
Gene3				
Genen				

Table 3.5. Data Structure for Indipendent Inputs Model

Since the input data transposed, there is a different dimensionality regarding the number of genes (and therefore of the lines) of the two omics; two different neural networks were used, one for each omic.

Each of these neural networks retains the structure defined in the first model and is trained based on the number of genes considered for the omics given in input.

The output of each of these networks is a matrix formed as follows:

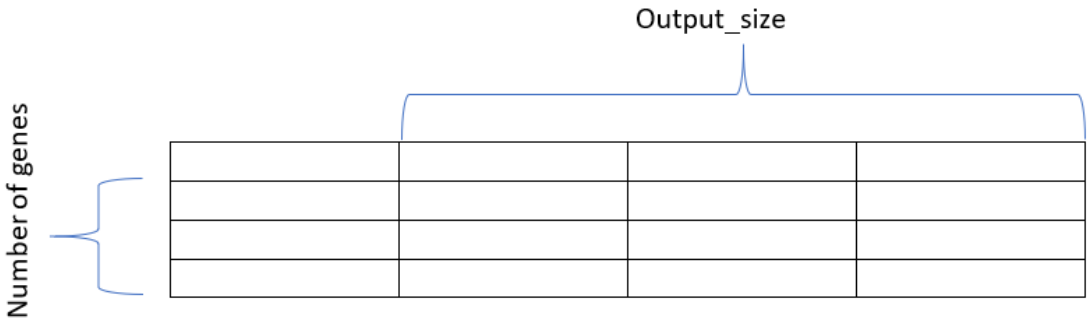


Figure 3.5. Output of Indipendent Inputs Model

To have a general overview of this model, the table just seen is corresponding to the y_pred1 and y_pred 2 of the drawing shown here.

Having two y_pred, each relating to an omic, it is as if we had generated for each omic a matrix of output_size centroids.

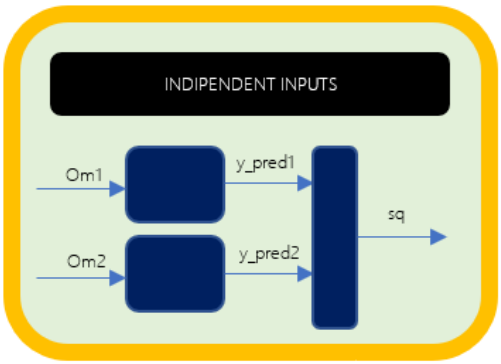


Figure 3.6. Indipendent Inputs Model

This function is defined in the function.py file and requires as input the original patient data frames and the centroid matrix generated by the neural network.

centroidsubject1(XOm1_scalerT, y_modScaled)

In particular, this function considers omic and centroid matrix individually and looks for the minimum distance of each subject of that omic from all the columns of the corresponding y_pred and therefore from the centroids.

Once the column with the shortest distance is found, that subject is assigned to this column to say to the centroid.

At the end of the processing, we obtained two tables, one for each omic where the centroid of assignment is indicated for each subject.

	Centroid
Subject1	
Subjectn	

Table 3.6. Subject-Centroid Table

From this subject-centroid matrix, the similarity matrix was generated. This matrix has the characteristic of identifying the subjects that have the same centroid.

It is a square matrix with a row and column index equal to the subject if two subjects are at the same centroid, crossing the two indices with the value 1.

This operation was performed for both omics, thus generating two similarity matrixes.
At the end of the process, the two similarity matrices were added together to obtain a single similarity matrix.

This methodology was kept for the final model but still has a significant limitation: two neural networks are trained independently.

Furthermore, the y_pred are independent and do not take into account the other omics under analysis.

3.4.3 Multiple Inputs Concatenate model

The third model aims to solve the limit imposed by the second model and therefore have a single neural network that receives both omics as input and generates a y_pred, which results from the processing of both inputs.
DISEGNO TERZO MODELLO

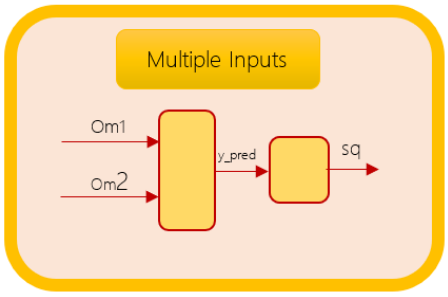


Figure 3.7. Multiple Imputs Concatenate Model

As already described, this model inherits some characteristics from the two previous ones:

- The omics are provided in input with the format row-gene, column-subject
- A Y label is not necessary. It is calculated as the average of the values of that kind for all patients
- The neural network model is composed as follows:
  - Dense Layer of 100 nodes
  - Dense layer of 80 nodes

  - Dense layer of output_size nodes
  - Concatenation with the layer generated in the same point by the other omic
  - Dense layer of output_size nodes

This model, in particular, solves the problem related to the generation of a y_pred, which takes into account both omics. One of the major problems of this evolution was the different dimensionality of the omics.
Four possibilities were analyzed to overcome this problem:

- Truncate the number of lines of the higher omic to the number of lines of the minor omic
- Generate random values to be included in the minor omic so that the number of rows equal to that of the major omic is reached
- Insert null values in the minor omic so that the number of rows equal to that of the major omic is reached
- Duplicate the values in the minor omic so that the number of rows equal to that of the major omic is reached

Initially, the first technique was used. That is, the greater omic was truncated to the number of lines of the minor omic.

Since this solution eliminated patient characteristics and thus affected the generation of centroids, the fourth option was used.

The second and third options were excluded because they added gene values to each subject that did not reflect the patient's identity.

This methodology was the last step in creating the final model.

Chapter 4

Results

The model's output just described is a similarity matrix which therefore presents the properties of symmetry and quadraticity.

Since the goal is to verify the actual gain, evaluations were made before and after the neural network. Clustering techniques were used to group the subjects belonging to the same class and therefore to the same type of leukemia to evaluate the results.

These clustering techniques, which we will see later, can group patients who are similar to each other in the same cluster. In particular, we need the similarity matrix to map patients to each other.

It is essential to evaluate the reliability of this model to compare the data between a clustering carried out with this model and without.
These evaluations are necessary to understand how effectively it is helpful to apply this analysis method to gain reliability in the clustering phase on the two types of leukemia. The two positions of performance analysis are before this model and after, it shown in red in the drawing

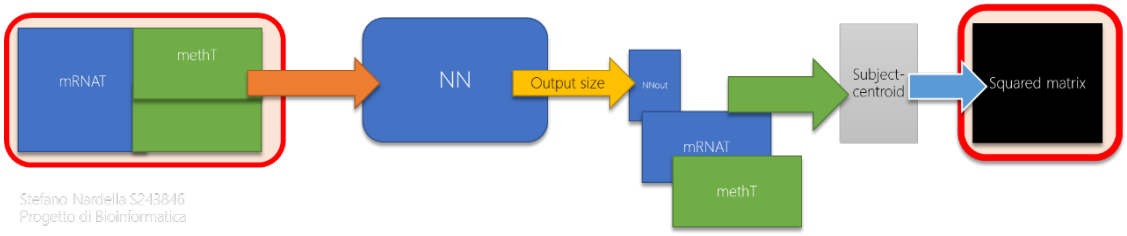


Figure 4.1. Point of analysis on model

In particular

- before the neural network, a direct clustering was carried out on the datasets of the two omics and then on their corresponding PCAs.
- After the neural network, and therefore on the square matrix, clustering techniques were applied first on the similarity matrix and then on the latter's PCA

In summary, we have 4 types of analysis, two for each point of model:

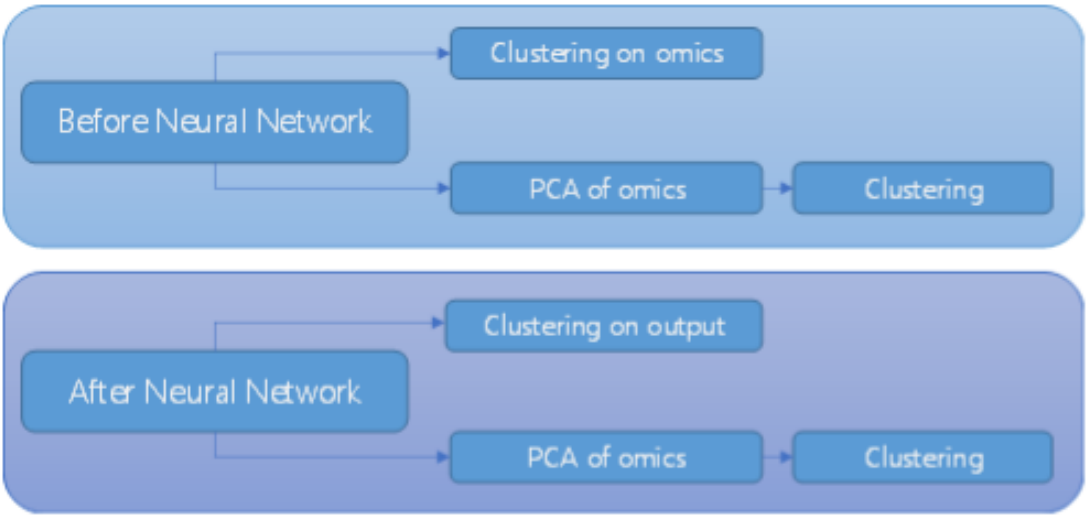


Figure 4.2. Summary of analysis

The clustering techniques that have been used are the following:

- Kmeans
- Spectral
- Gaussian
- Hierachical

These functions are implemented in the clustering.py file, and all maintain the same necessary parameter structure.

def clustering_kmeans (df, ncluster)
def clustering_spectral (df, ncluster)
def clustering_gaussian (df, ncluster)
def clustering_hierarchical (df, ncluster)

It is necessary to pass as a parameter the data frames to be clustered and the number of clustering to be found to invoke these clustering methods.
This structure is designed to make this model generalizable. In fact, in the chapter "Conclusions," these choices are motivated for the possible evolutions.

In this case, the number of clusters equal to 2 was used in all techniques, corresponding to our two starting classes: lymphoid and myeloid leukemia.

In some analysis points, it was not possible to apply all the techniques listed. For example, spectral clustering requires that the input be a square matrix.

In fact, in the first point of analysis, we do not yet have a similarity matrix available before the neural network, so it was not possible to apply this clustering technique.
Whenever the PCA of a data frame was used, it was impossible to apply spectral clustering as the main components are 2.

For each clustering technique, given a matrix as input, the prediction vector is returned, where the cluster to which it belongs is indicated for the i-th element.

4.0.1 Function of Evaluation

An evaluation function has been implemented to count how many are the actual matches and how many are not.

Since clustering techniques do not recognize the actual values that we have assigned to a cluster but are limited to clustering, we must understand if the prediction has reversed the class-value matching.

The evaluation function is therefore composed of two sub-functions:

- The first evaluates whether the clustering technique has inverted the value of the class.
  To do this, check the highest match value between the belonging class-belonging class and the opposite class-belonging class.
  In this case, for example, the matches 0-0 and 0-1 or 1-1 and 1-0 are controlled.
  If the number of matches is greater than with the opposite class, an inversion of class-value representation has occurred. Therefore, the correct values are adjusted.

- The second compares each predicted value with the actual value increases the "Match" value if correct and the "No Match" value if wrong.

This function, called *countzeroone*, receives clustering predictions and real values as input.

countzeroone(y_pred, y_real)

This function has no return value. It just prints the match and no match values

4.0.2 Clustering before Neural Network

The first analysis was carried out before using the model and, therefore, before the neural network.
In particular, two types of analyzes were carried out:

- On the data relating to the omics
- On the PCA of data relating to the omics.

The analysis consists of applying clustering techniques directly on the raw omics data, then on mRNA and miRNA.
This analysis is also performed on the PCAs of both omics.

It was impossible to use spectral clustering and gaussian since the two techniques require a square matrix as input to analyze omics data. The other applicable clustering techniques were then used.
For the analysis on the first part, we start from two rows-genes columns-patients tables, one for each omic.

	Subject1	Subject2	Subjectn
Gene1			
Gene2			
Gene3			

Table 4.1. Example of Data Input Omics

This table is the typical starting point for both assessments made in the first part of the analysis.

Below I summarized what has been described so far, from the input to the results table expected at the end.

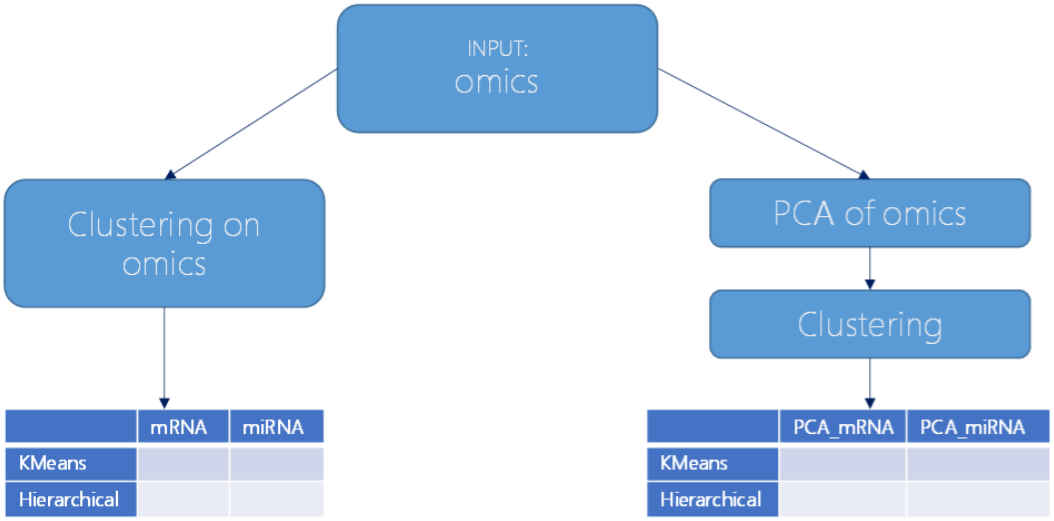


Figure 4.3. Analysis scheme before the neural network - First point

For the first analysis, therefore applying the clustering techniques directly on the omics, it was possible to apply only two methods: KMeans and Hierarchical. So the results obtained by applying these two techniques only the following:

	mRNA	miRNA
KMeans	52,9%	51%
Hierarchical	51,3%	51%

Table 4.2. Result of Clustering on Omics

The second analysis always starts from the same input, two tables in the format: TABELLA STRUTTURA FILE OMICA

	Subject1	Subjectn
Gene1		
Gene2		

Table 4.3. Structure of File for PCA Clustering before Neural Network

One for each omic. A PCA with many components equal to two was made on each of these, and then the results were plotted graphically.
A function has been defined in the *function.py* file, which, given a data frame as input, generates the PCA with two components.

def PCAf(df)

Before the plot, the data relating to the two omics have the following format:
TABELLA STRUTTURA PCA BEFORE

	PCA_Comp_1	PCA_Comp_2
KMeans		
Hierarchical		

Table 4.4. Structure of results of Clustring on PCA Before Neural Network

To plot the PCA graph, a function has been defined in the *function.py* file which, given as input a dataframe in 2 PCA components format, displays a video of the graph.

def visualizePCA(df)

After plotting the graphs of the two PCAs applied directly on the omics, we can see how the two clusters are not shown graphically separable.
Therefore, this technique does not provide an added value in phase clustering of this data. GRAFICO PCA

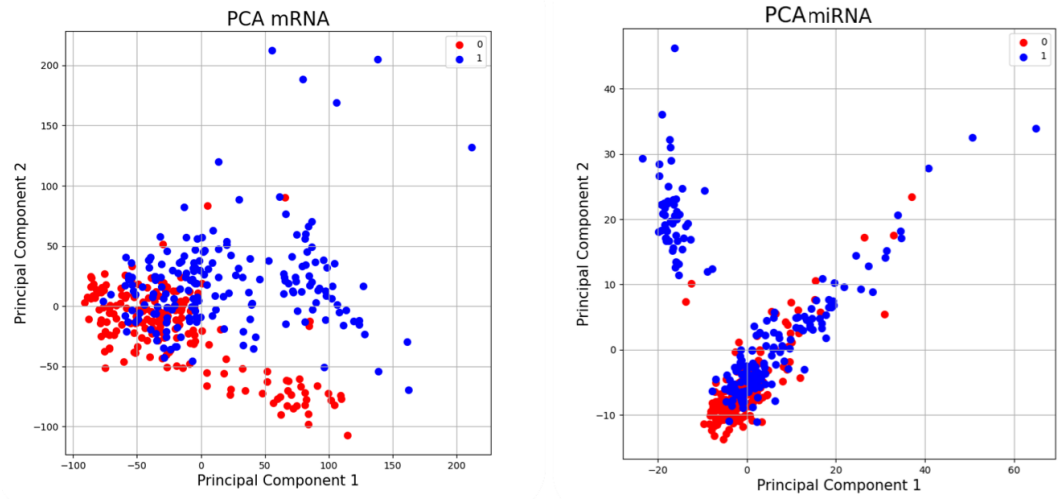


Figure 4.4. PCA Before Neural Network

After having displayed the distribution of the two omics, the clustering techniques were applied, more precisely the same ones used directly on the omics: KMeans and Hierarchical

	PCA_Comp_1	PCA_Comp_2
KMeans	52,9%	51%
Hierarchical	51,3%	51%

Table 4.5. Result of Clustering on PCA Before Neural Network

TABELLA RISULTATI PCA CLUSTERING BEFORE NN

As we can see, the application of the PCA before applying the clustering techniques does not provide a performance advantage of the model.

4.0.3 Clustering After Neural Network

The second macro-part of the analysis is focused downstream of the neural network and, therefore, on the square matrix of similarity in output to the model.

Also, for this analysis, two techniques were used:

- The first is a direct clustering on the "Squared matrix" or the similarity matrix

- The second is clustering on the PCA applied to the squared matrix.

For both analyzes, we start from the similarity matrix thus defined: mMATRICE DI SIMILARITA'

	Subject1	Subject2	Subject3	Subjectn
Subject1	1	0	1	0
Subject2	0	1	1	0
Subject3	1	0	0	1
Subjectn	0	0	1	1

Table 4.6. Similarity Matrix.

Below we have summarized what has been described so far, from the input of the clustering techniques to the results table expected at the end. SCHEMA RIASSUNTIVO DELLE DUE TECNICHE (5.5)

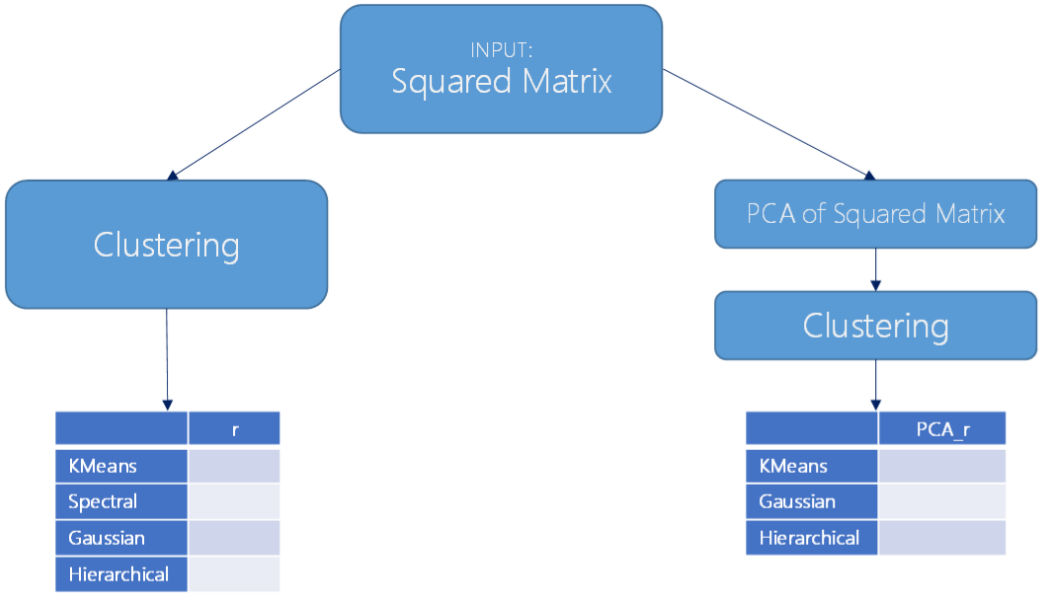


Figure 4.5. Analysis scheme after the neural network - Second point

The first analysis, as described, was done by directly applying the clustering techniques on the square matrix output to the model.
The first analysis, as described, was carried out by applying clustering techniques directly to the model on the output of the square matrix.

In this case, being a square matrix, it was possible to apply all 4 clustering techniques listed above. Below are the results of the first analysis:
TABELLA RISULTATI CLUSTERING SU R DOPO NN

	Squared Matrix
KMeans	66,9%
Spectral	65%
Gaussian	64,4%
Hierarchical	65%

Table 4.7. Result Clustering on Squared Matrix After Neural Network

Also, for the second analysis, we start from the similarity matrix. The first step for this analysis was to calculate the PCA of the square matrix, setting 2 as the number of components. After generating the PCA components of the square matrix, the values were plotted. The generic function was used to generate the PCA of the square matrix:

def PCAf(df)

this function receives as input the data frame of which you want to create the PCA with 2 components and returns the data frame with the generated components.

Once the PCA of the matrix was generated, the graph was displayed to identify the two classes. A function has been defined in the function.py file to display the PCA graph, which generates the graph given the input data frame to the plot.

def visualizePCA(df)

GRAFICO PC SQUARED MATRIX

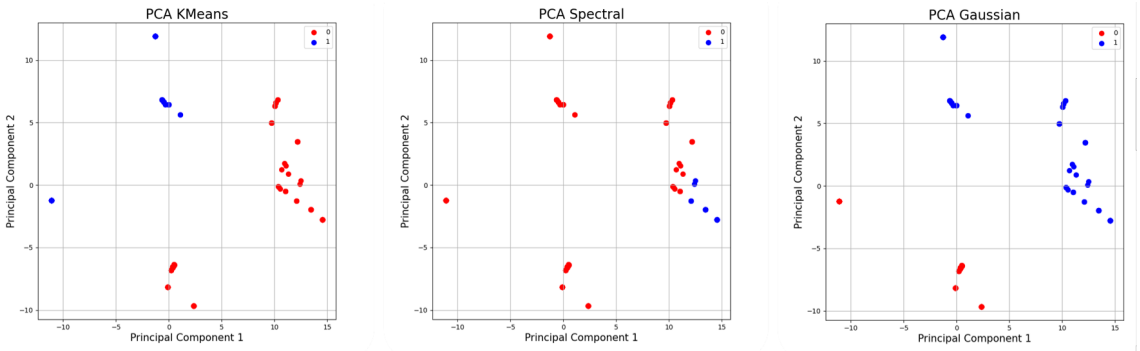


Figure 4.6. Plot of PCA on Squared Matrix

After applying the possible clustering techniques (in this case, it was not possible to apply Spectral Clustering), the evaluation function used for the other three analyzes was also recalled, reporting the following results:

TABELLA RISULTATI PCA AFTER

	Squared Matrix
KMeans	69%
Gaussian	69%
Hierarchical	69%

Table 4.8. Result Clustering on PCA of Squared Matrix After Neural Network

As you can see, we have a further increase in the match rate using this technique.

Chapter 5

Discussion

In this final chapter, we go to interpret the data extracted and collected in the previous chapter.
In particular, we will review the results from the analyzes carried out before and after the neural network. Point of attention for the points indicated in the figure. For each of the points highlighted, two types of analysis were

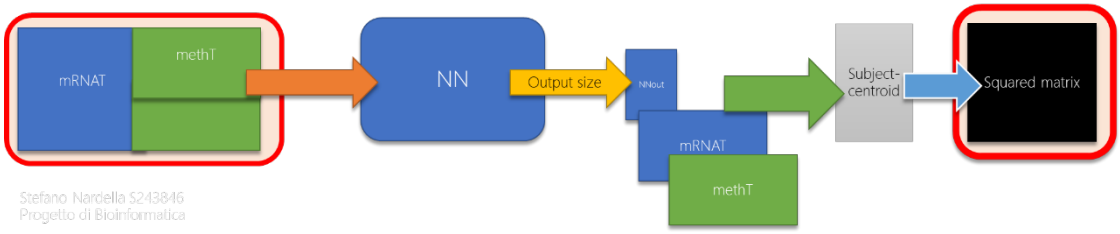


Figure 5.1. Point of analysis

carried out:

- PCA and clustering
- Clustering

Recall that the main objective was to understand if, applying a model composed as follows:

- Neural network for the identification of centroids
- Similarity matrix
- Clustering

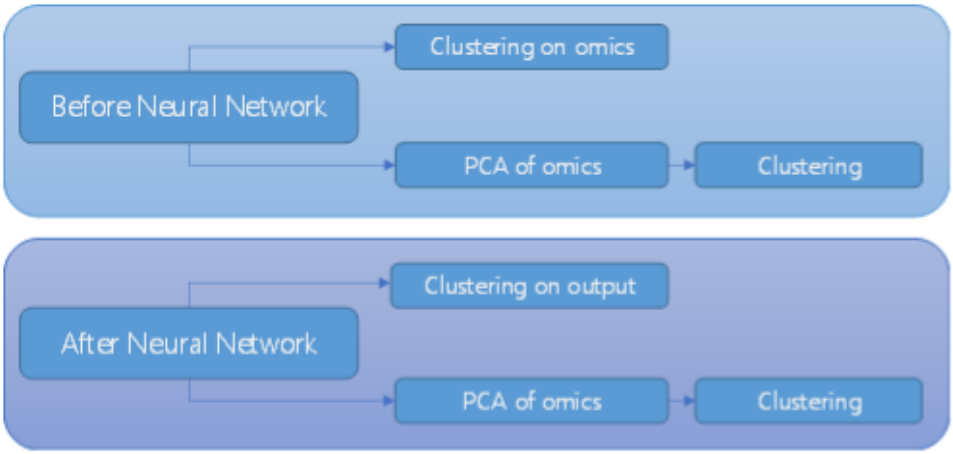


Figure 5.2. Summary of analysis

We can increase the clustering reliability of the model to direct omics clustering. Having implemented a custom function for the reliability calculation that is composed of two sub-functions:

- The first evaluates whether the clustering technique has inverted the value of the class.
  To do this, check the highest match value between the belonging class-belonging class and the opposite class-belonging class.
  In this case, for example, the matches 0-0 and 0-1 or 1-1 and 1-0 are controlled.
  If the number of matches is greater than with the opposite class, an inversion of class-value representation has occurred and therefore the correct values are adjusted.

- The second compares each predicted value with the real value, increases the "Match" value if correct and the "No Match" value if wrong.

This function, called *countzeroone*, receives clustering predictions and real values as input.

countzeroone(y_pred, y_real)

This function has no return value. It just prints the match and no match values.

The following results were obtained:

- Before Neural Network

	mRNA	miRNA
KMeans	52,9%	51%
Hierarchical	51,3%	51%

Table 5.1. Result of Clustering on Omics Before Neural Network

	PCA_Comp_1	PCA_Comp_2
KMeans	52,9%	51%
Hierarchical	51,3%	51%

Table 5.2. Result of Clustering on PCA Before Neural Network

- After Neural Network

	Squared Matrix
KMeans	66,9%
Spectral	65%
Gaussian	64,4%
Hierarchical	65%

Table 5.3. Result of Clustering on Squared Matrix

	Squared Matrix
KMeans	69%
Gaussian	69%
Hierarchical	69%

Table 5.4. Result Clustering on PCA of Squared Matrix

Comparing the results obtained from direct clustering on the omics and that on the PCA of the square matrix, a gain is noted. This result is not taken for granted, especially in the multi-omics field.
The previous results used as many neural networks as omics, process each omic independently and process the data by merging the individual results. The model presented is a solution that considers all the omics present, processing them all through a single multi-input neuronal network. Furthermore, the clustering carried out on the similarity matrix generated by the model is cleaner and avoids the integration step of the other omics downstream of the model.
Therefore, it can be said that this is a significant result for the multi-omics field and as a function of the precision gain obtained.
This pleasure can be extended to other contexts of analysis. Particular importance is given to the multi-omic approach, which, as described in this thesis, can benefit from a model of this kind.

Chapter 6

Conclusion

Leukemias are a group of blood cancers comprised of rapid and uncontrolled-growth of immature and atypical cells that infiltrate the bone marrow, whereall blood cells are produced. Leukemias make up 33-35% of childhood cancers; in children, about 80% ofcases are acute lymphoblastic leukemia. In particular, in recent years, innovative therapies have allowed an increasein the survival rate, leading to great progress in healing, going from minus 10% - 20% in the 90s to 80% in the current years. These advances have been possible thanks to research, in this area the contribution of the bioinformatics discipline has been invaluable.
The goal of this thesis in the bioinformatics field was to analyze a multi-omic approach to cluster patients so that similar ones are assigned to the same cluster, simultaneously considering all data sources, in this case omics data more precisely two types of transcriptomics data, miRNA and mRNA expression.
In detail, the proposed method has been evaluated on patients affected by myeloid and lymphoid leukemias (AML, ALL).
In this thesis, the proposed technique computes distances using pseudo-samples (also called centroids) generated by the neural network to identify the two classes, AML, and ALL diseases. Indeed, the network's output is a matrix of centroids generated from the data distribution of the input omics. The method is based on a Multi-Layer Perceptor (MLP) architecture which takes as independent inputs the miRNA and mRNA expression matrices. In this sense, the method can be defined as a multi-input approach. In particular, the structure of the model is as follows:

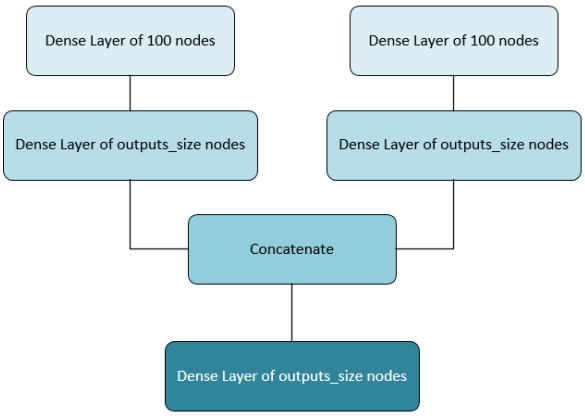


Figure 6.1. Internal Model

In addition, it is not necessary to have a Y label given as input in the training phase. Indeed, the proposed method computes an 'artificial' Y label from the expression values of mRNA and miRNA input matrices. For each patient, the artificial label is computed as the average expression values of all its features.
The output of the neural network is a matrix that for each omics outputs the centroids. Since this matrix is in the same dimensional space as the input features, computing distances between patients and centroids, I assigned all the samples to the closest centroid.

Subject	Centroid
2	4
4	5

Table 6.1. Example Subject - Centroid Table.

In the end, a similarity matrix is computed. Then, I applied KMeans, Spectral, Gaussian Mixture and Hierachical clustering techniques both on the similarity matrix and the original data, with and without a PCA dimensionality reduction. An evaluation function has been implemented that allows you to count how many are the actual matches and how many are not. In detail, it verifies if the clustering technique has correctly matched the cluster label, counts the correct matches, and returns a compatibility percentage. The results confirm that using this model brings a gain:

	mRNA	miRNA
KMeans	52,9%	51%
Hierarchical	51,3%	51%

Table 6.2. Result of Clustering on Omics Before Neural Network

	PCA_Comp_1	PCA_Comp_2
KMeans	52,9%	51%
Hierarchical	51,3%	51%

Table 6.3. Result of Clustering on PCA Before Neural Network

	Squared Matrix
KMeans	66,9%
Spectral	65%
Gaussian	64,4%
Hierarchical	65%

Table 6.4. Result of Clustering on Squared Matrix After Neural Network

	Squared Matrix
KMeans	69%
Gaussian	69%
Hierarchical	69%

Table 6.5. Result Clustering on PCA of Squared Matrix After Neural Network

in particular, this model, structured as follows:

- Neural network for the identification of centroids
- Similarity matrix
- PCA on Similarity matrix
- Clustering

increases about 20% in the clustering applied on the input omics and those with the PCA on the similarity matrix.

Bibliography

[1] Speciale Leucemie,
https://www.fondazioneveronesi.it/magazine/speciali/speciale-leucemie,
Fondazione Umberto Veronesi

[2] Tumori pediatrici - Leucemie,
https://www.airc.it/cancro/informazioni-tumori/guida-ai-tumori-pediatrici/leucemia-linfoblastica-acuta-bambino, Agenzia Zoe - AIRC

[3] Che cosa è Multiomics?,
https://www.news-medical.net/life-sciences/What-is-Multiomics-(Italian).aspx, Dr. Surat P - News Medical Life Sciences

[4] Artificial Networks,
https://en.wikipedia.org/wiki/Artificial_neural_network, Wikipedia

[5] Machine learning - Reti neurali demistificate,
https://www.spindox.it/it/blog/ml1-reti-neurali-demistificate/, Spindox

[6] The cluster analysis,
http://www.rescoop.com/multivariata/AnalisiCluster.htm, RESCoop

[7] Steps of the K-mean clustering algorithm,
https://www.researchgate.net/figure/Steps-of-the-K-mean-clustering-algorithm_fig5_321051036, ResearchGate

[8] Comparison between K-Means and spectral clustering,
https://www.researchgate.net/figure/Comparison-between-K-Means-and-spectral-clustering_fig1_319284000, ResearchGate

[9] Gaussian Mixture Models Explained,
https://towardsdatascience.com/gaussian-mixture-models-explained-6986aaf5a95, Toward Data Science

[10] Gaussian Mixture model Cluster,
https://stackoverflow.com/questions/12627338/visualize-gaussian-mixture-model-clusters-in-matlab, Stack Overflow

[11] On Constrained Spectral Clustering and Its Applications ,
https://www.catalyzex.com/paper/arxiv:1201.5338, Catalyzex

[12] Alcuni metodi matriciali per lo Spectral Clustering,
, Serena Marotta
[13] Guide to Neural Networks,
https://www.kdnuggets.com/2016/08/begineers-guide-neural-networks-r.html, KDNeggets
[14] Hierarchical Clustering Explained,
https://vitalflux.com/hierarchical-clustering-explained-with-python-example/, Data Analytics - Data Science, Machine Learning, AI - Vitalflux