11 -Auto encoders

TOPICS	Auto encoders, Neural networks
DATE	@May 25, 2022 2:00 PM
LAST EDITED BY	
LAST EDITED TIME	@May 29, 2022 7:05 PM
MADE BY	
PROF	Calogero
Recording	Calogero Lesson 25052022.m4a
STATUS	✓

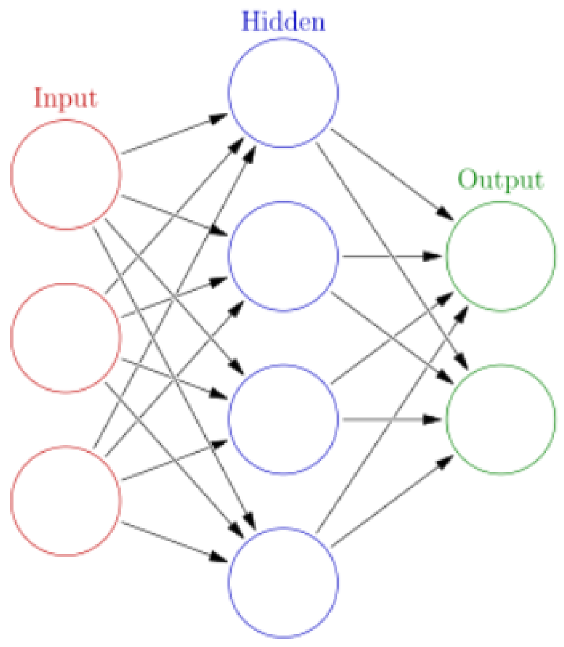
https://s3-us-west-2.amazonaws.com/secure.notion-static.com/13893c52-46bb-4322-8456-25a95b50649d/BMC_-_TECH_C9.pdf

📌 EXAM: 23th June at 9:30am

He will upload an example of exam on Moodle (max Friday) and then one every two days (more or less 4 in totally). There are no solutions for the coding part and he will add the open question for the theory.

There will be one open question of theory.

Today we'll finish the theory part focusing specifically on the **autoencoder usage for single cell analysis**.



Autoencoders are a kind of **neural networks** which is characterised by systems that try to get a solution even if they do not know the optimal path to it. The neural networks try to get a solution without knowing the correct path to reach it. The artificial network is composed by an **inner layer** fully connected to the **input** and then a **output**.In principle, an artificial neural network is characterized by an input, a hidden layer that is fully connected to the input layer and the output that provide the final results of the analysis. The hidden layer could be very complex (in the picture above the hidden layer is relativly simple structure which all the input are connected to each node of the hidden layer and an output of only two nodes). The idea of an artificial network is to train the network with a dataset containing the information that you would like to extract and a dataset that does not contain that information. You are aspecting to optimize the learning, and if is not good you can try to optimaze the hidden layer. What you are actually doing is optimazing the weight that are associated to the connection that connect the input to the hidden layer on the basis of the oputput results. You want to optimize the learning to identify the majority of the events from the data you know. Then when you have generated the network you take and input dataset and try to identify the output. Then you take a new dataset and you search in that what is true and what is false.

These networks require a prior knowledge as the information about what you want to extract from the dataset. You need the information you would like to extract although you don’t know the optimal situation to extract those information.

A limitation of thiese network is that you need two dataset one contaning the information and other without the infomation, and this is mainly used ofr classification porpose. In reality for single cell you don’t know exactly what you want to extract.

Another point is that you need a lot of data to train a neural network, you cannot use simply three replicated to train a neural network, the traning cannot be done. You need 100 of 1000 of samples to run a neuronal network and extract the hidden information. So you cannot train a network with only three replicate of a single cell experiment.

The idea of neural network is to try to reach a solution even without knowing the path to arrive to the solution. All the inputs are linked to hidden nodes, and then the hidden nodes allow to get the output (the information that you want to extract from the dataset, that is not containing this info).

With neural network you can optimize the learning, or you can modify the hidden layers.

Basically, you take a dataset that contain both the layers of information (the ones depicted as input and output in the picture), and another dataset that is not containing the information that you want to extract (so a dataset containing only the input part). Thus, the first acts as a training dataset while the second is the experiment. You need many info to train the artificial neural network to extract the hidden information from your dataset. Usually with training the error rate of the neural network will go down in an asyntotic way, if the curve of the error rate start to flattens earlier probably means that there is something wrong in the hidden nodes definition.

The artificial network try to learn how to discriminate on the difference between the two dataset.

Learning

The learning is based on the weight that you are assigning to each of the edges that connect the input and the hidden nodes. On the basis of the output that you get you try to back correct the weight on the error that you get. Basically you are traning based on reducing the error rate of the know dataset. For example, you start traning the neural network and only the 70% of the real sample can be discriminate and this is not enought. So, you have to do something to correct the neural network and there are two possibility to do that: one possibility is to correct the hidden layer modifyning the fuction that is used to estmate the error rate, or the other possibility is adding more sample. You need to repeat the traning of the neural network until the error rate become constantly (never go to zero).

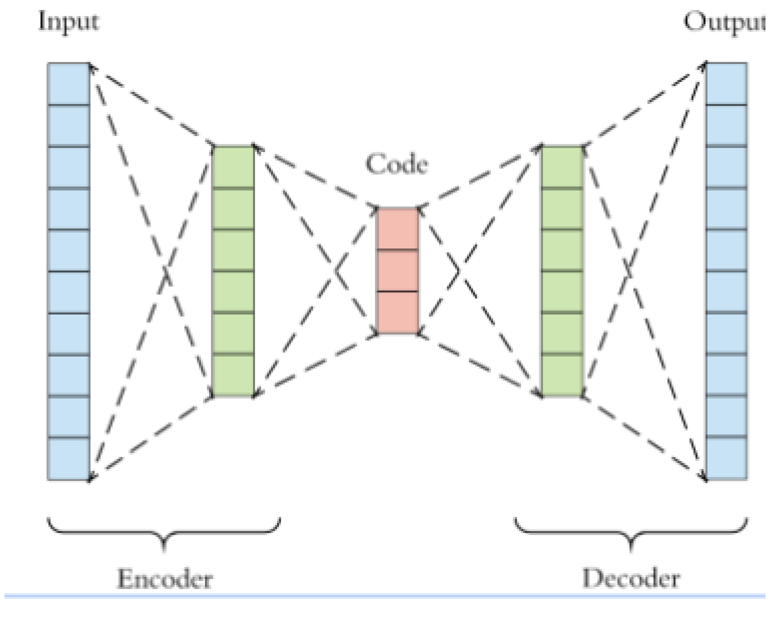
You need to repeat the training till the error rate is low and constant. If you repeat the traning at certain moment the error become flat. The criticality is that if the error get flat but is nor reduce very much (high value of error) this means that the network was not able to learn enoght form the data that you are using. At that moment you have to change completly the structure of the overall network tring to extract the information that you want.

If the error rate goes to plateau but it’s still big it means that the neural network cannot learn from the dataset you are feeding it → you need to change the structure of the hidden layer.

Autoencoder for single cell analysis

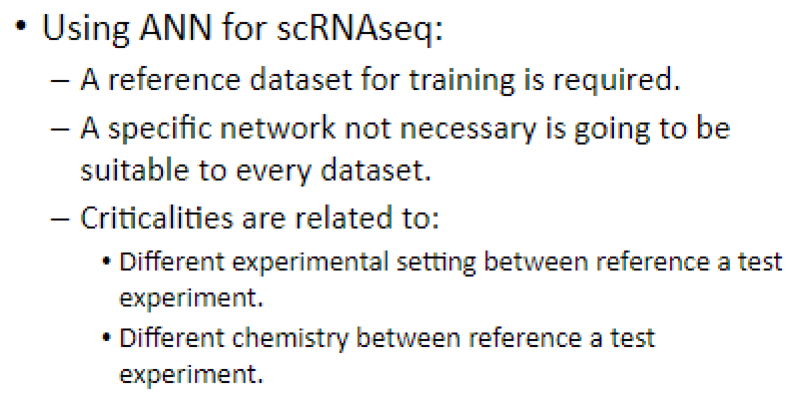
The autoencoders are a specific type of neural network used for single cell analysis.

The neural network used for single cell analysis are Autoencoders. Initially autoencoder were mainly used for the compression of sound for example. mp3 and mp4 sound file are sound file based on the compression using autoencoders.



The overall idea is that you provide as input the data that you want to compress, the data is compressed in a lower dimension and then when you need it can be decompress back to the input information. Dependig on how well is designed the autoencoder the compressed infomation will be able to recunstruct the input material.

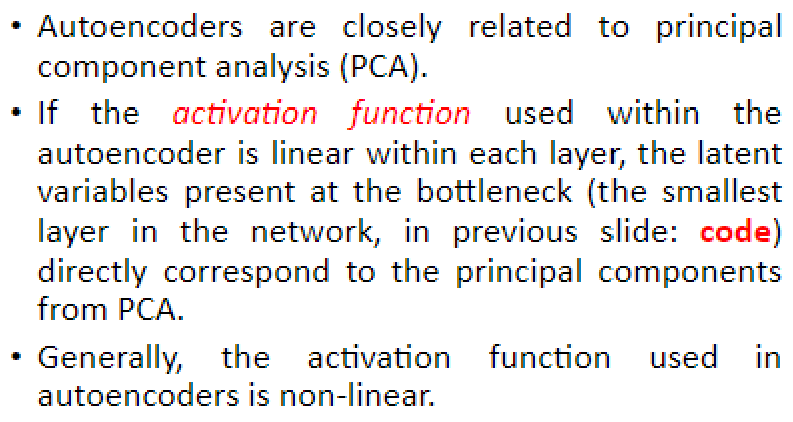
The critical issue for the ANN (Artificial Neural Network) for single cell RNA-seq is that you require a training dataset. Another criticality is that since you are going to have different experimental settings between training set and experimental one you can get some errors.



- Using ANN for scRNAseq:
  - A reference dataset for training is required.
  - A specific network not necessary is going to be suitable to every dataset.
  - Criticalities are related to:
    - Different experimental setting between reference a test experiment.
    - Different chemistry between reference a test experiment.

The autoencoders are based on the **activation function**. The **activation function** is the function used to calculate the weights that are assocaited to the edges that connect the input and the output layer to the hidden layer. The **activation function** are localized at the level of the hidden layer.

If the activation function is a linear function what you get is very similar to what you get with PCA, but this is not what we want becosue we want to extract additional infomation that cannot be see with PCA. What you want is to use an activation function that is not linear, becasue the information that you want to extract cannot be extract simply with PCA.



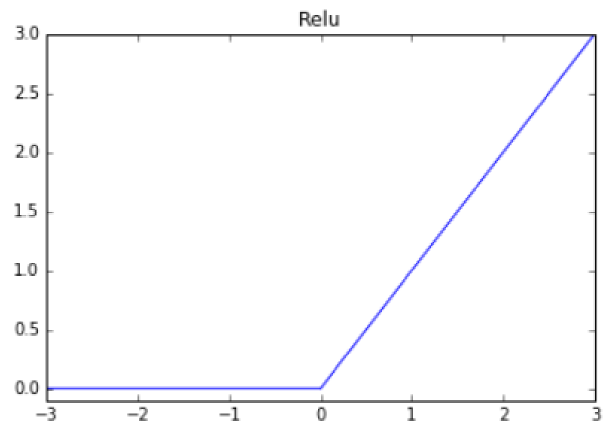
- Autoencoders are closely related to principal component analysis (PCA).
- If the *activation function* used within the autoencoder is linear within each layer, the latent variables present at the bottleneck (the smallest layer in the network, in previous slide: **code**) directly correspond to the principal components from PCA.
- Generally, the activation function used in autoencoders is non-linear.

A typical activation function has to be two important charactheristic:

- non linear
- be able to differatiatable (able to build the derivates of that function)

A typical activation function that is used in autoencoders is the **ReLU (rectified linear unit)** function.

This function is 0 if the input value is lower or equal to 0 and the value become higher than 0 the funcition becames the specific value. The function returns the inputted value if the value is greater than 0.



ReLU (rectified linear unit): This is one of the most popularly used activation functions.

Function:

$$f(x) = \begin{Bmatrix} 0 & for & x < 0 \\ x & for & x \geq 0 \end{Bmatrix}$$

or (another way to write the ReLU function is...)

$$f(x) = \max(x, 0)$$

Derivative:

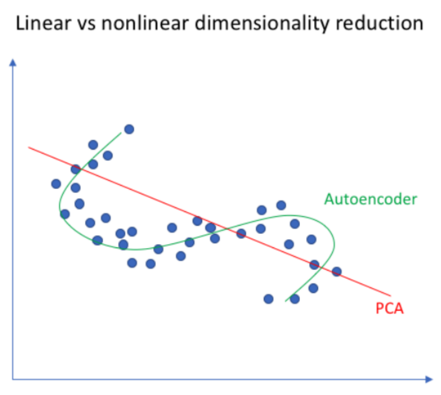
$$f'(x) = \begin{Bmatrix} 0 & for & x < 0 \\ 1 & for & x \geq 0 \end{Bmatrix}$$

//www.jeremyjordan.me/neural-networks-activation-functions/

And if you generate the first derivative: the function return 0 if the information is lower than 0, while the fuction return 1 if the value is higher or equal 0.

The advanatges of the non-linear function is that you can grab non-linear relationship that exist in the dataset while function as PCA that is linear only provide linear information.

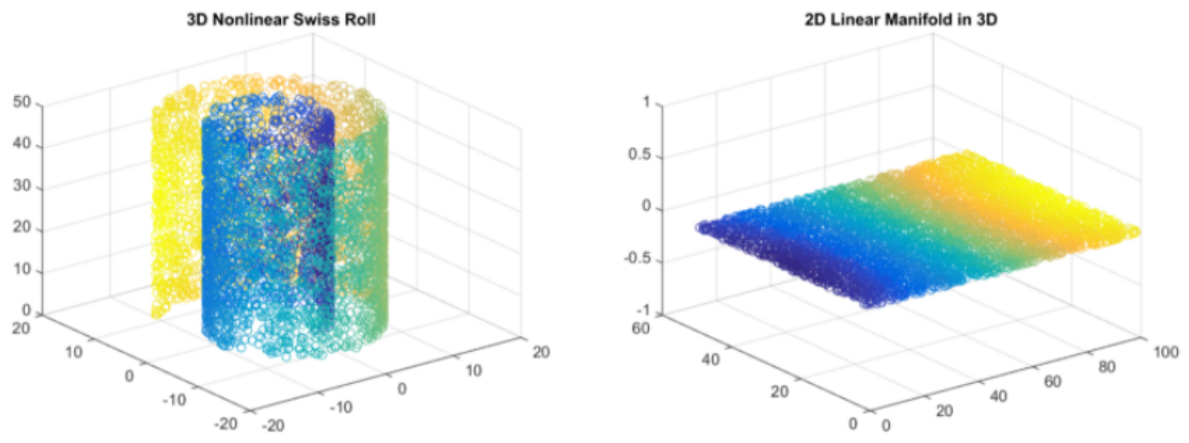
In the graph you see the difference between the PCA and the autoencoder dimensionality reduction. With PCA you loose the shape that is characterizing your data, while with autoencoder you are getting an approximed trend of the dots.



The interesting point is that the hidden layer in which you are compressing the data, is simply representing the data in lower dimentional space without loosing to much information.

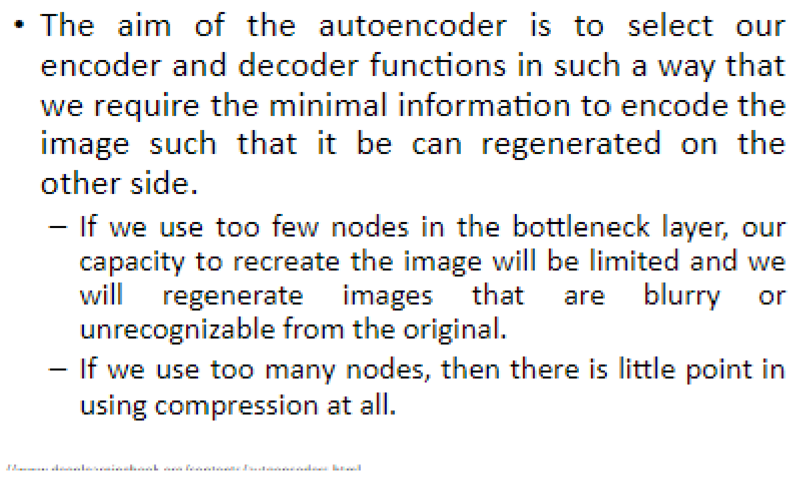
For example the information that you get in the mp3 sound, the soud that you get is not perfecly idential to the one used as input but the compressed file is used simply to have the sound in a smaller space, not to be listen.

The sound that you get out of the mp3 is not exactly identical to the input “recorded” sound since it’s compressed; however the quality is high enough that the normal ear cannot distinguish it from the “real” sound. The compressed version is used to have the sound in a smaller space, but in the compressed space you remove the noise.



The roll is characterised by having different colours progressively changing → you can retransform the info into a 2D linear swiss role maintaining the important features, i.e. the colour change, while the less important “quaternary - roll - structure” is lost.

The exmaple is given by the swiss roll. The swiss roll is charcterized by having different color progressvely changing (from yellow to blue in the inner part). The swiss roll can be transformed in a different non linear space, the important feature of the swiss roll is the change in colour over the stucture. So also with the 2D manifold you are able to represent the same information.



- The aim of the autoencoder is to select our encoder and decoder functions in such a way that we require the minimal information to encode the image such that it be can regenerated on the other side.
  - If we use too few nodes in the bottleneck layer, our capacity to recreate the image will be limited and we will regenerate images that are blurry or unrecognizable from the original.
  - If we use too many nodes, then there is little point in using compression at all.

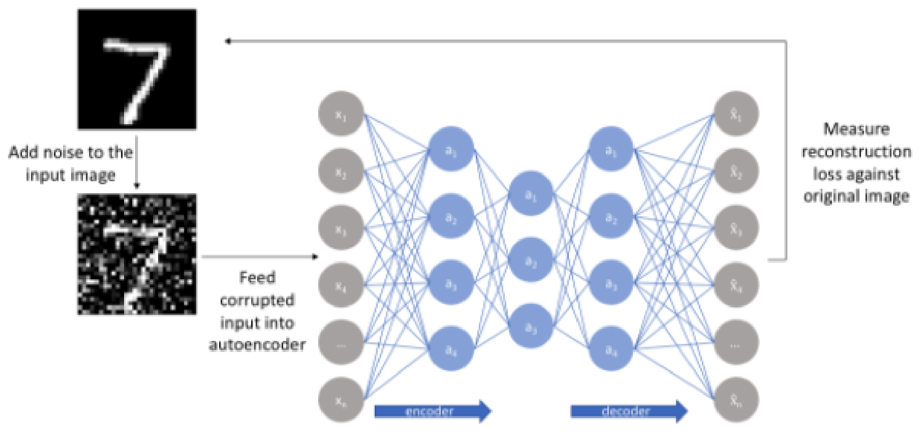
//www.deeplearningbook.org/contents/autoencoders.html

The idea of the autoencoder is to try to find a sort of balance between the size of the input and the hidden layer. If the hidden layer is too small, the compression will be ery little similar to the input, basically you lose a lot of information and the quality drops. If the hidden layer is too big doesn't make sense to do the compression. If the hidden layer is too big, there is no advantage to do the compression.

Autoencoders compromise the dimension of the hidden layer in order to compress the input data and get the output more similar to the input data.

Denoising autoencoders

A typical use of autoencoders is building up picture, but the same concept can be apply to single cells. Denosing the single cell data is like denosing an image.

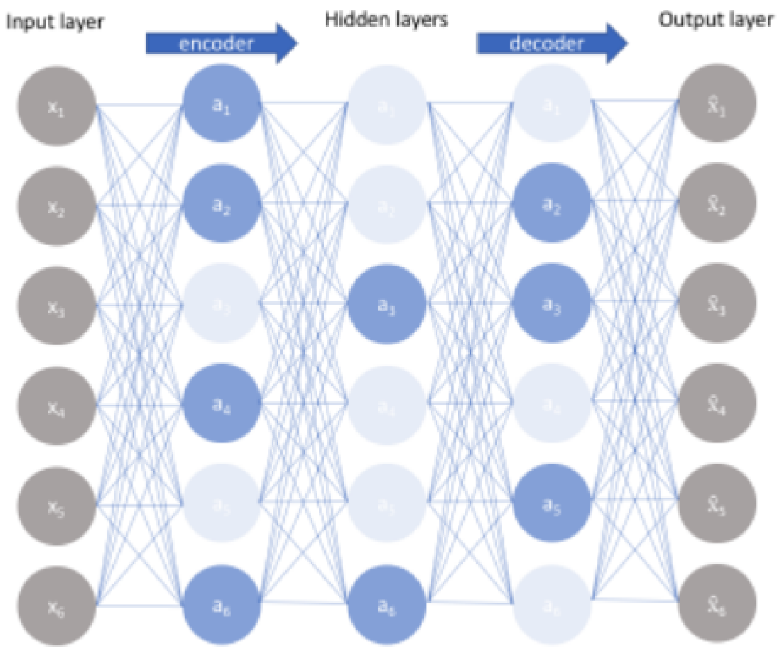

Add noise to the input image
Feed corrupted input into autoencoder
Measure reconstruction loss against original image
encoder
decoder

For example in the image above you have a picture with the number seven that is rich of a lot of noise. The ability of the hidden layer is to compress the most representative information. The most representative information is the path that charactherized the highest signal. You have to consider that you are not passing a single images, but you are passing thousands of images of that seven. The autoencorder learn what is the important part of the picure an remove all the part that is random noise (the system will learn how to depict the area with the signal and the areas with just noise). This can be made by training the autoencoder with many similar images, and then it became able to identify the image and how to exclude the non useful information.

This kind of autoencoder is veri simple, you have the input then the hidden layer for the compression and for the decrompression you do exaclty the same.

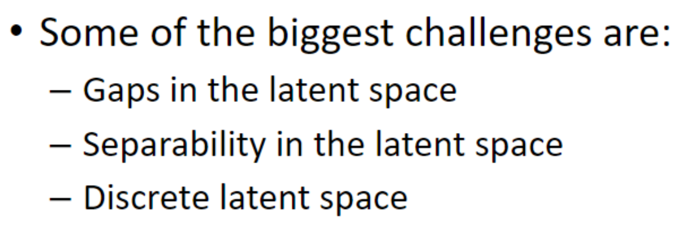
Sparse autoencoders

Another possbility to use the autoencoder is the **Sparse Autoencoder**. In this case the concept of having that the hidden layer is smaller the input layer is not necessary. In this case the hidden layer that is much larger then the input one, but not all the nodes of the hidden layer are used during the traning, only a small part. basically any time you do the traning only a subset of nodes are used. This means that some nodes are never used and this means that are not important. There are some hidden layer nodes that won't be used, "fired", and some that will be fired always.



Autoencoder limitations

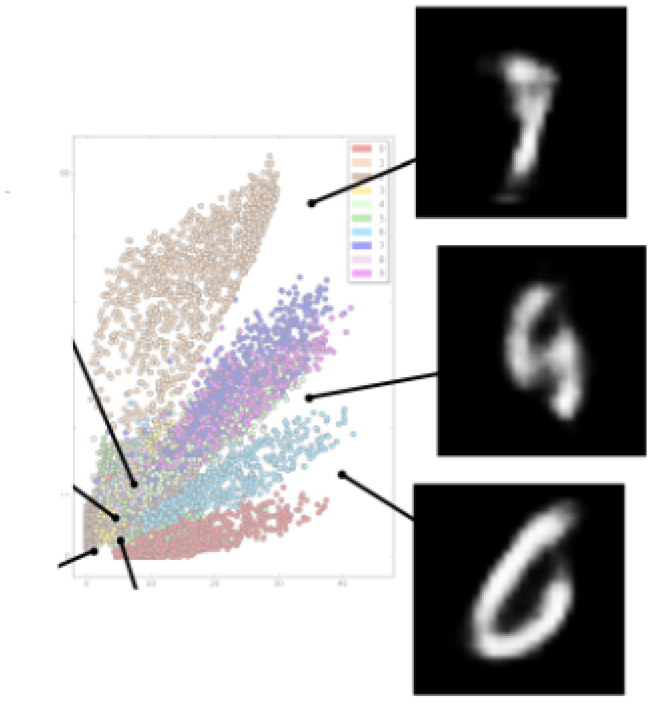
Autoencoders seams to be good but show some limitation.



- Some of the biggest challenges are:
  - Gaps in the latent space
  - Separability in the latent space
  - Discrete latent space

Autoencoder have also some criticality.

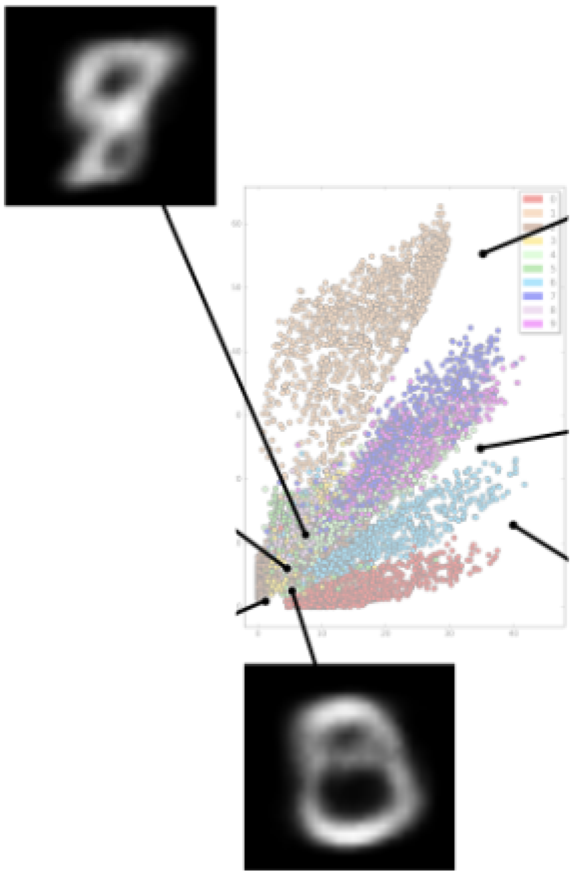
- **Gaps in the latent space**:



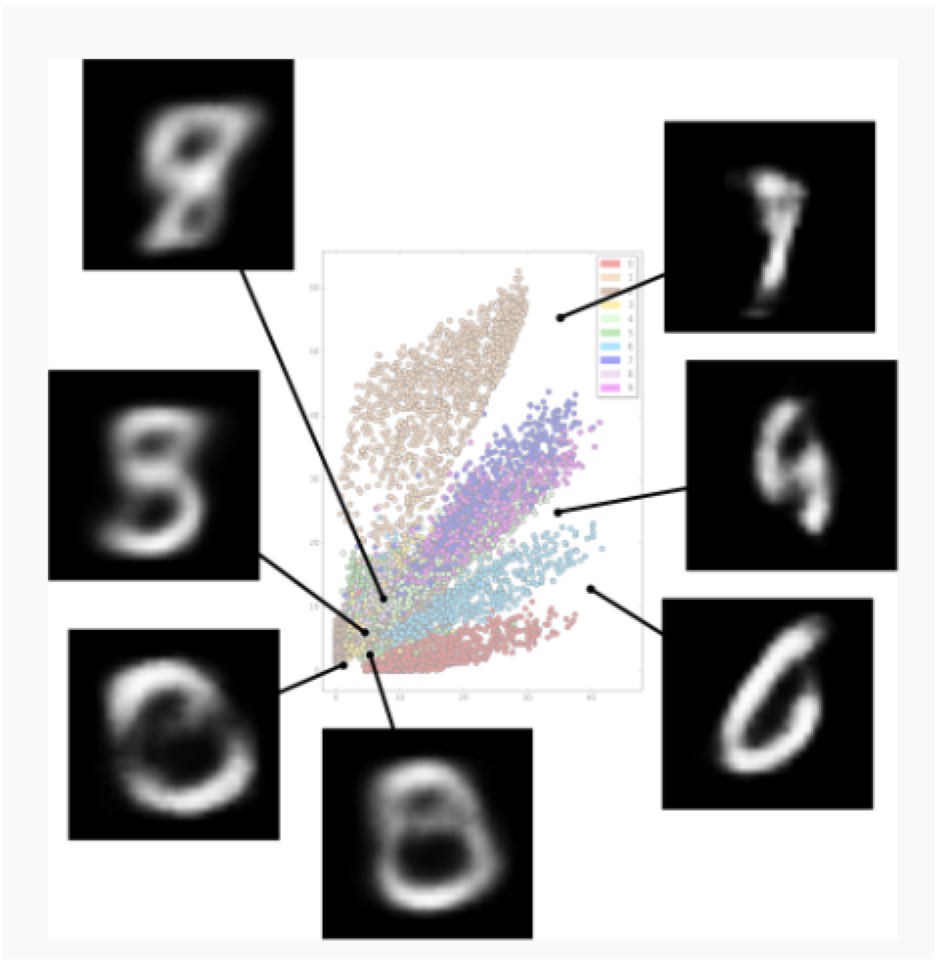
This above is an image of the autoencoder space. If you use autoencoders to clean up pictures with hand written numbers, the information that are eliminated by the hidden layer (for the compression) are lost. So not all the hand written charcter are icked up by the hidden layer. Depending on how is designed the hidden layer try the best to discriminate the separation of the information that you want to compress, but is same case when the picture drops in the area in which there are no information or partially overlaps, that information are lost.

- **Separability in the latent space:** lack of separability of the area in the latent space

  There is overlap between 9 and 8 causes some difficulties. So also in case of partial overlaps the information can be lost.



- **Discrete latent space**



Sparsely-connected Autoencoders for Gene-Set projections

They develop a new kind of autoencoder at MBC and was published last years.

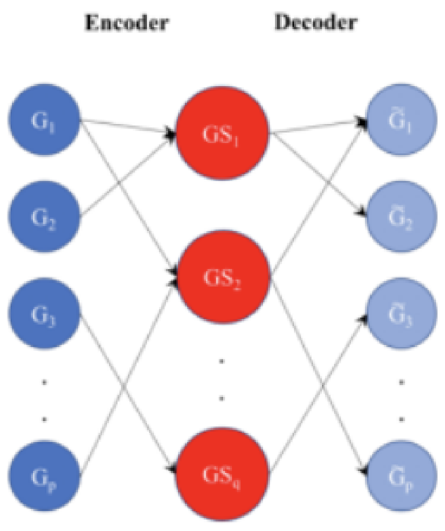
The criticality of teh autoencoders is that since the input and the output are fully connected with teh hidden layer, you cannot associate any specific characteristics to the nodes in the hidden layer. Each nodes of the hidden layer get informatiion from all the nodes of the input layers, therfore you have no idea if a specific nodes of the hidden layer is more linked to a specific feature.

For this reason they decide to build up what is called **sparsely-connected autoencoders.**

The sparsely-connected autoencoders is cahracherized by having the hidden layer not fully connected, but is connected only on the basis of specific biological features. the biologial feature can be kinesis, and the connection are only the genes that are target of that kinesis. Using the trascription information that we are collecting, we try to extrapolate the information that are comming from the upper layer. The hidden layer can be also transcription factors and the input are only the gene that can be activated by those transcription factor.

You are in a situation in which the hidden layer is connected with the input and the output basen on the feature that characterized that gene.

For example you have the gene G1 and gene G2 that are connected to the GS1 trascrption factor, but G1 is also connected with the GS2 trascription factor (see image below).



The advantage of that system is that you can grab information of the functional biologial feature that are linked to the input trascription infoamtionn that you have colled form single cell.

What they did is to use information that were experimentally validated. What they used was different databased (picture below) that contain only infomation that were experimantally validated.

https://bioinfo.uth.edu/kmd/

KinaseMD: Kinase mutations and drug response

https://mirtarbase.cuhk.edu.cn/~miRTarBase/miRTarBase_2022/php/index.php

miRTarBase

https://www.grnpedia.org/trrust/

TRRUST version 2

Transcriptional Regulatory Relationships Unraveled by Sentence-based Text mining

http://www.cuilab.cn/transmir

TransmiR v2.0 database

The first databed for transcription factor and kinase and the other for transcription factors and miRNA relationship.

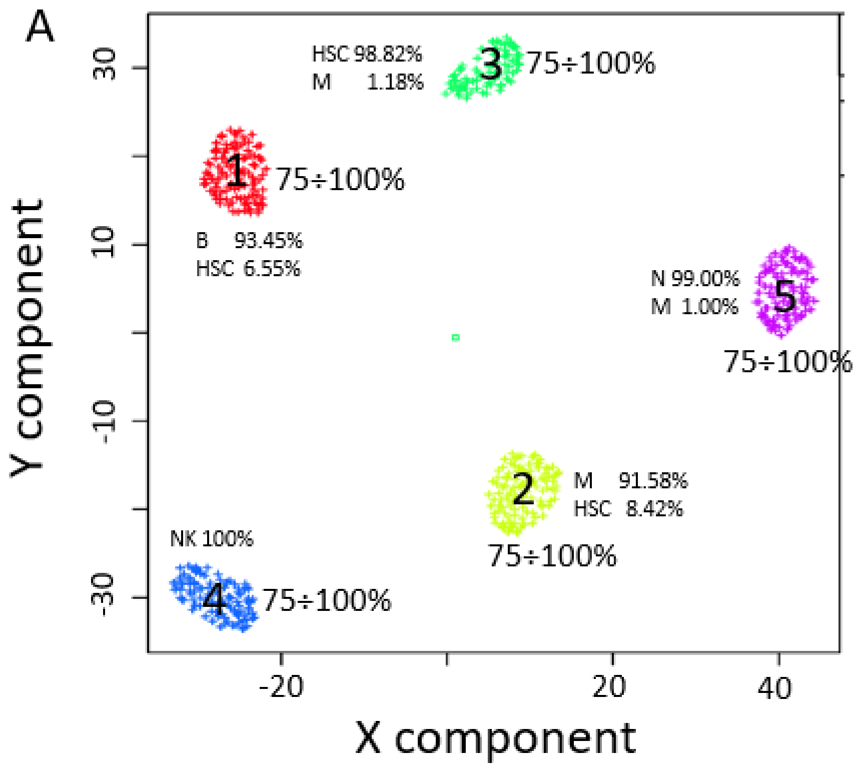
There is an advantages of this representation. For example, until now miRNA cannot be depicted by single cell analysis, but since miRNA are controlling the protein products that is coming from that RNA, we can extrapolate information on miRNA on the basis of the trascriprional information that you get with single cell. If I get the traget gene epxressed this means that the miRNA targeting this gene is not expressed, thus miRNA can be indirectly studied.

With this models it is possible to infer the miRNA expression considering the target gene expression; for TF you can do the same by looking at their target genes.

The first artivle that they build up was based on the idea to extarpolate information from an already available cluster. They want to extrapolate informaiton on the upper leayer regulator that are characterising that specific cluster.

The idea was: first do clustering and generate a set of cluster and then try to recunstruct that cluster picture with the upper layer regulators e.g TF.

For example, they do the clustering on a dataset using all the genes and they get 5 cluster.



Cluster 1 → B cells

Cluster 2 → Monocytes

Cluster 3 → stem cells

Cluster 5 → naïve T cells

Cluster 4 → NK cells

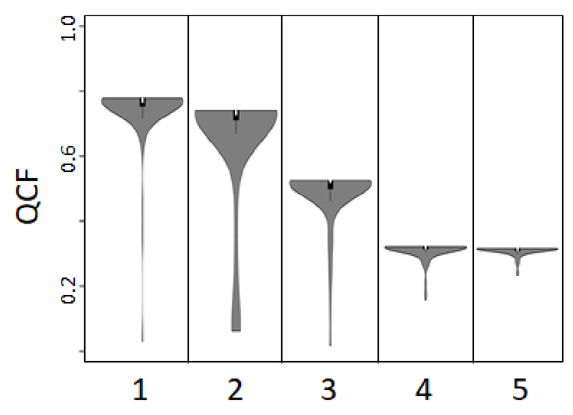
The question is: *are we able to recustrust this clusters using only TF?*

To do that we need some quality controll to understand is we are able to recustruct the initial picture.

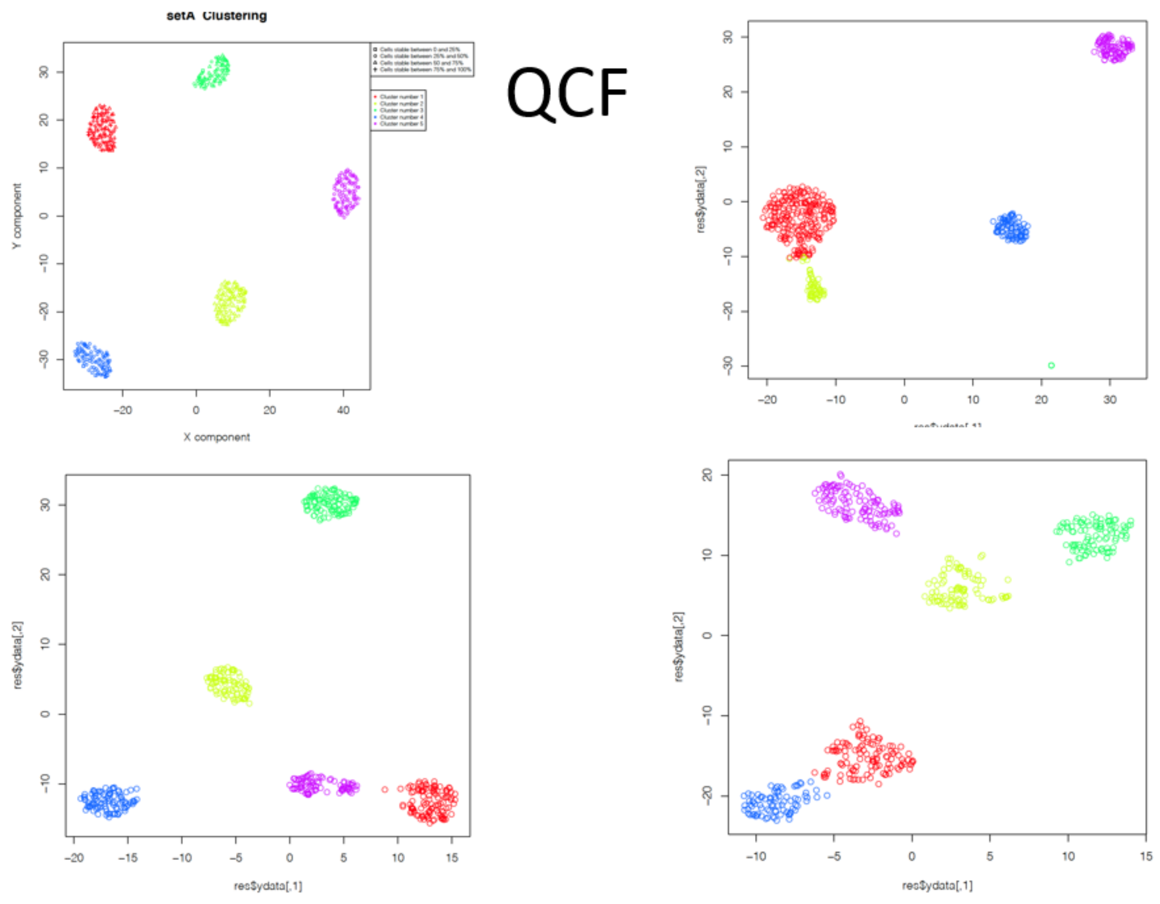
1. QCF (qualiy control feature matrix)

The QCF is obtaned raniking multiple times with the autoencoders and checking if the hidden layer infoation are able to rebuild the structure that you have at the level of the genes. If the TF characterised very well the cluster, theh autoencoders should be able to recustruct the original clsuters. Since any time you are traning the autoencoder any time the traning start from a complete random set, you are actually testing if the input data is able to be compress the data structure.

The QCF is measuring the number of times that you are able to reconstruct the initial data structure during the training.



In this case only 3 (n1, n2, n3) out of the 5 clusters can be reconstructed (have the QCF value higher then the 50%), the other two clusters do not have enough information at the level of TFs to build the same clustering.



Looking at the QCF you see in the fisrt panel the clusring with all the genes, while the other panel are the ones generated with the autoencoders and you have to compare this ones with the clustering done with all the genes.

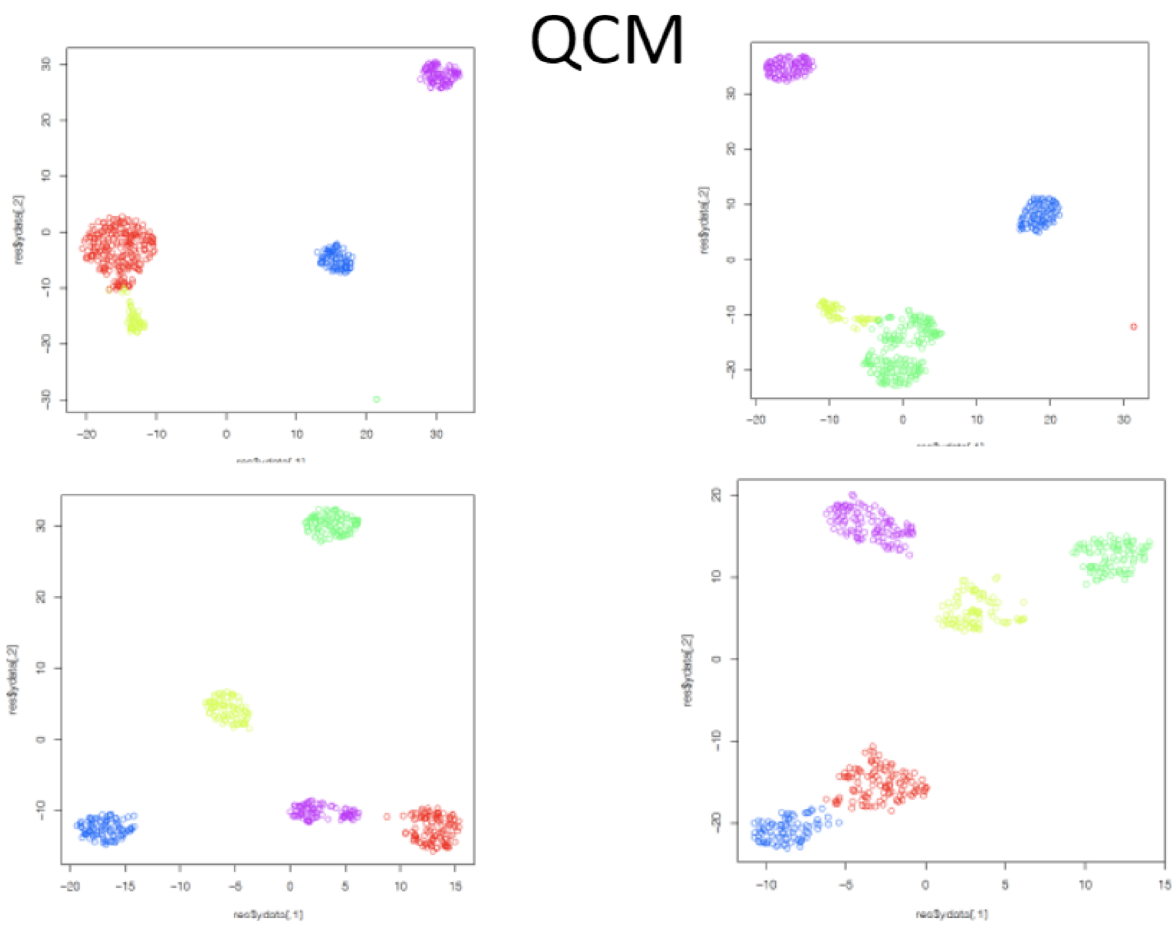
2. QCM (quality control of model similarity)

Another qaulity controll is the QCM. Since the training is repeated multiple times (you get get any times different picture), you can measure how many times pairs of the models are similar to each other.

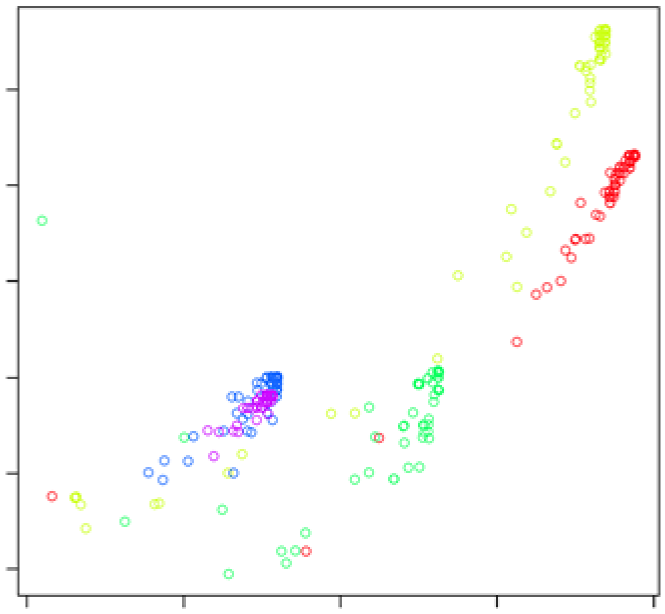
Only if in more than 50% of training you are able to reconstruct the cluster, i.e. 50% of your reconstructed clusters are similar to each other, it means that the cluster can be reconstructed.

In this specific example only the cluster 1 and 2 were able to be recustruct.

This is an example of model that are generated. By taking 4 random sets , you can observe 3 well separated clusters (Don’t look at the color but at the structure).

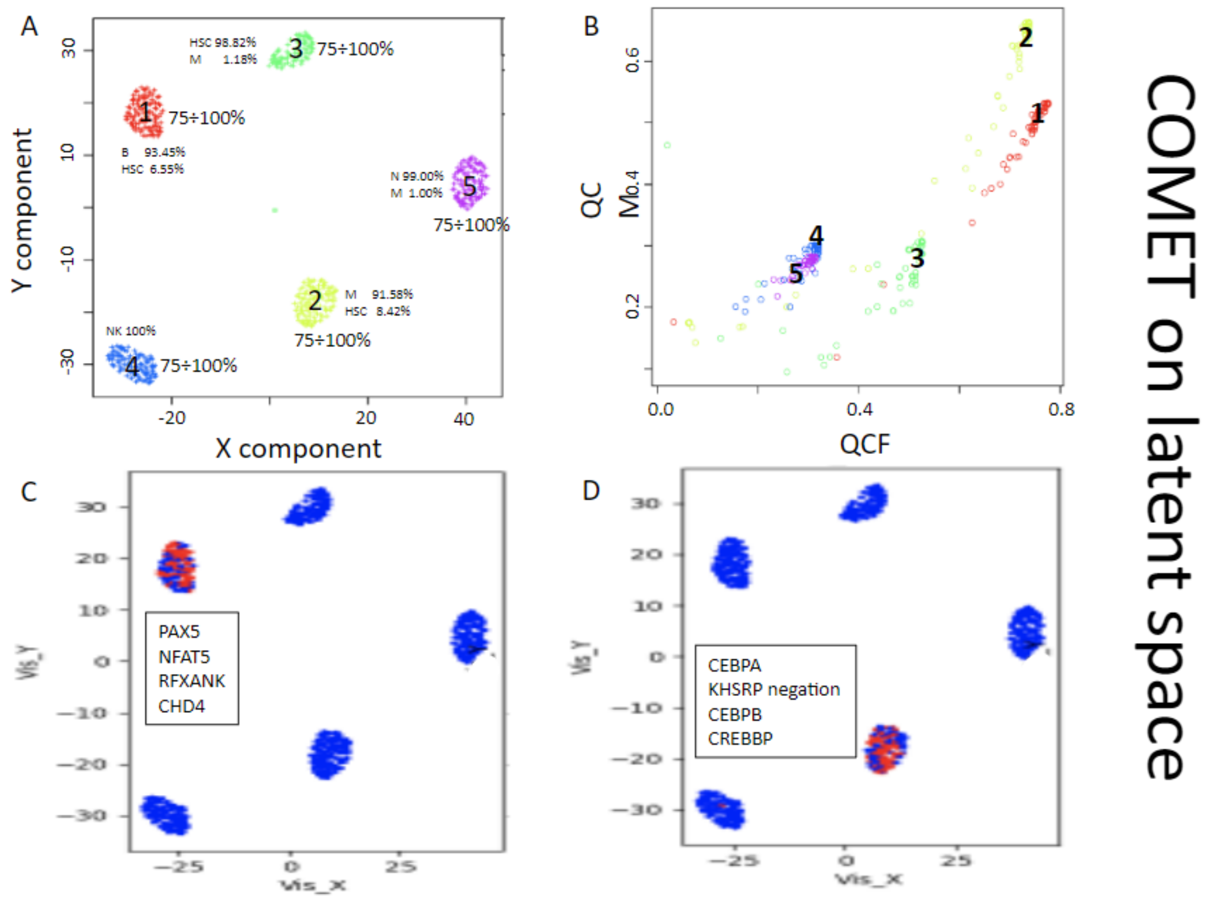


Cluster 1 and 2 are the only two that are able to pass both the treshold (greater than 0,5 for both) for QCF adn QCM.



The advantages of this approach is that ones you have the recustruction you can run COMET over that to identify the specific genes that can better separat among clusters.

COMET on latent space

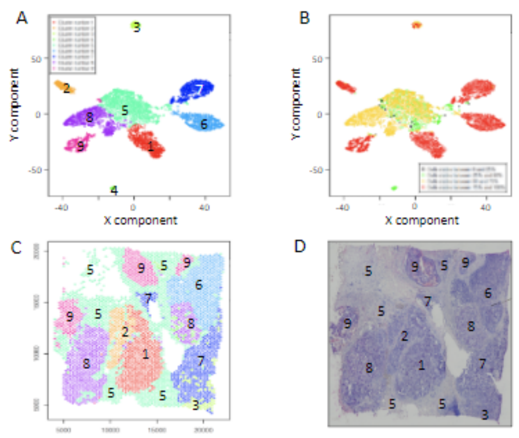


We run COMET on those representations of clusters and it allows you to depict specific genes that are the best one to discriminate among clusters.

For cluster 1 you get 4 genes were being collected (PAX5, NFAT5, RFXANK, CHD4) that are 4 TFs that can discriminate this cluster 1 respect to the others. In the case of the QCF we get other genes, (htree expressed and one negation means that is not present in that cluster/dataset). Doing this compression we know that the TFs are the driving force of the cluster 1 and 2 formation. In addiction, by running COMET we can identify a signature specific of each of the two clusters that are governed by transcription factors activity.

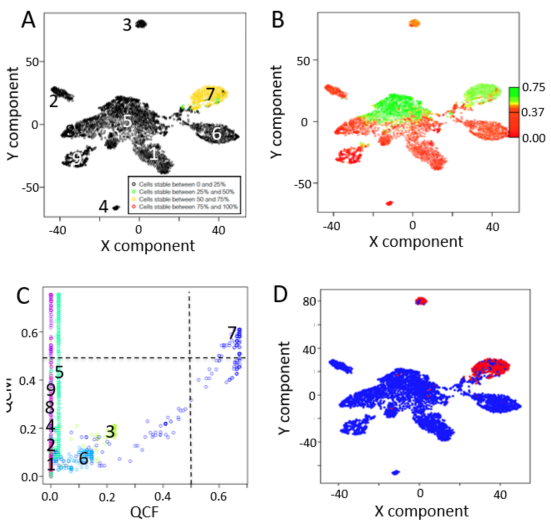
They also took a more complex dataset, a tumor dataset generated by spatial trascriptomic. Basically with this approach you can extract also the information regarding the spatial position of subset of cell of the dataset.

In the A panel you have the clusters generated only by analysing the transcription profile. In C panel you see the spatial position of the cluster in the



tissue and this is also overlay to the eosin staning.

The interesting part is that all these elements are different subsets of the same infiltrating tumour that are characterised by a different transcription profile.



Only using TFs they were able to reconstruct the cluster 7 (panel A). Only the cluster 7 pass the QCF and QCM tresholds (panel C). The model reproducibility was very good for cluster 7 and were only partially good for cluster 5.

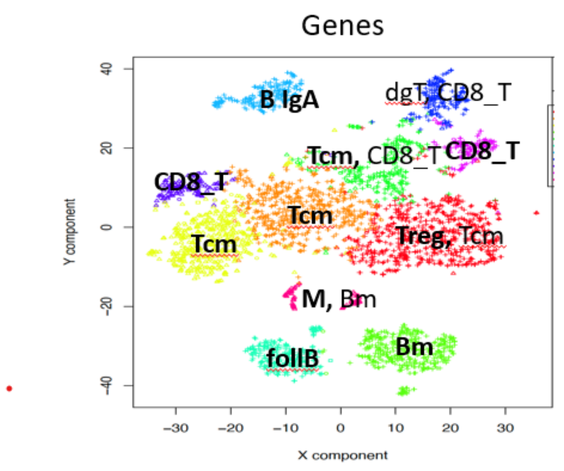
If you are looking to the reproducibility model (panel B) also the cluster 5 displace a good reproducibility. This means that there are some models that are reproduced multiple times but they are not able to recunstruct the initial cluster. The cluster 5 is represent by stromal cells and might represents different amount of infiltrating cell and seam that the cluster that has be done with the TFs is masking some information.

This opened up to change how the analysis was performed: use the autoencoder in a different way. Take first he input data, generate the autoencoder adn then makle clustering. You generate a new set of data clustered on the base of the hidden layer. In order to do this they took the GUT dataset.

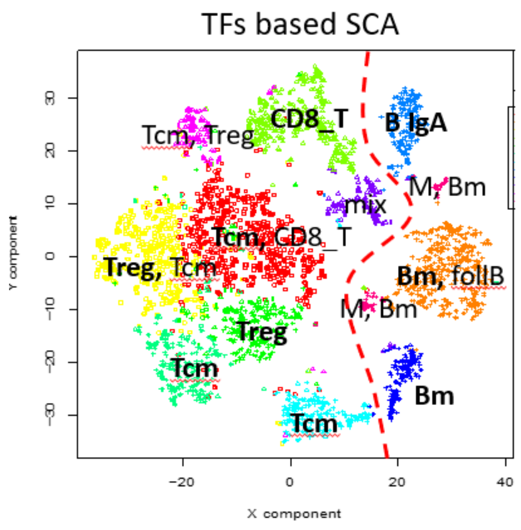
GUT dataset

The immune cells are relatively similar each other and thus are tricky to separate.

The GUT atlas contains already annotated cell information. The colon immune atlas is based on 5 healthy donors from which the immune cells have been extracted from the colon.

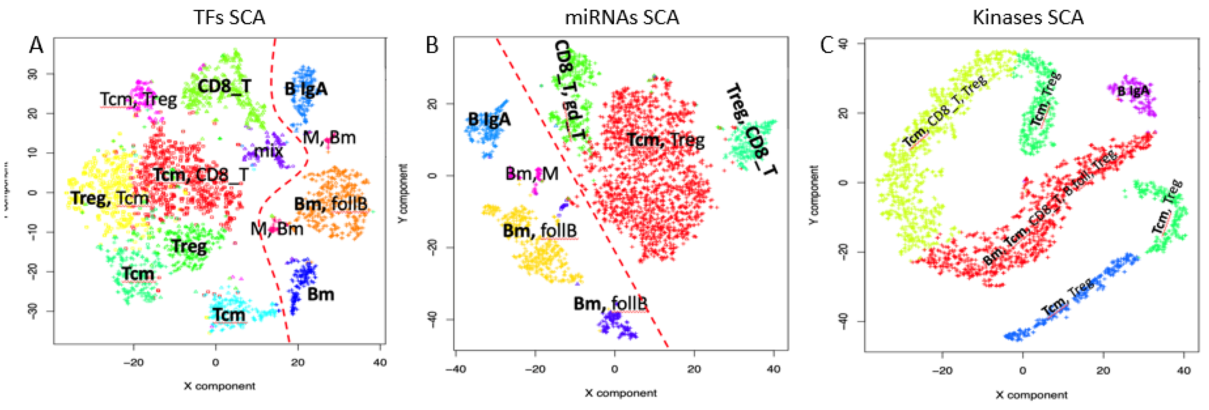


From the clustering (get with all the genes) it is possible to see that the memory cells are splitted between different clusters, while B cells adn T cells are mixed in these clustering (dipsersed amond different clusters).

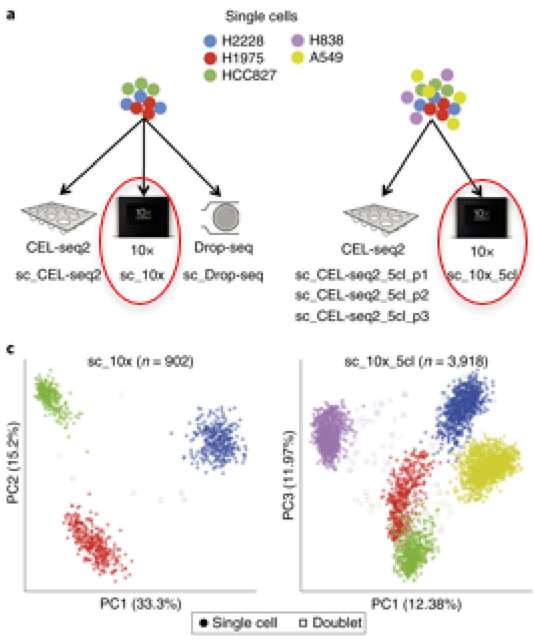


If you took only the trascpriton factor you see that the cell start to be more separated.

All TFs and miRNA are able to make a good separation between group A and group B, while kinases cannot (too far from transcription to really model something at the level of genomic). Both the TF and the miRNA were able to generate a good separation, while kinases are too far biologically to get an info from expression data.



Bechmark dataset



They took another dataset **Bechmark dataset** based on 3 adenocarcinoma cell lines and 5 adenocarcinoma cell lines that were mixed together and analysed by 10X genomics.

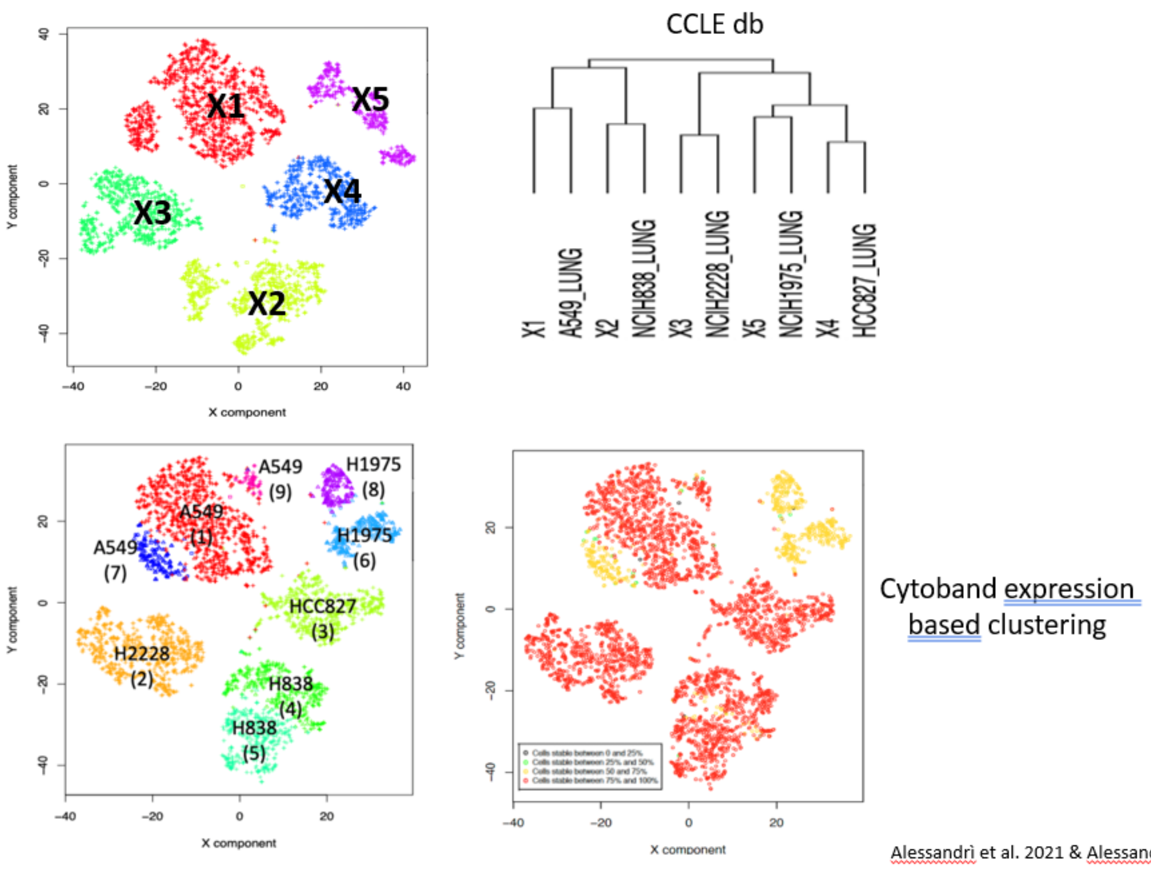
If you at that cell lines, thay are not homogenous and show some clusters, bu tyou can isolate them.

You aspect that in the tumours there is a constant reorganization of the genome. In the tumors there are different cell lines that have different part of the genome swiched on or off based on epigeentics modifications.

As reference they decide to take the cytobands of the human genome (karyotype of healthy donors will give the same structure of those bands). Giemsa bands: karyotipe from healthy people have the same giemsa bands.

So we transform our data on the basis of the aspect of the cytobands: all healthy cells are represented by cells with normal cytobands. If the cytobands have the same organisation in all the cells in the same cell line it means that those cell line have the sam eorganisation and so there are no changes.

If you take a bulk set of cells and compared that to the single cell data, you can compare each cluster to a cell lines.



Alessandrì et al. 2021 & Alessandrì et

When you start to represent the data as cytoband you get a slighly modified organization within each cell lines. This give the possibility to select eah set of cell and investigate the changes in the chromosome that are specific for that subset of cells. By changning the cytobands with something more informative such as IC-data.

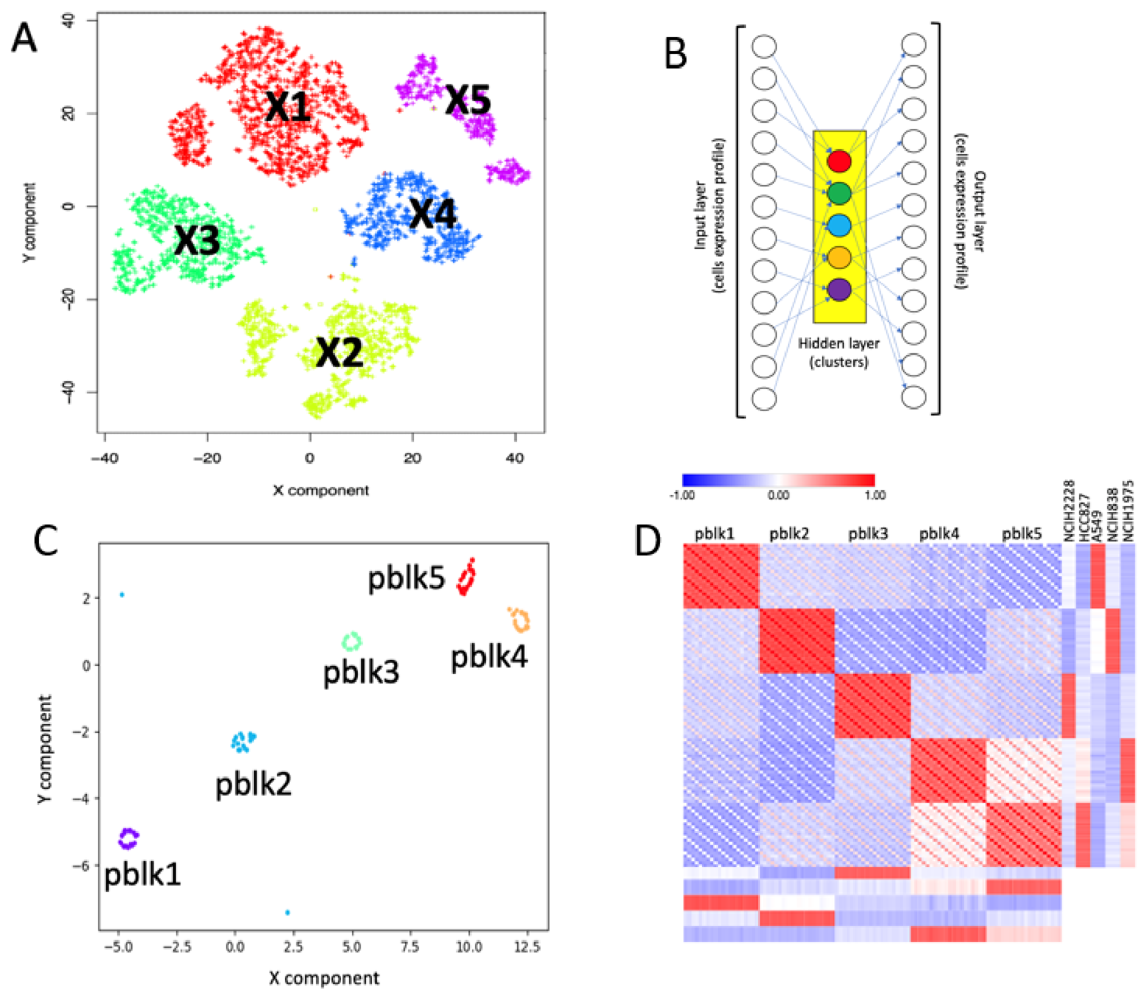
This autoencoder organization gives a lot of power to re-transform the data in something else.

Pseudo-bulk and pseudo-replicates

One of the criticallity of the single cell is that are zero-inflatted. If you put toghether all the cell of a cluster you get a sort of **Pseudo-bulk** representing the bulk expression of the cells of a certain cluster. You can do that, but you cannot used taht to generate autoencoders because you have no replicates.

One interesting data is to use the sparsing-interconnecting autoencoders trasposing the matrix not aplling them on genes but apply to the cluster in which you have reorganize your data. If you put together all the cells of a cluster of scRNA analysis you can get a pseudo bulk to analyse the gene expression using the **sparsely-connected autoencoder** .

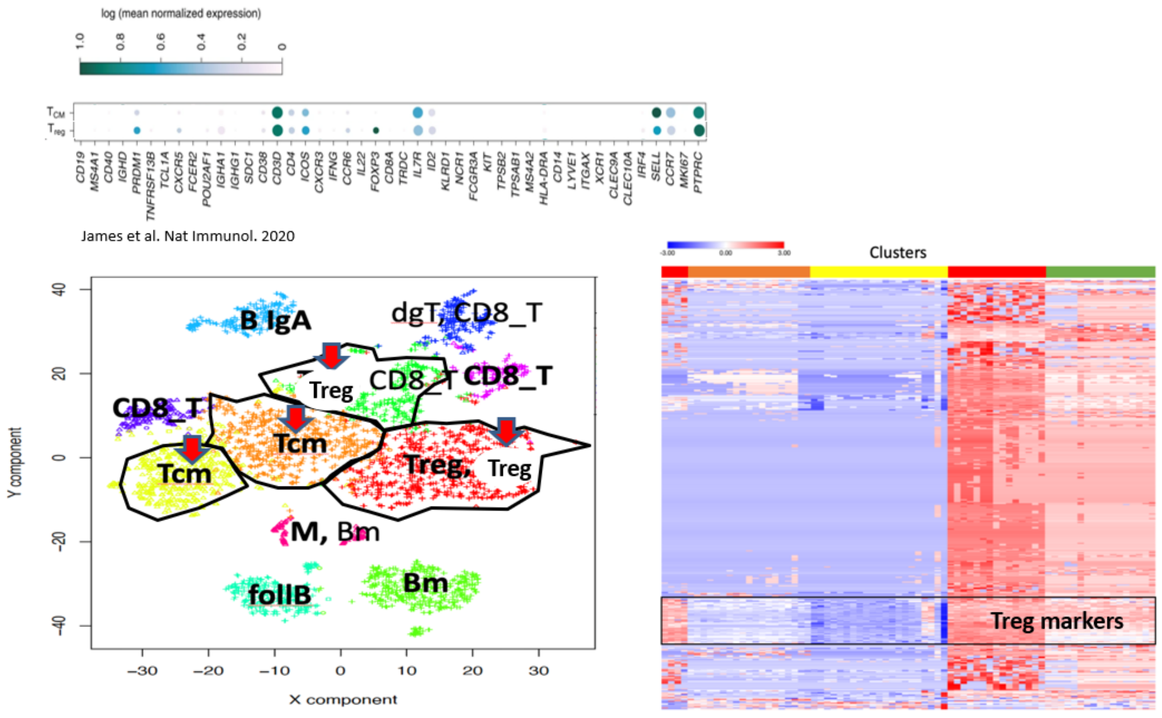
What is the advantage of this approach? Since you repeat this approach multiple times make pseudo-replicates and at the end you get a pseudo-dataset that resempble the bulkRNAseq on which you can aplly the standert DE analysis.



Taking the dataset of 5 different cluster, you can generate the pseudo cluster that corralate very well with the clusters of the trascription profile. You can see in the D panel the different expression profile of the different 5 cell line and clearly correlate very well with the pseudo-bulk (e.g A549 cell line corralete with the pseudo-bulk1).

The pseudo-bulk is a sort of summary of the cluster with a little bit of noise; if the model is too smooth the samples are too similar to each other and you will get artefacts in the DE analysis.
Using this approach if you try to compare the experiment done with 5 cell line with the one made with 3 cell line, you are able to associate the cluster of the 3 cell line to the clusters of the 5 cell lines. Basically you have two indipendent experiment (one with the three cell lines and the other with the five cell lines) and you make a Pearson's correlation score you can build a similarity matrix and associate the cluster of one experiment to the cluster of the other experiment. This allow you do compare different single cell experiment that have something in common.

For example if you take the GUT datatset you get in the picture the clusters created considering all genes. The names that are given to each cluster are done by associating some marker gene expression to a cluster. In the upper part you see how the association of the marker gene to the cell lines is done (e. g. central T memory (Tcm)).



What is strange is that the Tcm are sread in multiple clusters, but you ahve also the Tcm with the Treg. *Is the representation correct?*

They built a pseudo bulk for all this cluster and for the green cluster and the red cluster they found a set of genes that are Treg markers. Using a pseudo bulk they were able to organize the data in a way for which they were able to better annnotate the cell types (reannotate these clusters).

Conclusions

Sparsely Connected Autoencoders (SCA) represent a new way of looking at Deep Learning tools.

Specifically, SCAs offer the opportunity to transform data in a controlled way to grasp from single cell data hidden biological information like relations among cell subpopulations and simplify the inspection of functional relationships among regulatory elements as transcription factors or miRNAs.

The peculiar ability of the autoencoder to retain only the important part of a signal can help in discriminating between true differences among cell subpopulations and clustering overfitting.

The overall conclusion is that these autoencoders sparsely -connected has the advanatage that allow to predict hidden relationship that exist among genes and these relationship can be of different characteristic (localization in the DNA structure, TFs, expression regulation).

The single cell data can be also exploited to extract some functional networks that are peculiar to the clusters that you are analysing. You can use the ANN or AI only if you have high number of data and single cell in the only one experiment that allow to do that.