Geometric Deep Learning for gene networks

FRANCESCO MORANDIN

UNIVERSITY OF PARMA

CuriosAI lab

- Maurizio Parton (University of Chieti-Pescara)
- Carlo Metta (ISTI, CNR Pisa)
- Marco Fantozzi (University of Parma)
- Alessandro Marchetti (Campus Biomedico, Roma)
- Silvia Galfrè (University of Pisa)
- Antonio Di Cecco (University of Chieti-Pescara)f
- ...

Disclaimers

Disclaimers

- Super work-in-progress

Disclaimers

- Super work-in-progress
- There is some biology

Disclaimers

- Super work-in-progress
- There is some biology
- Very applied research

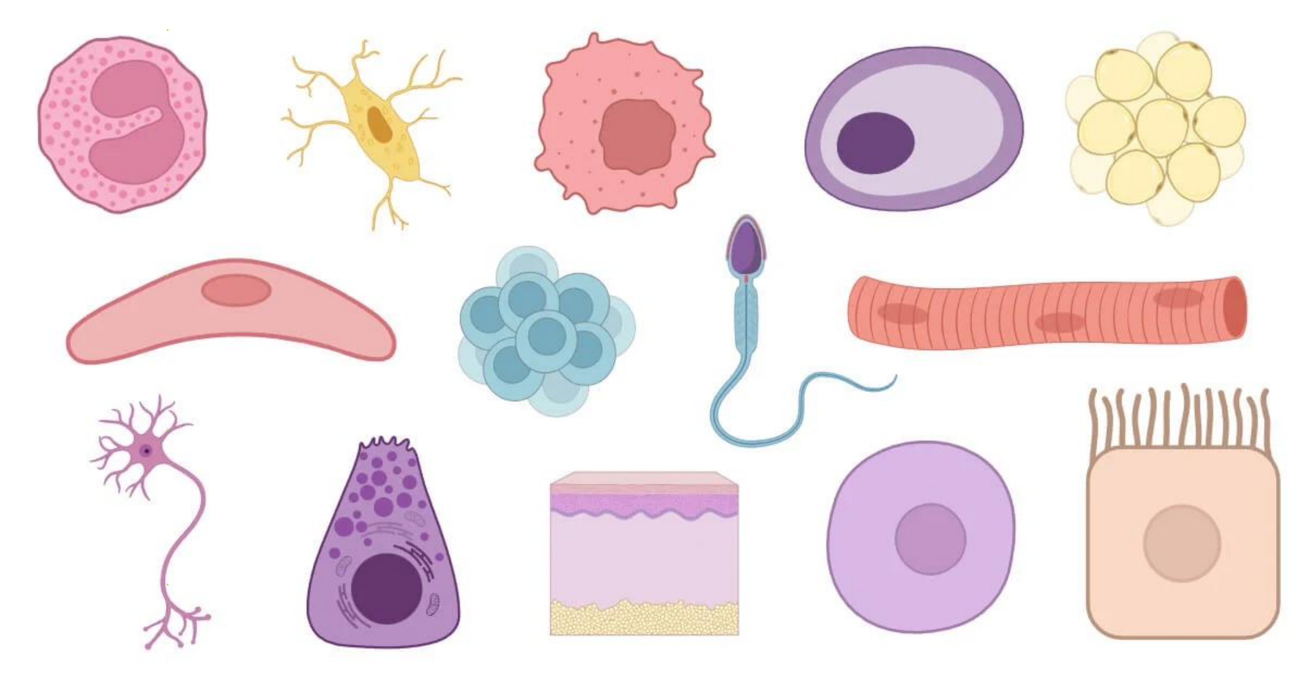
The task: scRNA-seq to cell type

The task: scRNA-seq to cell type

- Cells of an organism have all the same DNA…

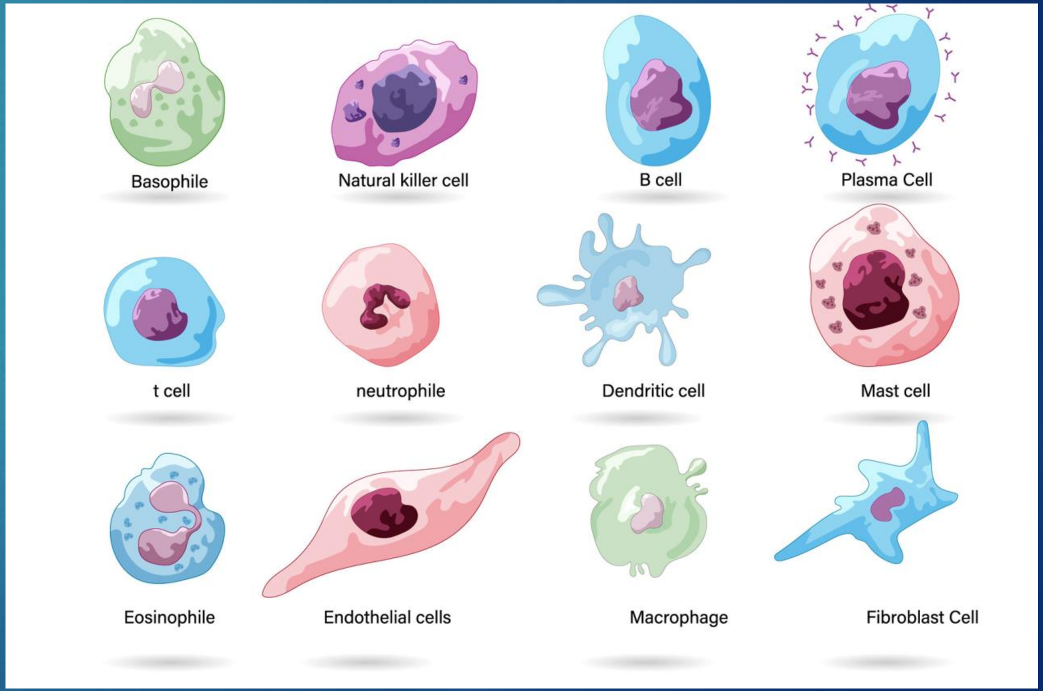
The task: scRNA-seq to cell type

- Cells of an organism have all the same DNA...
- ...but they can be very different!



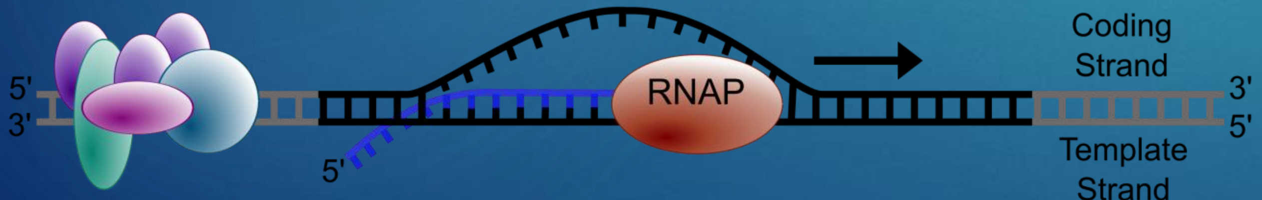
The task: scRNA-seq to cell type

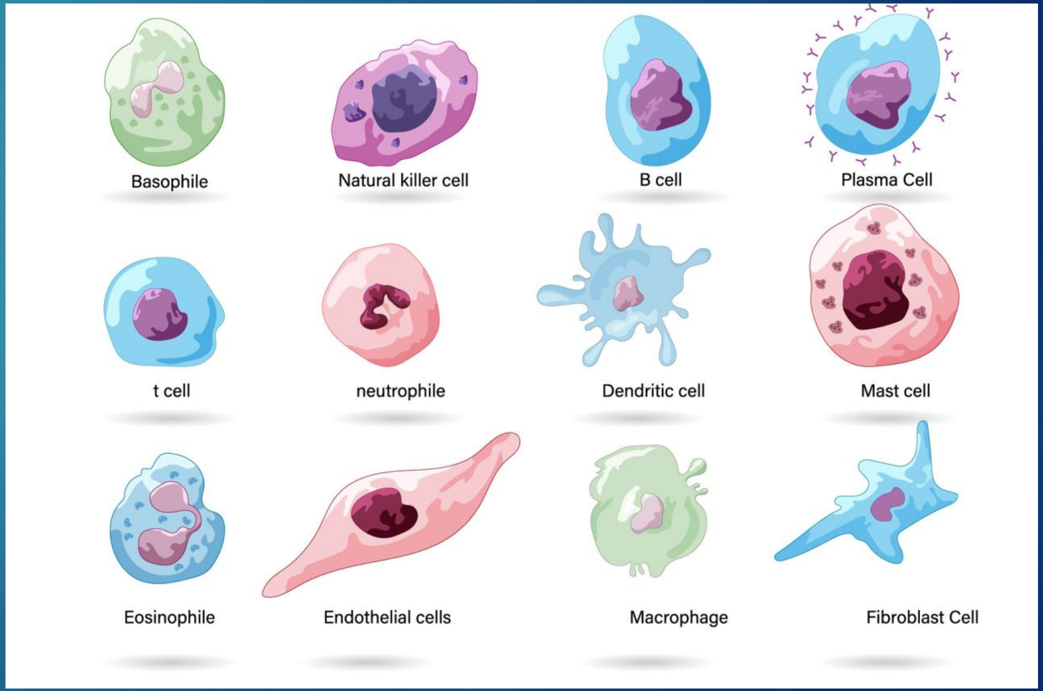
- Cells of an organism have all the same DNA...
- ...but they can be very different!
- Different cell types use different genes from the huge DNA library (genome)



The task: scRNA-seq to cell type

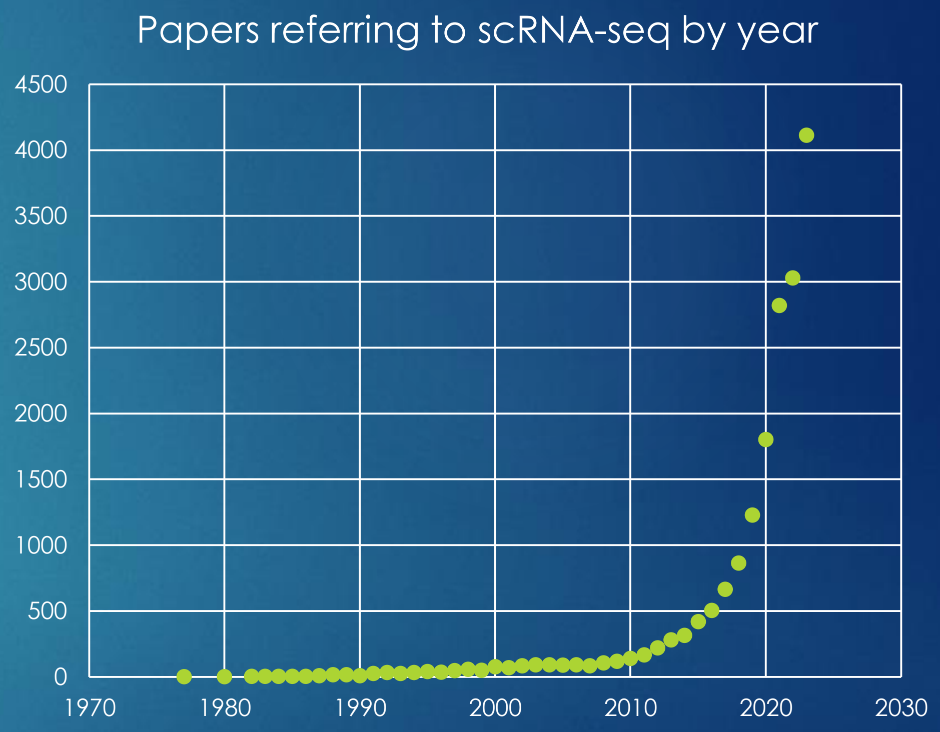
- Cells of an organism have all the same DNA…
- …but they can be very different!
- Different cell types use different genes from the huge DNA library (genome)
- Genes "in use" are transcribed to **RNA**

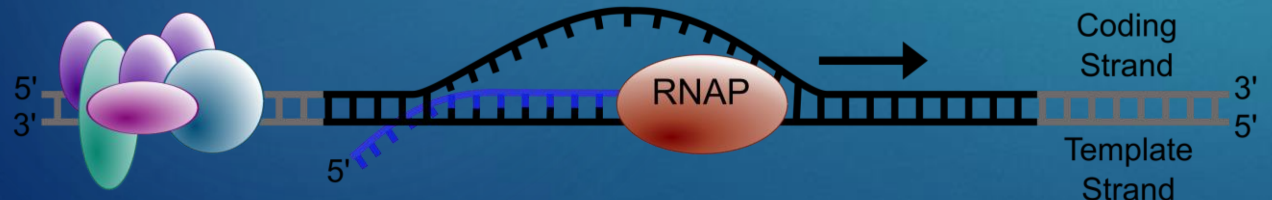




The task: scRNA-seq to cell type

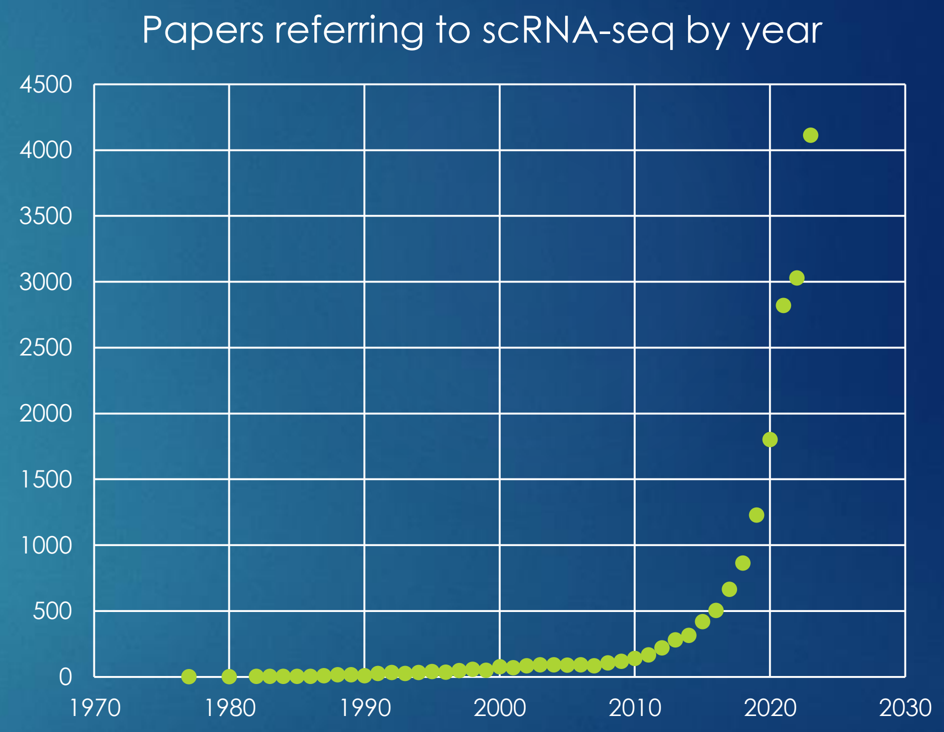
- Cells of an organism have all the same DNA…
- …but they can be very different!
- Different cell types use different genes from the huge DNA library (genome)
- Genes “in use” are transcribed to **RNA**
- Single-cell RNA sequencing (**scRNA-seq**) detects the RNA molecules in the cells





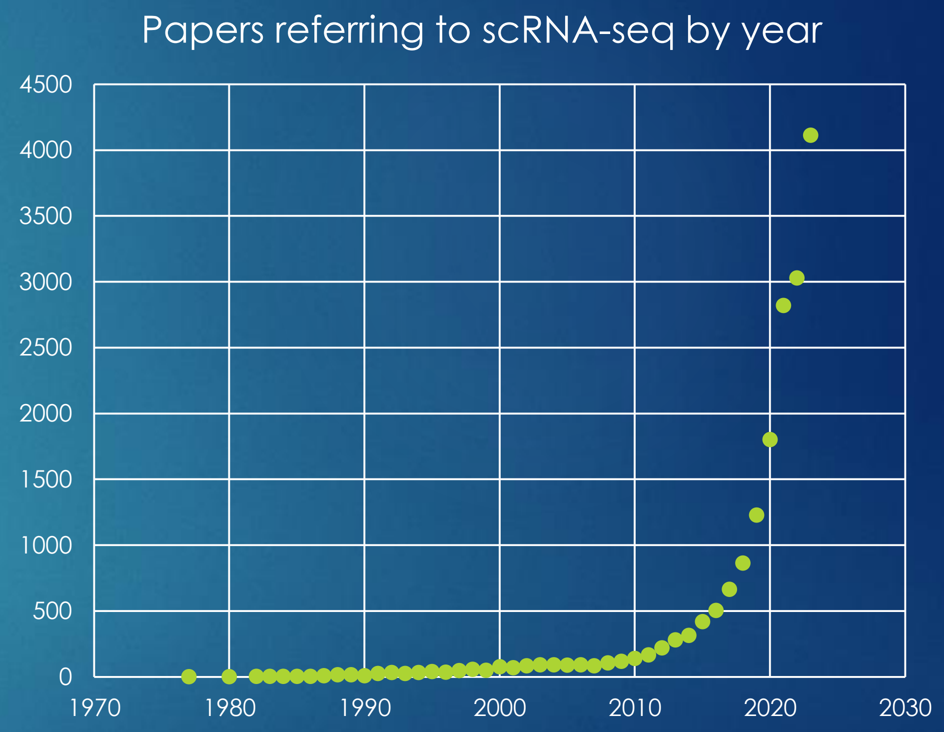
The task: scRNA-seq to cell type

- Cells of an organism have all the same DNA…
- …but they can be very different!
- Different cell types use different genes from the huge DNA library (genome)
- Genes "in use" are transcribed to **RNA**
- Single-cell RNA sequencing (**scRNA-seq**) detects the RNA molecules in the cells
- For each cell in a sample and each gene in the genome we get a molecule count



The task: scRNA-seq to cell type

- Cells of an organism have all the same DNA...
- ...but they can be very different!
- Different cell types use different genes from the huge DNA library (genome)
- Genes "in use" are transcribed to **RNA**
- Single-cell RNA sequencing (**scRNA-seq**) detects the RNA molecules in the cells
- For each cell in a sample and each gene in the genome we get a molecule count
- Task: classify cells by type, from gene counts of RNA



The task: scRNA-seq to cell type

	TUBB2A	ZNF217	SNHG7	STK19	KIAA1324	RNF41	RASA3	ELP1	THRA	LINC01431	label
AAACCTGAGAGTGAGA	0	0	0	0	0	0	1	0	0	0	82
AAACCTGAGGCATTGG	0	0	0	1	0	0	0	0	0	0	19
AAACCTGCACCAGGTC	0	0	0	0	0	0	0	0	0	0	28
AAACCTGCAGGGATTG	0	0	0	0	0	0	2	0	0	0	58
AAACCTGCAGTCAGAG	0	0	0	0	0	0	1	0	0	0	66
...	...	...	...	...	...	...	...	...	...	...	...
TTTGTCAGTTGATTCG	0	0	0	0	0	0	0	0	0	0	37
TTTGTCATCATAACCG	0	0	0	0	0	0	1	0	0	0	63
TTTGTCATCATCATTC	0	0	0	0	0	0	0	0	0	0	34
TTTGTCATCATGCTCC	0	0	3	0	1	1	1	0	0	0	65
TTTGTCATCCACGTTC	0	0	0	0	0	0	0	0	0	0	63

10137 rows × 13055 columns

The task: scRNA-seq to cell type

- Common approach: no neural networks (NN), unsupervised

The task: scRNA-seq to cell type

- Common approach: no neural networks (NN), unsupervised **clustering** [`Seurat`, `Scanpy`]

The task: scRNA-seq to cell type

- Common approach: no neural networks (NN), unsupervised **clustering** [`Seurat`, `Scanpy`]
- Alternative 1: use NN, unsupervised

The task: scRNA-seq to cell type

- Common approach: no neural networks (NN), unsupervised

  clustering [`Seurat`, `Scanpy`]

- Alternative 1: use NN, unsupervised

  variational autoencoder clustering [`scvi`]

The task: scRNA-seq to cell type

- Common approach: no neural networks (NN), unsupervised

  clustering [`Seurat`, `Scanpy`]

- Alternative 1: use NN, unsupervised

  variational autoencoder clustering [`scvi`]

- Alternative 2: no NN, self-supervised with pseudo-labels

The task: scRNA-seq to cell type

- Common approach: no neural networks (NN), unsupervised

  clustering [`Seurat`, `Scanpy`]

- Alternative 1: use NN, unsupervised

  variational autoencoder clustering [`scvi`]

- Alternative 2: no NN, self-supervised with pseudo-labels

  logistic regression [`CellTypist`]

The task: scRNA-seq to cell type

- Common approach: no neural networks (NN), unsupervised

  clustering [`Seurat`, `Scanpy`]

- Alternative 1: use NN, unsupervised

  variational autoencoder clustering [`scvi`]

- Alternative 2: no NN, self-supervised with pseudo-labels

  logistic regression [`CellTypist`]

- Why not to use NN with pseudo-labels?

Transcriptomics challenges

Transcriptomics challenges

- **Sparsity**: most counts are zero, some are hundreds

Transcriptomics challenges

- **Sparsity**: most counts are zero, some are hundreds
- Many genes are expressed at few RNA molecules per cell

Transcriptomics challenges

- **Sparsity**: most counts are zero, some are hundreds
- Many genes are expressed at few RNA molecules per cell
- **Low efficiency**: RNA molecules have a small probability of detection

Transcriptomics challenges

- **Sparsity**: most counts are zero, some are hundreds
- Many genes are expressed at few RNA molecules per cell
- **Low efficiency**: RNA molecules have a small probability of detection
- Many zeros would be non-zero with a higher efficiency

Transcriptomics challenges

- **Sparsity**: most counts are zero, some are hundreds
- Many genes are expressed at few RNA molecules per cell
- **Low efficiency**: RNA molecules have a small probability of detection
- Many zeros would be non-zero with a higher efficiency
- **High dimensionality**: number of genes $N \sim 15k$

Transcriptomics challenges

- **Sparsity**: most counts are zero, some are hundreds
- Many genes are expressed at few RNA molecules per cell
- **Low efficiency**: RNA molecules have a small probability of detection
- Many zeros would be non-zero with a higher efficiency
- **High dimensionality**: number of genes $N{\sim}15k$
- **Non-continuity**: small integers are hardly Gaussian variables

Transcriptomics challenges

- **Sparsity**: most counts are zero, some are hundreds
- Many genes are expressed at few RNA molecules per cell
- **Low efficiency**: RNA molecules have a small probability of detection
- Many zeros would be non-zero with a higher efficiency
- **High dimensionality**: number of genes $N \sim 15k$
- **Non-continuity**: small integers are hardly Gaussian variables
- Expression levels X behave a little better than raw counts R

Transcriptomics challenges

- **Sparsity**: most counts are zero, some are hundreds
- Many genes are expressed at few RNA molecules per cell
- **Low efficiency**: RNA molecules have a small probability of detection
- Many zeros would be non-zero with a higher efficiency
- **High dimensionality**: number of genes $N \sim 15k$
- **Non-continuity**: small integers are hardly Gaussian variables
- Expression levels X behave a little better than raw counts R

$$\{0,1,2,\dots\} \ni R_{i,j} \to X_{i,j} := \log\left(1 + \frac{10^4}{N}\frac{R_{i,j}}{R_{i,*}}\right), \qquad \text{cell } i\text{, gene } j$$

Transcriptomics challenges

Transcriptomics challenges

- Genes can be very different

Transcriptomics challenges

- Genes can be very different
- **Constitutive genes** are needed for all cell types

Transcriptomics challenges

- Genes can be very different
- **Constitutive genes** are needed for all cell types
- **Marker genes** are specific to one type of cells

Transcriptomics challenges

- Genes can be very different
- **Constitutive genes** are needed for all cell types
- **Marker genes** are specific to one type of cells
- **Regulatory genes** switch on/off other genes (low expression)

Transcriptomics challenges

- Genes can be very different
- **Constitutive genes** are needed for all cell types
- **Marker genes** are specific to one type of cells
- **Regulatory genes** switch on/off other genes (low expression)
- Useful to rank genes by variability of expression in the sample

Transcriptomics challenges

- Genes can be very different
- **Constitutive genes** are needed for all cell types
- **Marker genes** are specific to one type of cells
- **Regulatory genes** switch on/off other genes (low expression)
- Useful to rank genes by variability of expression in the sample
- Most methods restrict the genome to the highly variable genes

Transcriptomics challenges

Transcriptomics challenges

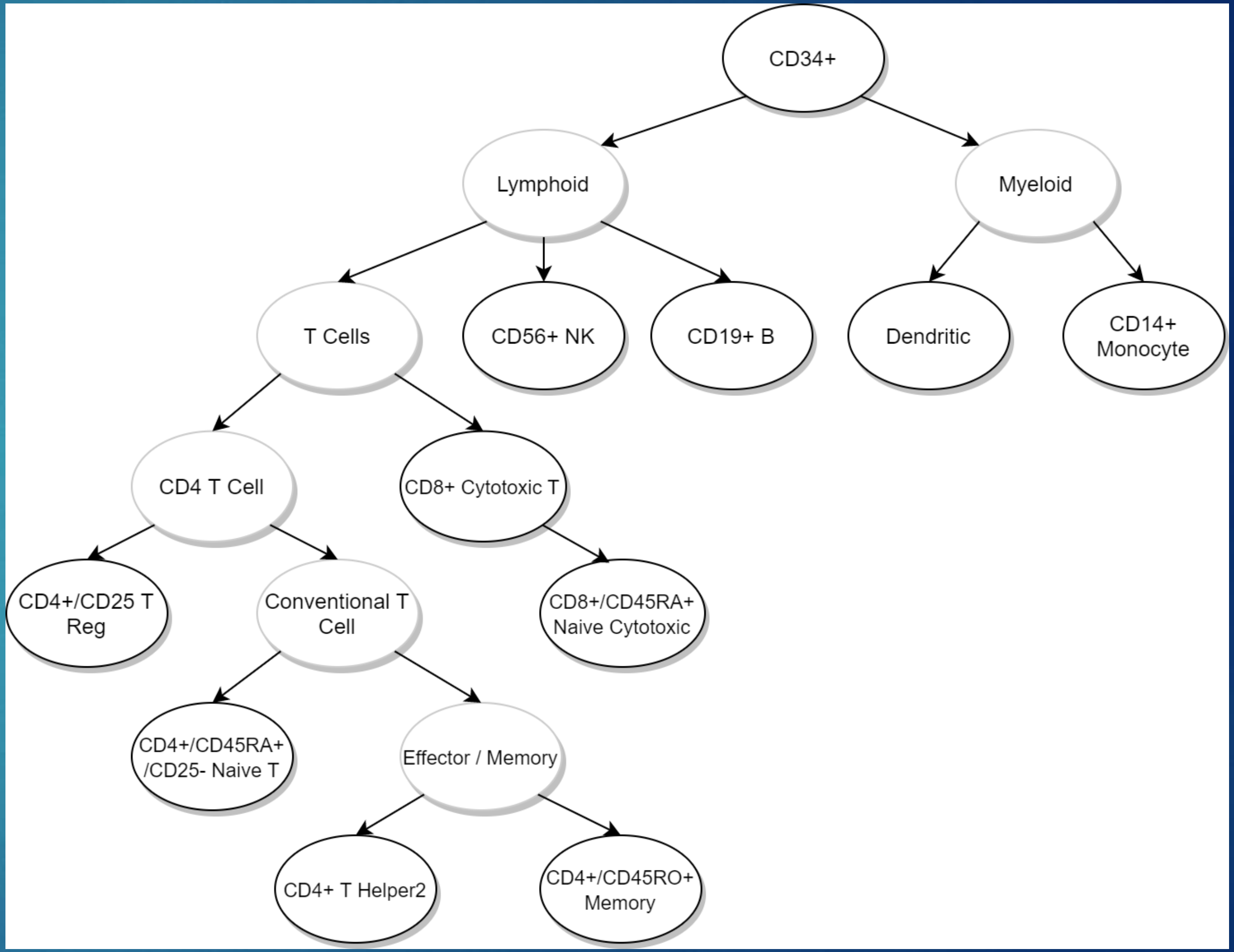
- Cell types are not well defined

Transcriptomics challenges

- Cell types are not well defined
- There is really a continuum of types

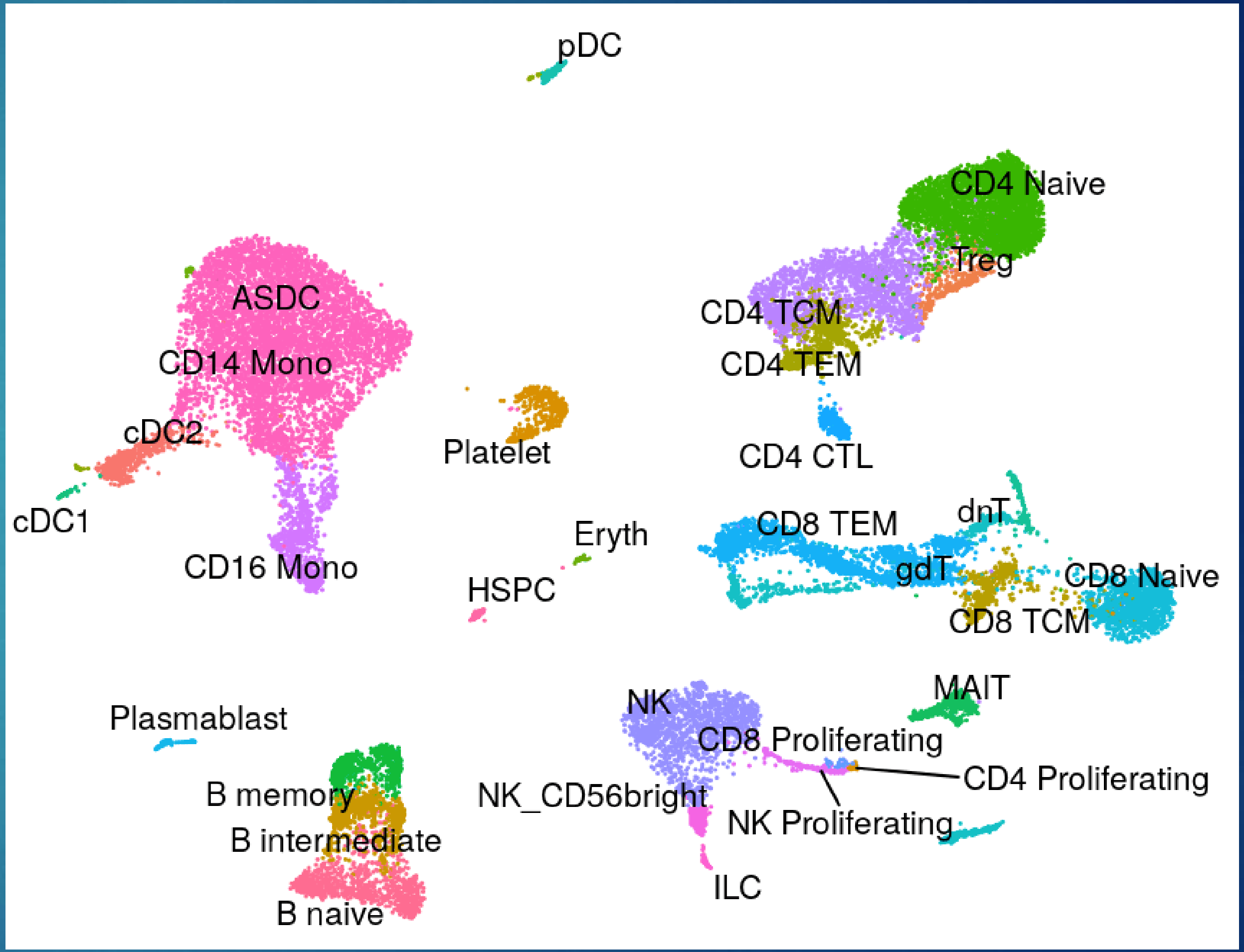
Transcriptomics challenges

- Cell types are not well defined
- There is really a continuum of types
- They show a hierarchical structure



Transcriptomics challenges

- Cell types are not well defined
- There is really a continuum of types
- They show a hierarchical structure
- Clustering yields **pseudo-labels**



The task: scRNA-seq to cell type

The task: scRNA-seq to cell type

- **Expression levels** $X_{i,j} = \log(1 + cR_{i,j}/R_{i,*})$ as input

The task: scRNA-seq to cell type

- **Expression levels** $X_{i,j} = \log(1 + cR_{i,j}/R_{i,*})$ as input
- **Pseudo-labels** Y_i as output

The task: scRNA-seq to cell type

- **Expression levels** $X_{i,j} = \log(1 + cR_{i,j}/R_{i,*})$ as input
- **Pseudo-labels** Y_i as output
- Some neural network as classifier

The task: scRNA-seq to cell type

- **Expression levels** $X_{i,j} = \log(1 + cR_{i,j}/R_{i,*})$ as input
- **Pseudo-labels** Y_i as output
- Some neural network as classifier
- There is no foundation model

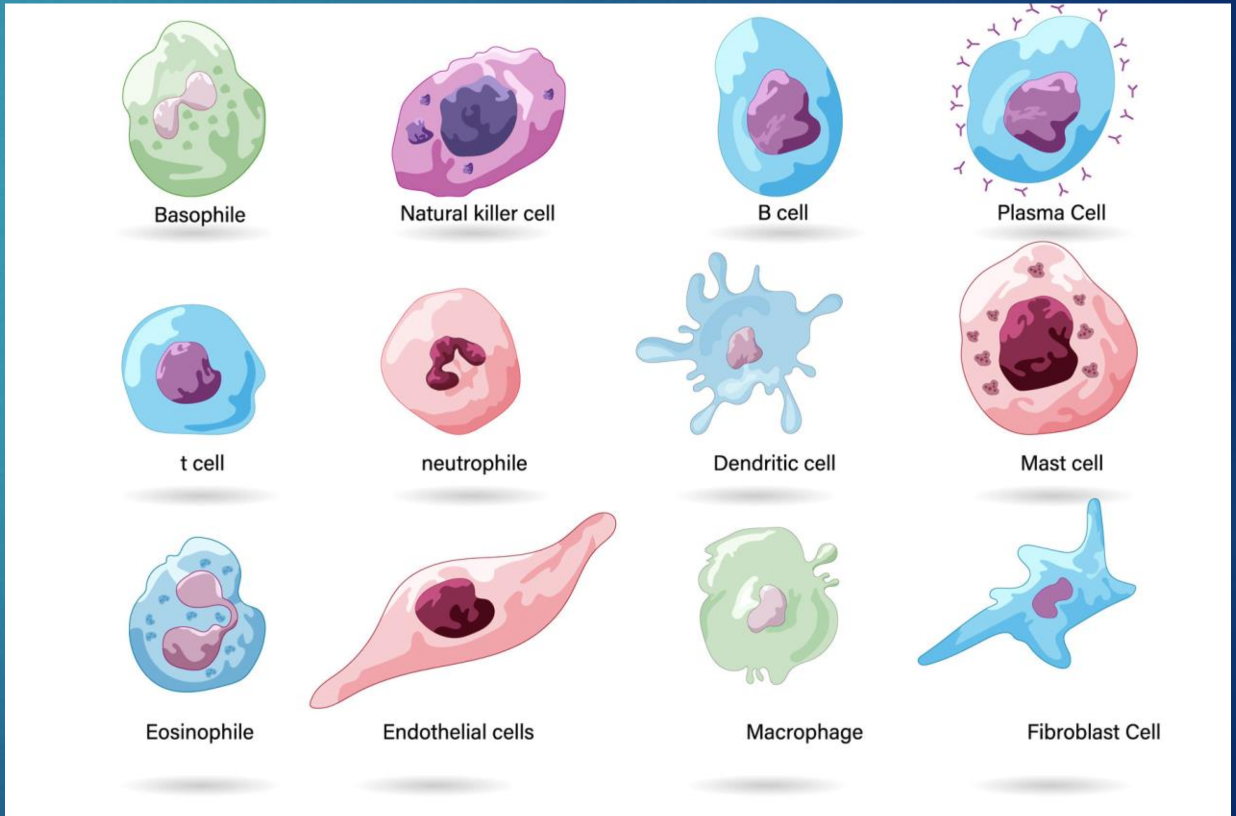
The task: scRNA-seq to cell type

- **Expression levels** $X_{i,j} = \log(1 + cR_{i,j}/R_{i,*})$ as input
- **Pseudo-labels** Y_i as output
- Some neural network as classifier
- There is no foundation model
- We investigate network architectures fit for scRNA-seq data

The dataset: 10k PBMC sample

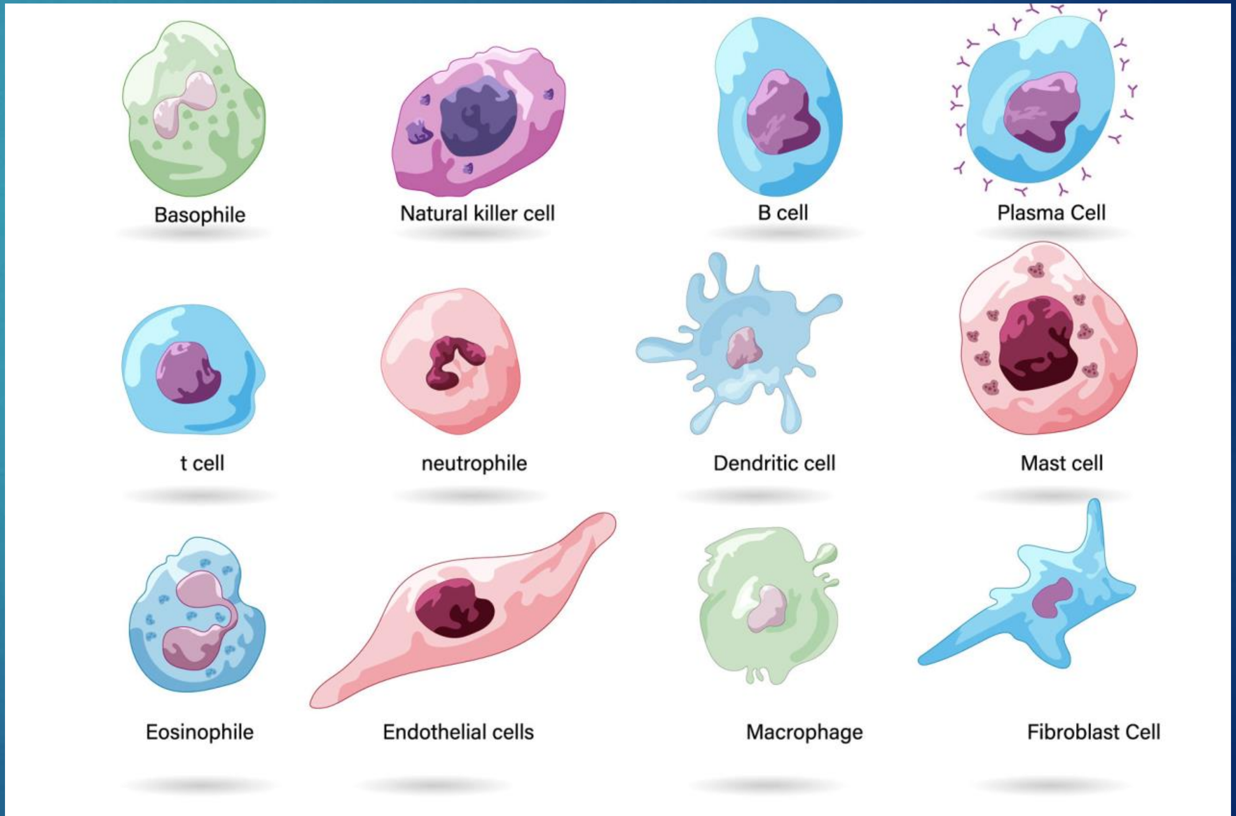
The dataset: 10k PBMC sample

- Peripheral Blood Mononuclear Cells (PBMC) [10x]



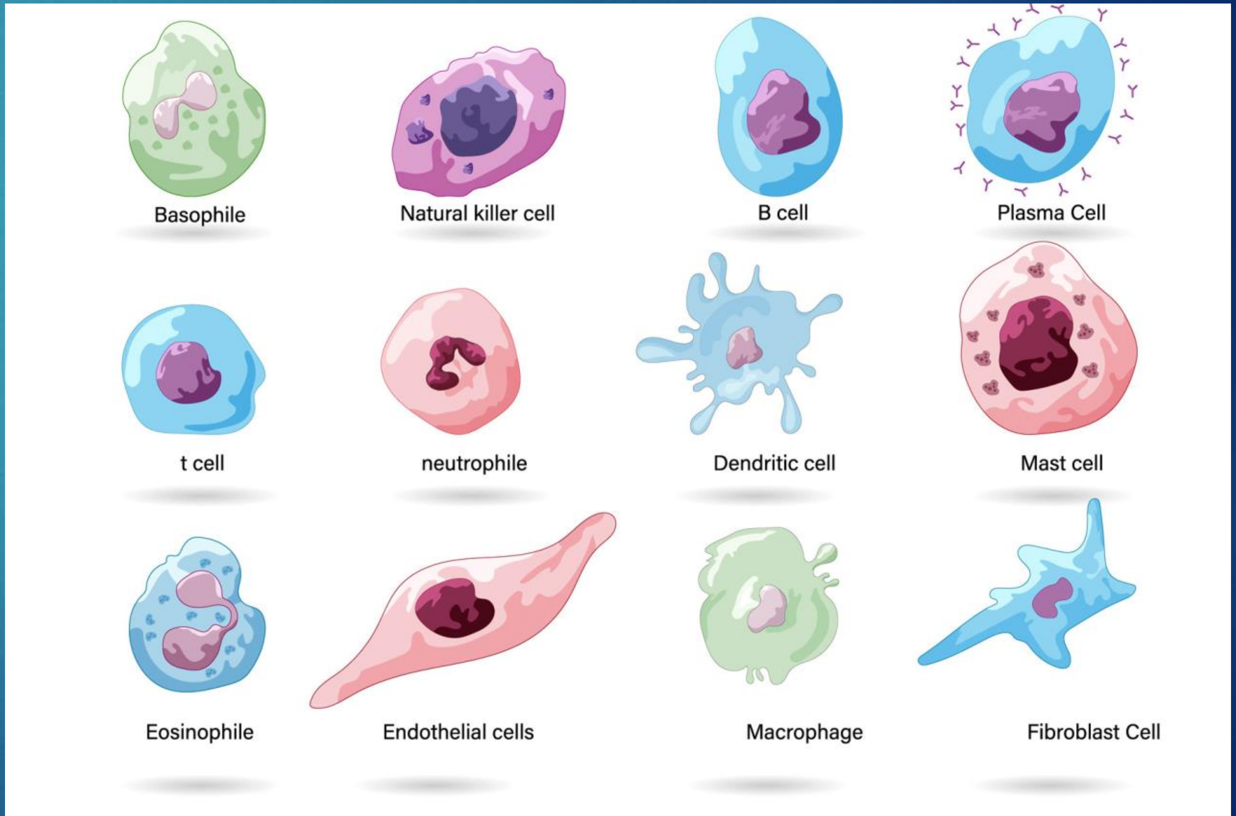
The dataset: 10k PBMC sample

- Peripheral Blood Mononuclear Cells (PBMC) [`10x`]
- $M = 10114$ cells



The dataset: 10k PBMC sample

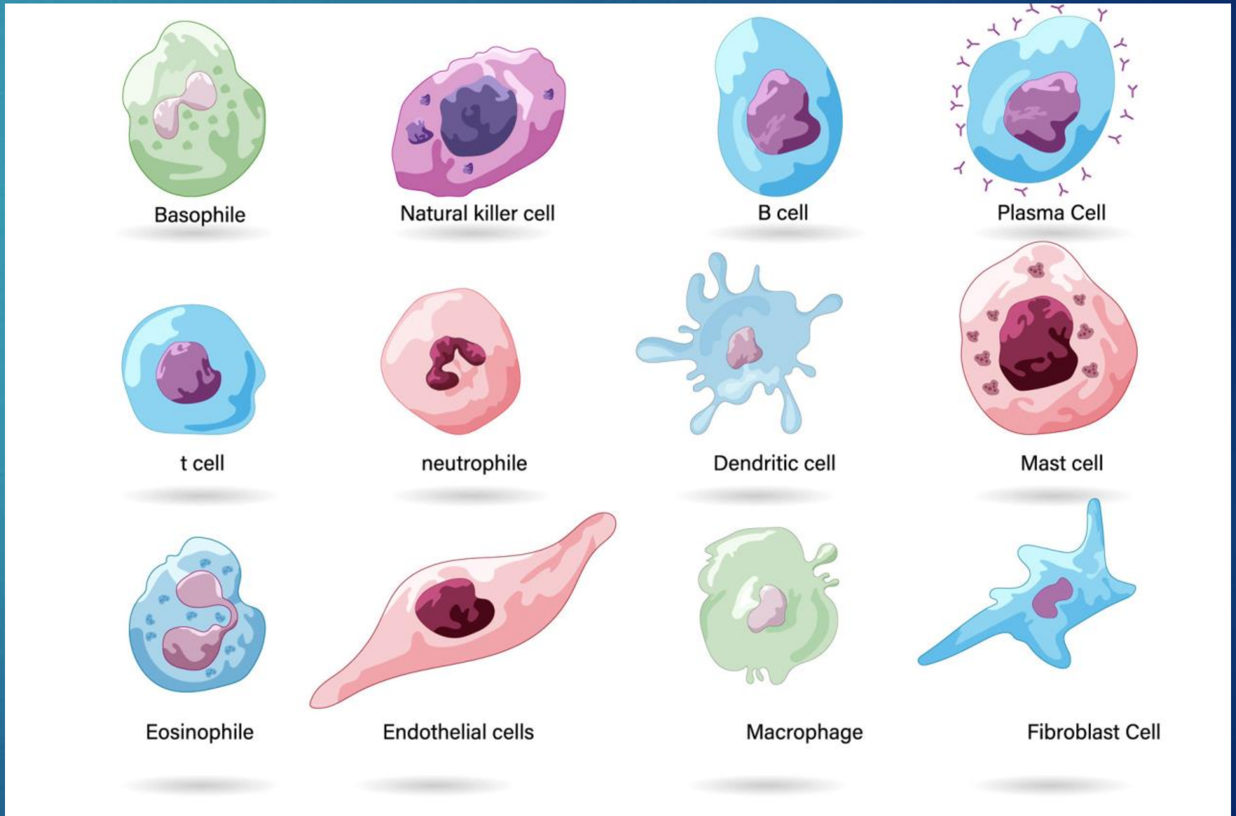
- Peripheral Blood Mononuclear Cells (PBMC) [10x]
- $M = 10114$ cells
- $N = 13054$ genes



The dataset: 10k PBMC sample

- Peripheral Blood Mononuclear Cells (PBMC) [10x]
- $M = 10114$ cells
- $N = 13054$ genes
- Sparse counts $R_{i,j}$

0	1	2	3+
89%	7.5%	1.5%	2%

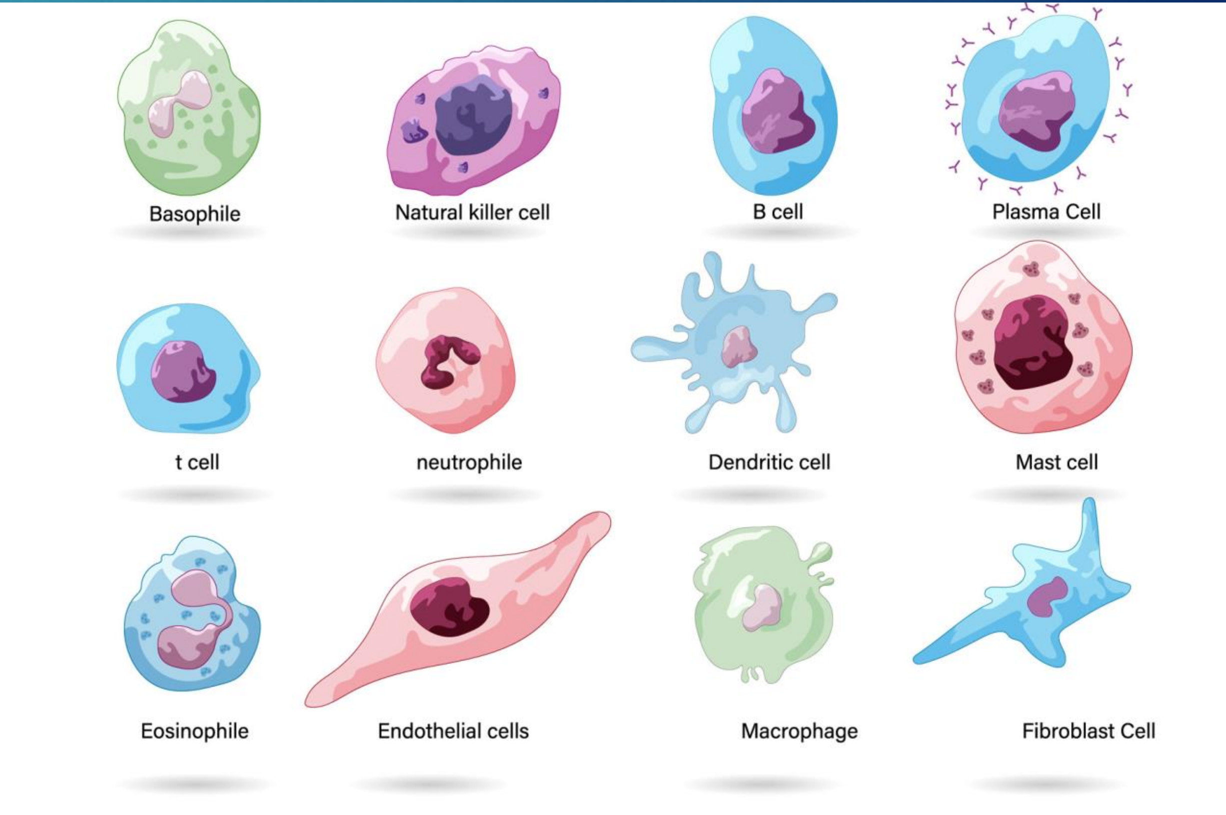


The dataset: 10k PBMC sample

- Peripheral Blood Mononuclear Cells (PBMC) [10x]
- $M = 10114$ cells
- $N = 13054$ genes
- Sparse counts $R_{i,j}$

0	1	2	3+
89%	7.5%	1.5%	2%

- 80% genes $R_{*,j} < 0.2$

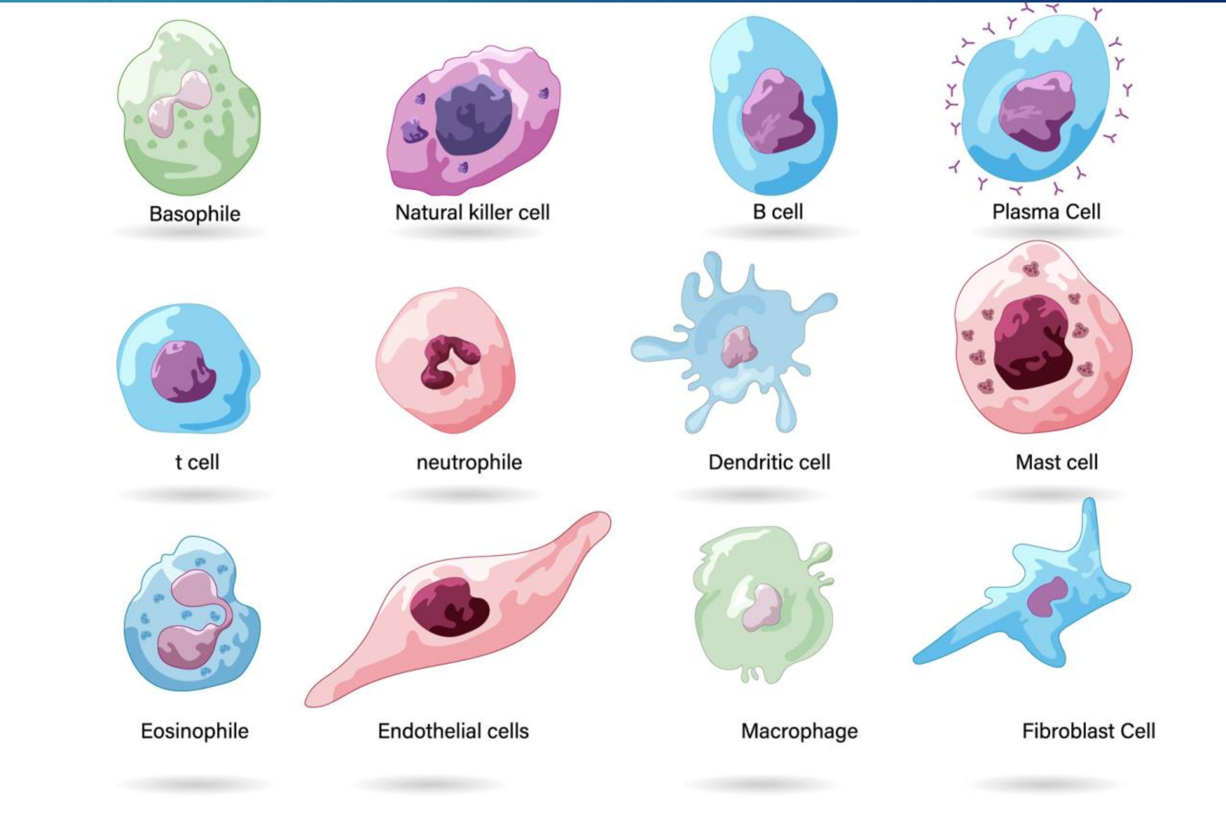


The dataset: 10k PBMC sample

- Peripheral Blood Mononuclear Cells (PBMC) [`10x`]
- $M = 10114$ cells
- $N = 13054$ genes
- Sparse counts $R_{i,j}$

0	1	2	3+
89%	7.5%	1.5%	2%

- 80% genes $R_{*,j} < 0.2$
- 15% genes $R_{*,j} < 0.01$

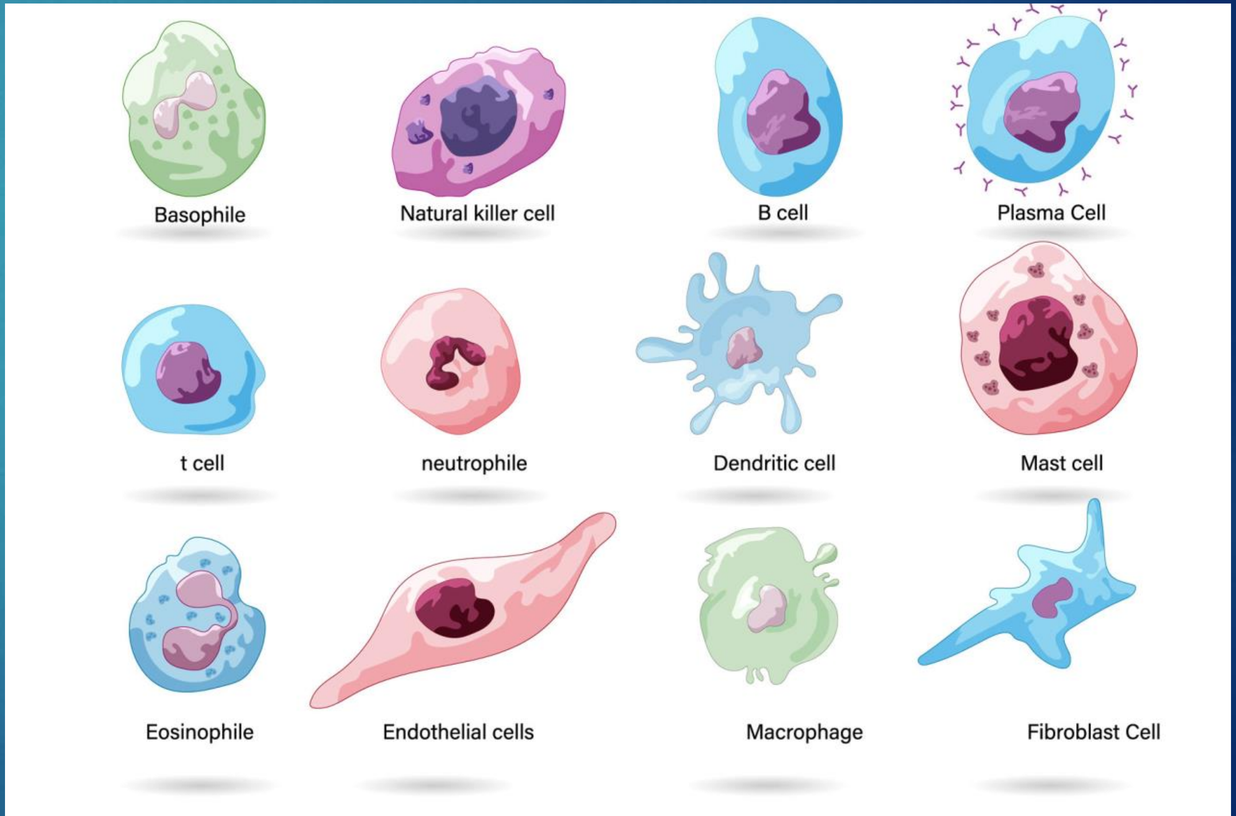


The dataset: 10k PBMC sample

- Peripheral Blood Mononuclear Cells (PBMC) [10x]
- $M = 10114$ cells
- $N = 13054$ genes
- Sparse counts $R_{i,j}$

0	1	2	3+
89%	7.5%	1.5%	2%

- 80% genes $R_{*,j} < 0.2$
- 15% genes $R_{*,j} < 0.01$
- 5% genes have counts only 0 and 1



The dataset: 10k PBMC sample

	TUBB2A	ZNF217	SNHG7	STK19	KIAA1324	RNF41	RASA3	ELP1	THRA	LINC01431	label
AAACCTGAGAGTGAGA	0	0	0	0	0	0	1	0	0	0	82
AAACCTGAGGCATTGG	0	0	0	1	0	0	0	0	0	0	19
AAACCTGCACCAGGTC	0	0	0	0	0	0	0	0	0	0	28
AAACCTGCAGGGATTG	0	0	0	0	0	0	2	0	0	0	58
AAACCTGCAGTCAGAG	0	0	0	0	0	0	1	0	0	0	66
...	...	...	...	...	...	...	...	...	...	...	...
TTTGTCAGTTGATTCG	0	0	0	0	0	0	0	0	0	0	37
TTTGTCATCATAACCG	0	0	0	0	0	0	1	0	0	0	63
TTTGTCATCATCATTC	0	0	0	0	0	0	0	0	0	0	34
TTTGTCATCATGCTCC	0	0	3	0	1	1	1	0	0	0	65
TTTGTCATCCACGTTC	0	0	0	0	0	0	0	0	0	0	63

10137 rows × 13055 columns

The dataset: 10k PBMC sample

	TUBB2A	ZNF217	SNHG7	STK19	KIAA1324	RNF41	RASA3	ELP1	THRA	LINC01431	label
AAACCTGAGAGTGAGA	0.0	0.0	0.000	0.000	0.00	0.000	1.265	0.0	0.0	0.0	82
AAACCTGAGGCATTGG	0.0	0.0	0.000	2.678	0.00	0.000	0.000	0.0	0.0	0.0	19
AAACCTGCACCAGGTC	0.0	0.0	0.000	0.000	0.00	0.000	0.000	0.0	0.0	0.0	28
AAACCTGCAGGGATTG	0.0	0.0	0.000	0.000	0.00	0.000	1.806	0.0	0.0	0.0	58
AAACCTGCAGTCAGAG	0.0	0.0	0.000	0.000	0.00	0.000	1.265	0.0	0.0	0.0	66
...	...	...	...	...	...	...	...	...	...	...	...
TTTGTCAGTTGATTCG	0.0	0.0	0.000	0.000	0.00	0.000	0.000	0.0	0.0	0.0	37
TTTGTCATCATAACCG	0.0	0.0	0.000	0.000	0.00	0.000	1.265	0.0	0.0	0.0	63
TTTGTCATCATCATTC	0.0	0.0	0.000	0.000	0.00	0.000	0.000	0.0	0.0	0.0	34
TTTGTCATCATGCTCC	0.0	0.0	2.408	0.000	3.66	2.434	1.265	0.0	0.0	0.0	65
TTTGTCATCCACGTTC	0.0	0.0	0.000	0.000	0.00	0.000	0.000	0.0	0.0	0.0	63

10137 rows × 13055 columns

The dataset: 10k PBMC sample

- $C = 86$ **classes** (pseudo-labels)

The dataset: 10k PBMC sample

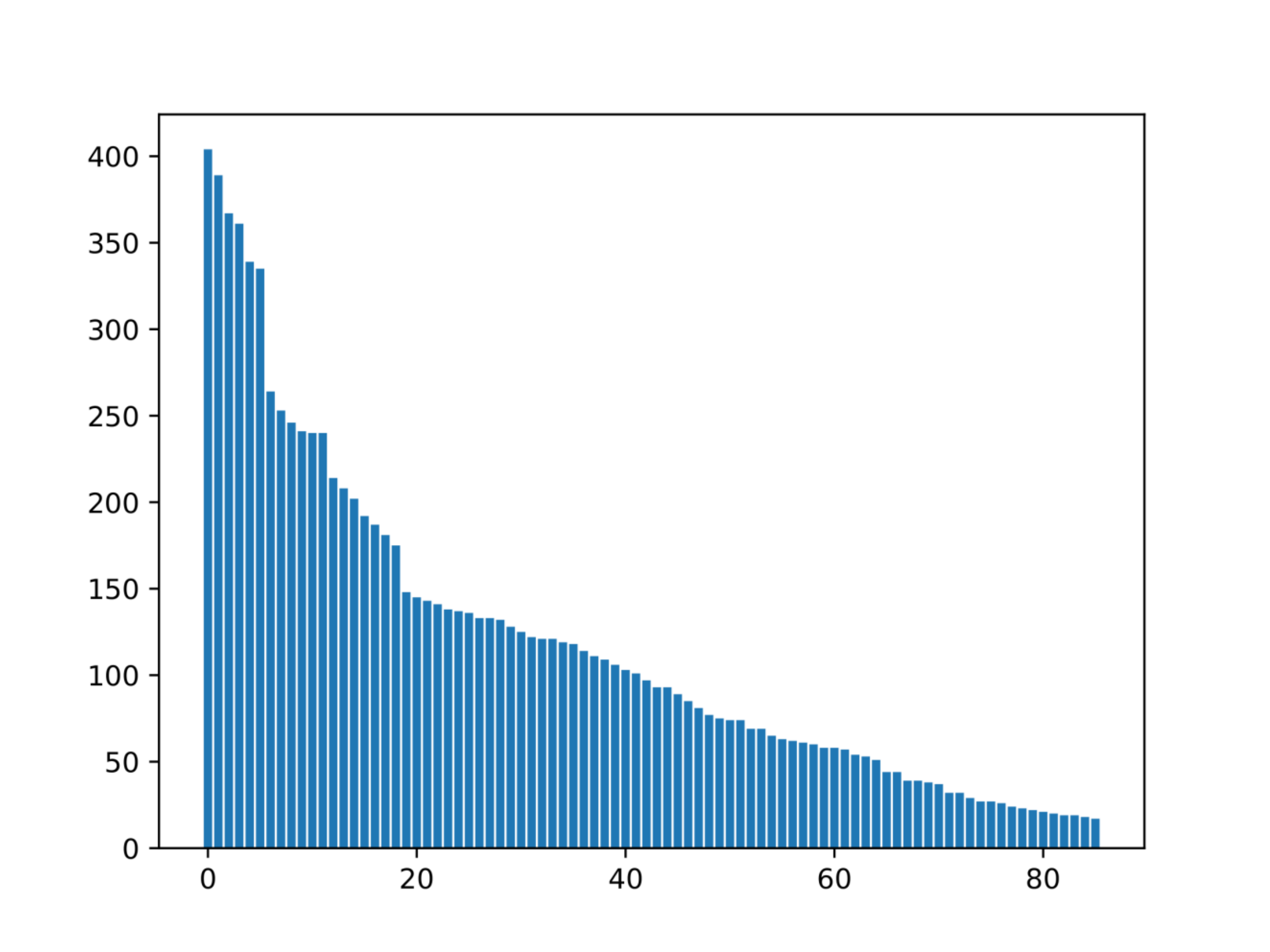
- $C = 86$ **classes** (pseudo-labels)
- statistically uniform [COTAN]

The dataset: 10k PBMC sample

- $C = 86$ **classes** (pseudo-labels)
- statistically uniform [COTAN]
- no biological information

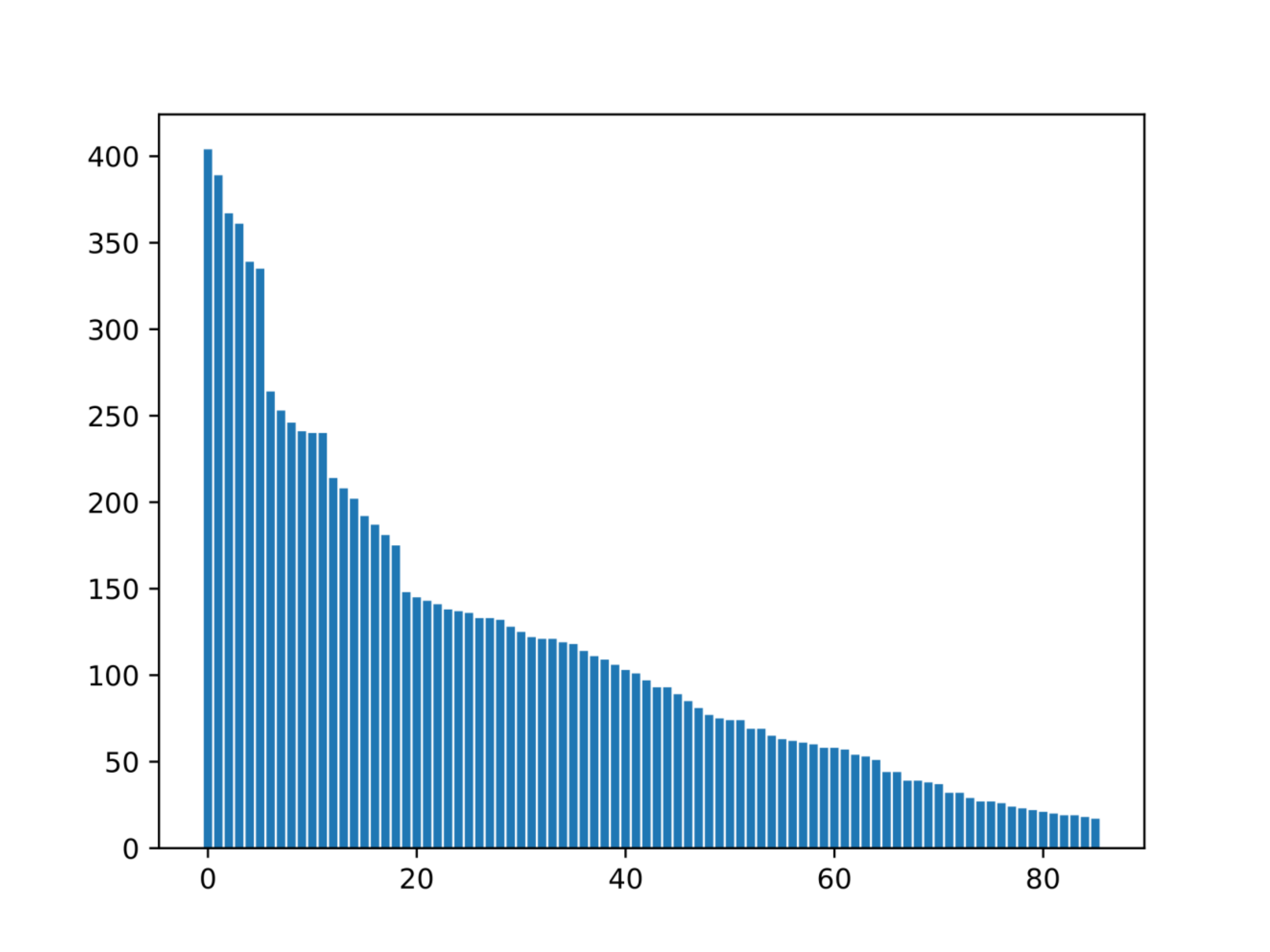
The dataset: 10k PBMC sample

- $C = 86$ **classes** (pseudo-labels)
- statistically uniform [COTAN]
- no biological information
- smallest has 17 cells, largest 404



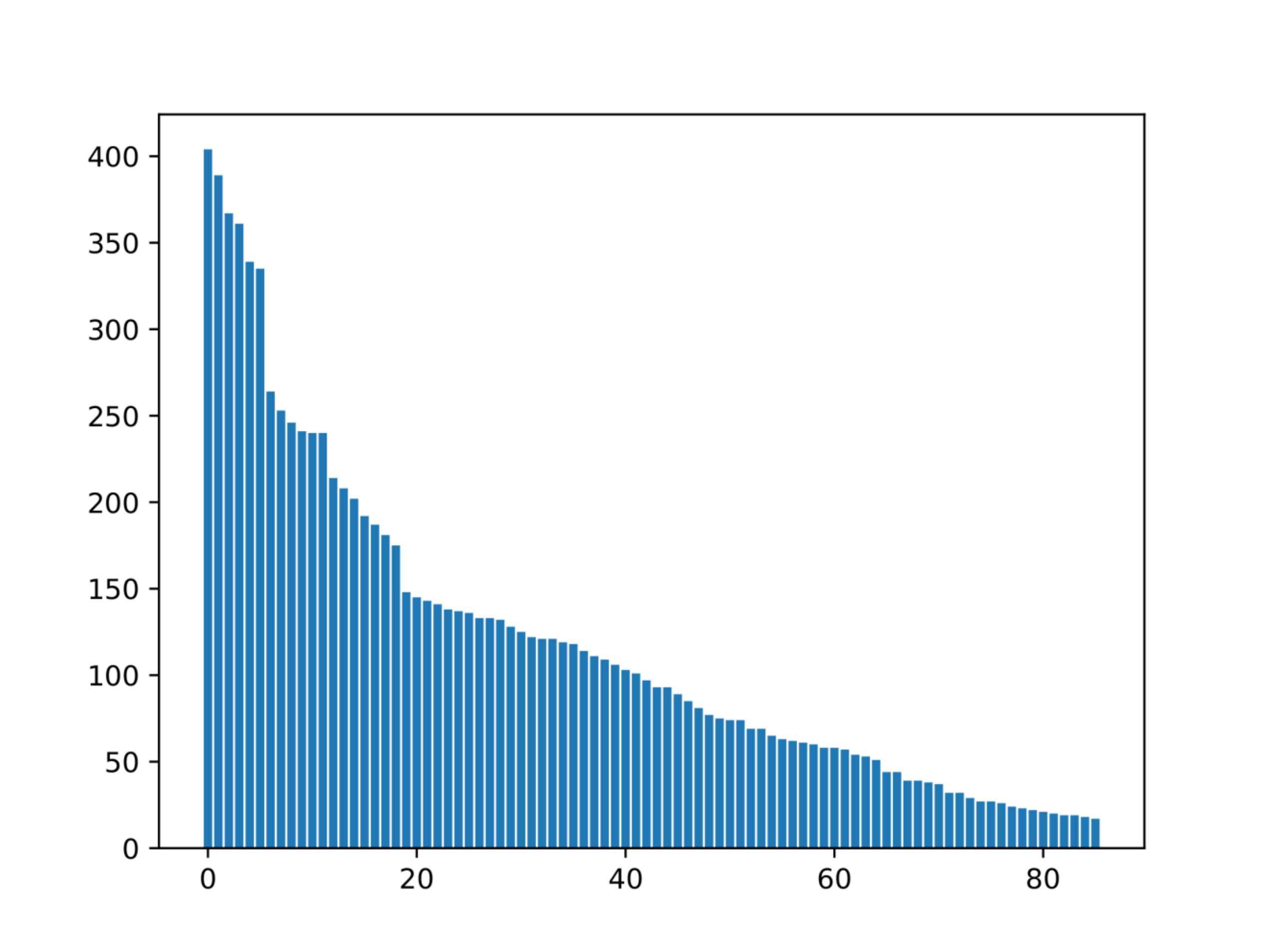
The dataset: 10k PBMC sample

- $C = 86$ **classes** (pseudo-labels)
- statistically uniform [COTAN]
- no biological information
- smallest has 17 cells, largest 404
- genes **sorted by variability**



The dataset: 10k PBMC sample

- $C = 86$ **classes** (pseudo-labels)
- statistically uniform [COTAN]
- no biological information
- smallest has 17 cells, largest 404
- genes **sorted by variability**
- only the first 2000 used in the typical pipeline



Vanilla models

Vanilla models

- n highest variability genes are used as input

Vanilla models

- n highest variability genes are used as input
- n from 100 to $N = 13054$

Vanilla models

- n highest variability genes are used as input
- n from 100 to $N = 13054$
- AdamW optimizer

Vanilla models

- n highest variability genes are used as input
- n from 100 to $N = 13054$
- AdamW optimizer
- Strong L^2 regularization, standard learning rate

Vanilla models

- n highest variability genes are used as input
- n from 100 to $N = 13054$
- AdamW optimizer
- Strong L² regularization, standard learning rate
- 3 replicates per experiment

Vanilla models

- n highest variability genes are used as input
- n from 100 to $N = 13054$
- AdamW optimizer
- Strong L^2 regularization, standard learning rate
- 3 replicates per experiment
- 10k cells split in 70% training, 15% validation, 15% test (stratified for labels)

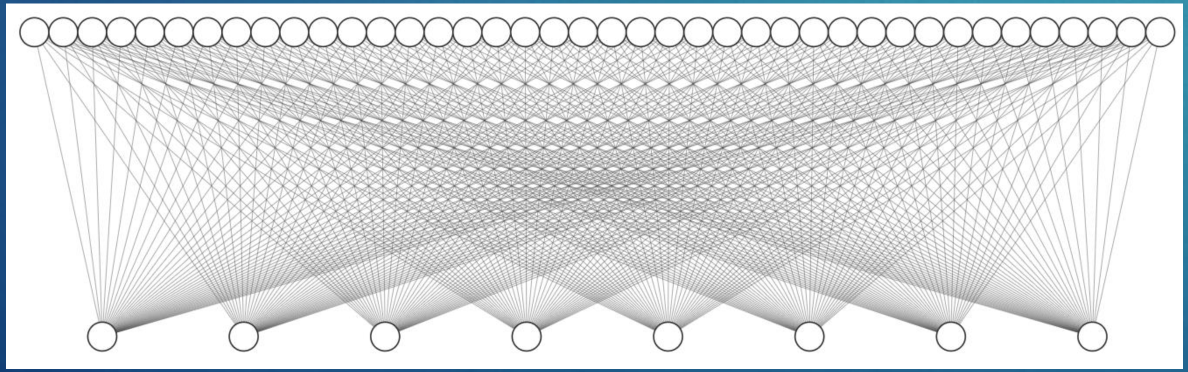
Vanilla models

- n highest variability genes are used as input
- n from 100 to $N = 13054$
- AdamW optimizer
- Strong L² regularization, standard learning rate
- 3 replicates per experiment
- 10k cells split in 70% training, 15% validation, 15% test (stratified for labels)
- Score is **test accuracy (TA)** for the model with best validation accuracy (checked at epoch end)

Vanilla models
Logistic Regression (LR)

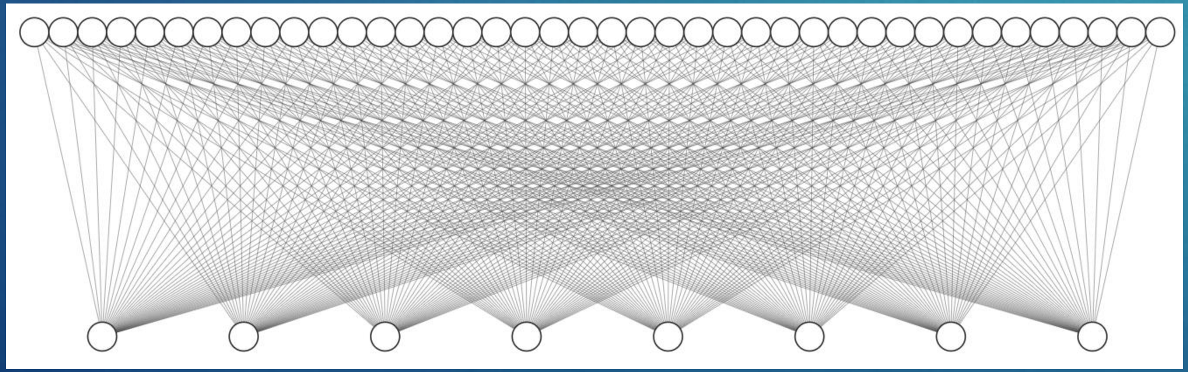
Vanilla models
Logistic Regression (LR)

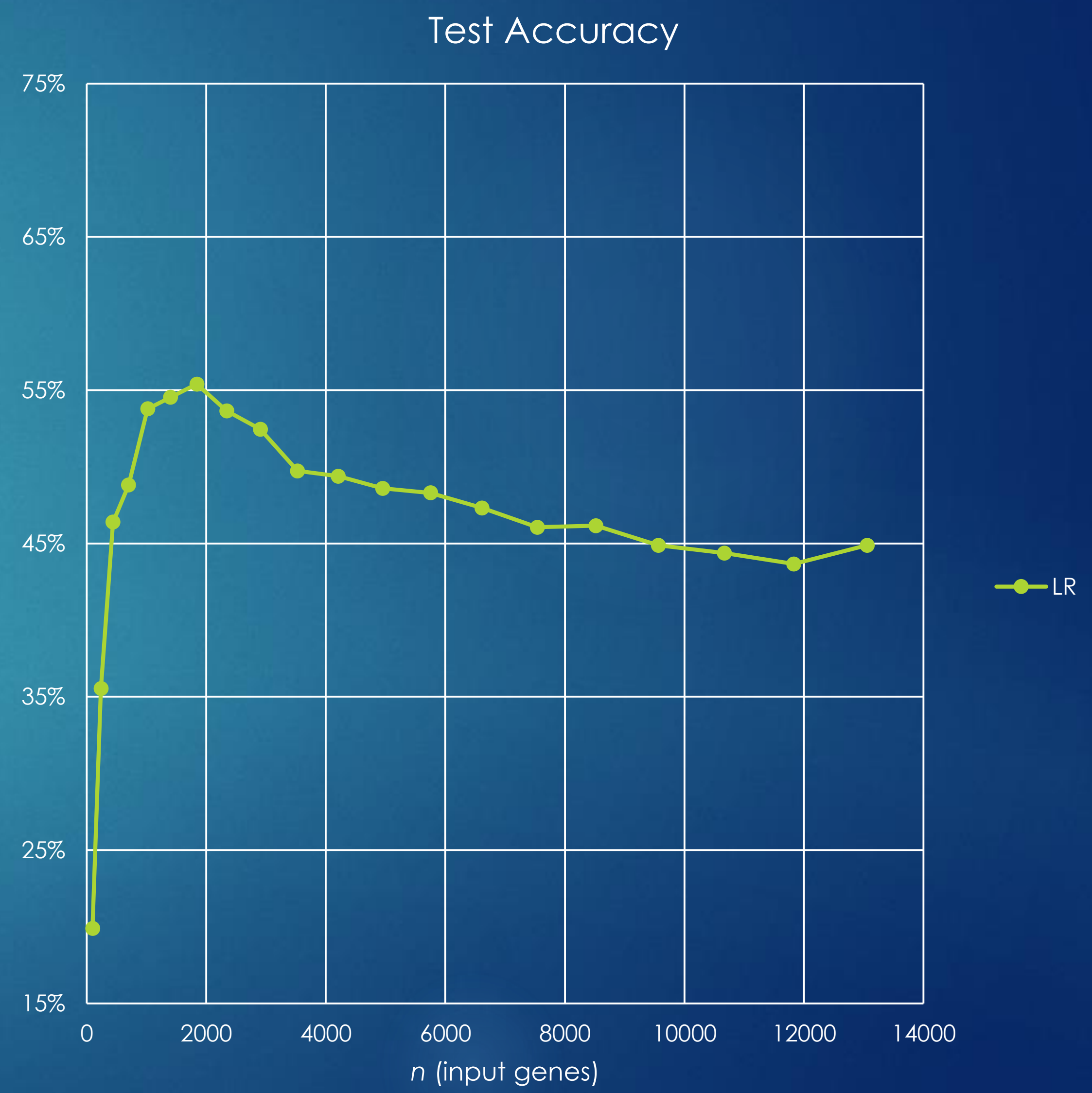
- $86n$ parameters, from 9k to 1.1M



Vanilla models
Logistic Regression (LR)

- $86n$ parameters, from 9k to 1.1M
- top TA 55.4% at $n = 1800$

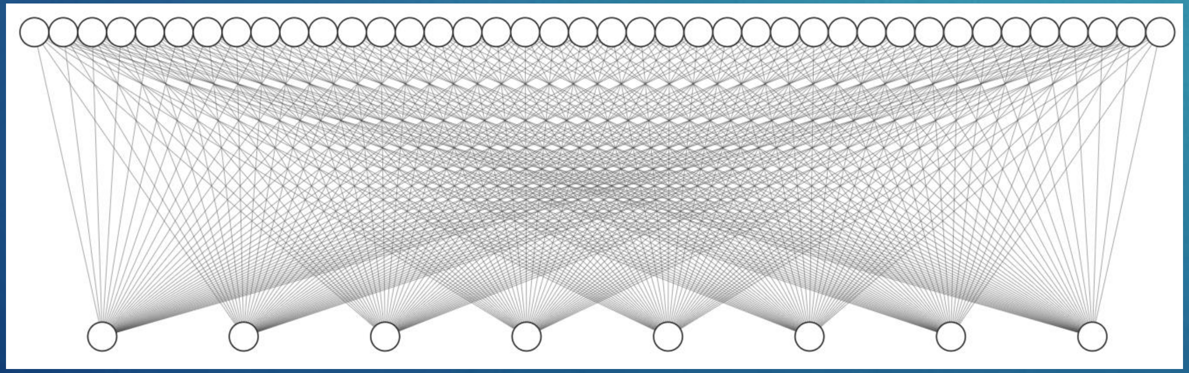


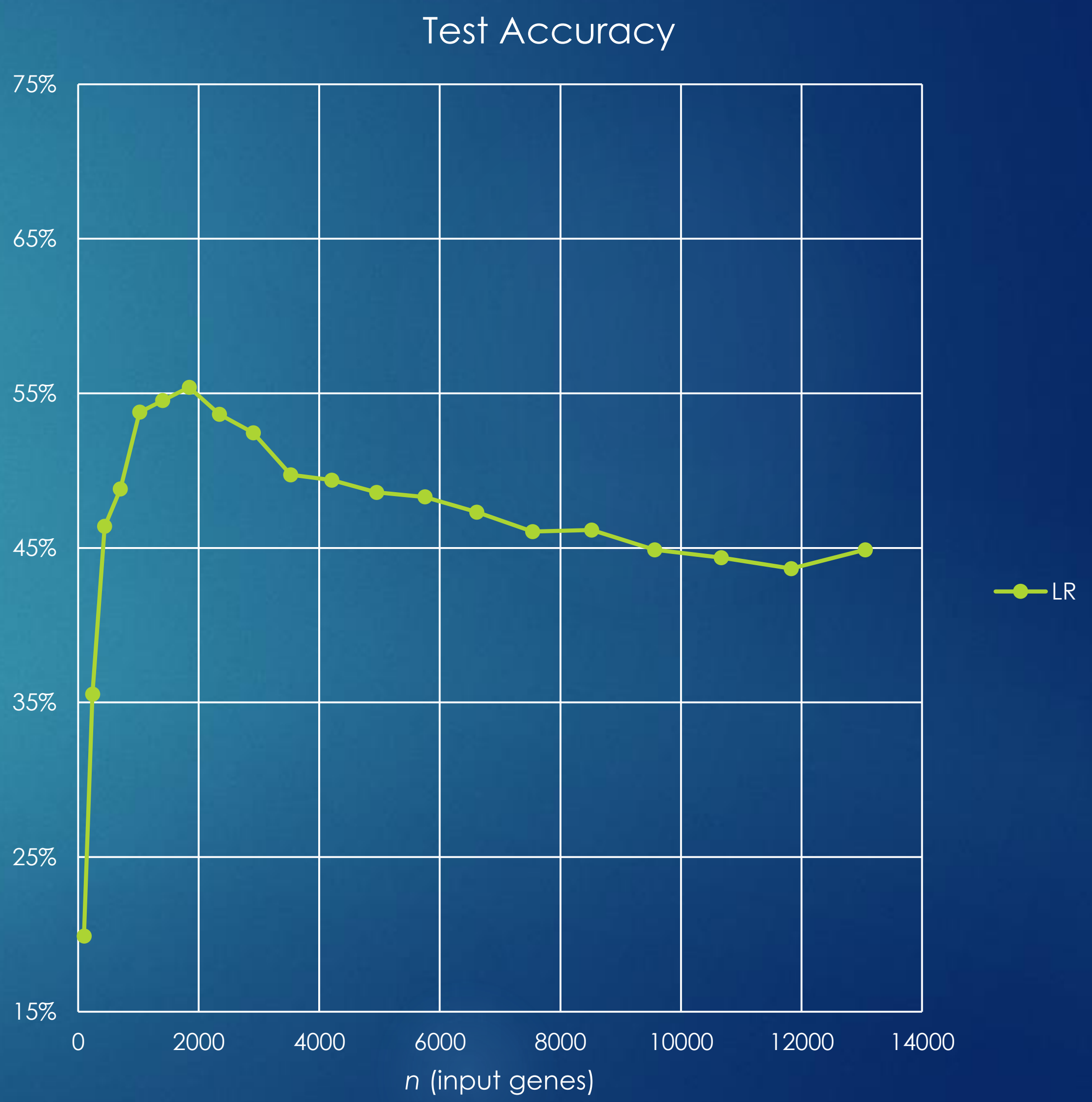


Vanilla models

Logistic Regression (LR)

- $86n$ parameters, from 9k to 1.1M
- top TA 55.4% at $n = 1800$
- for large n, TA decreases from 49% to 44%



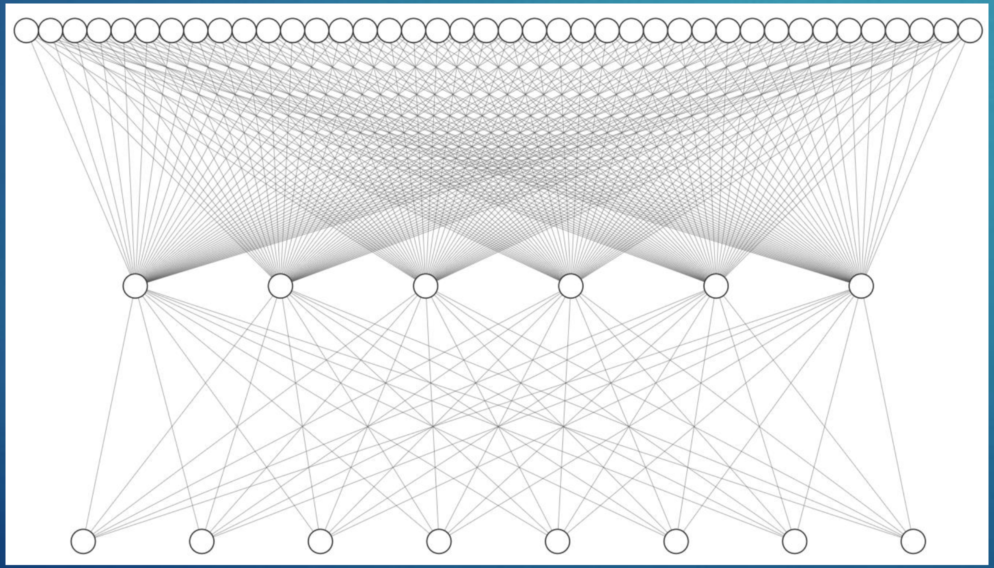


Vanilla models

Fully connected NN (MLP)

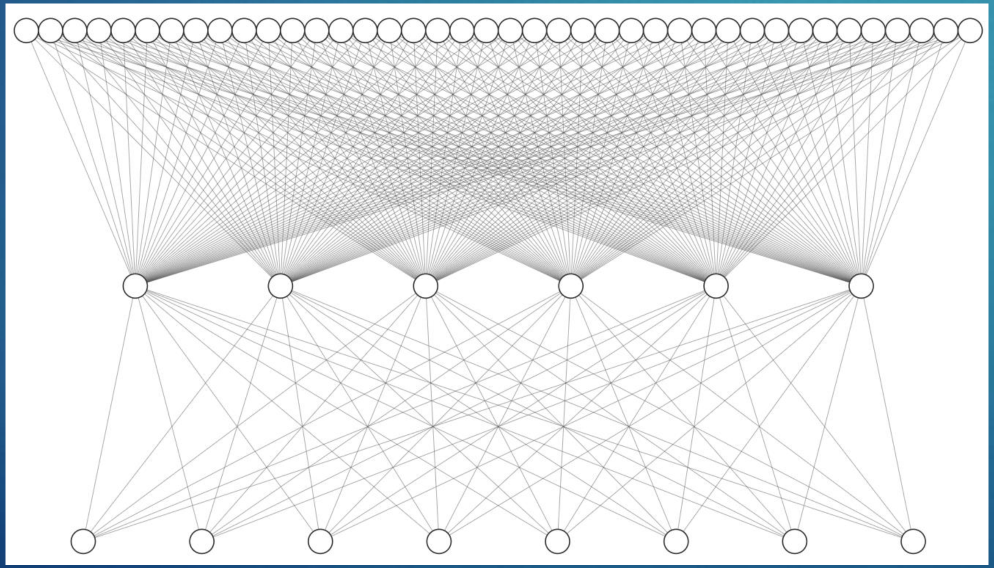
Vanilla models
Fully connected NN (MLP)

- 1 hidden layer with 64 units



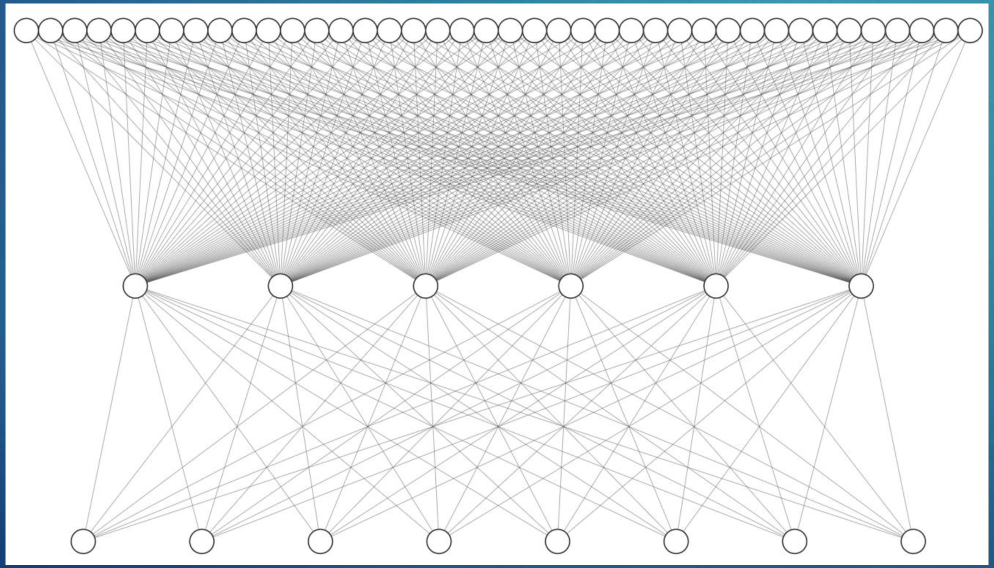
Vanilla models
Fully connected NN (MLP)

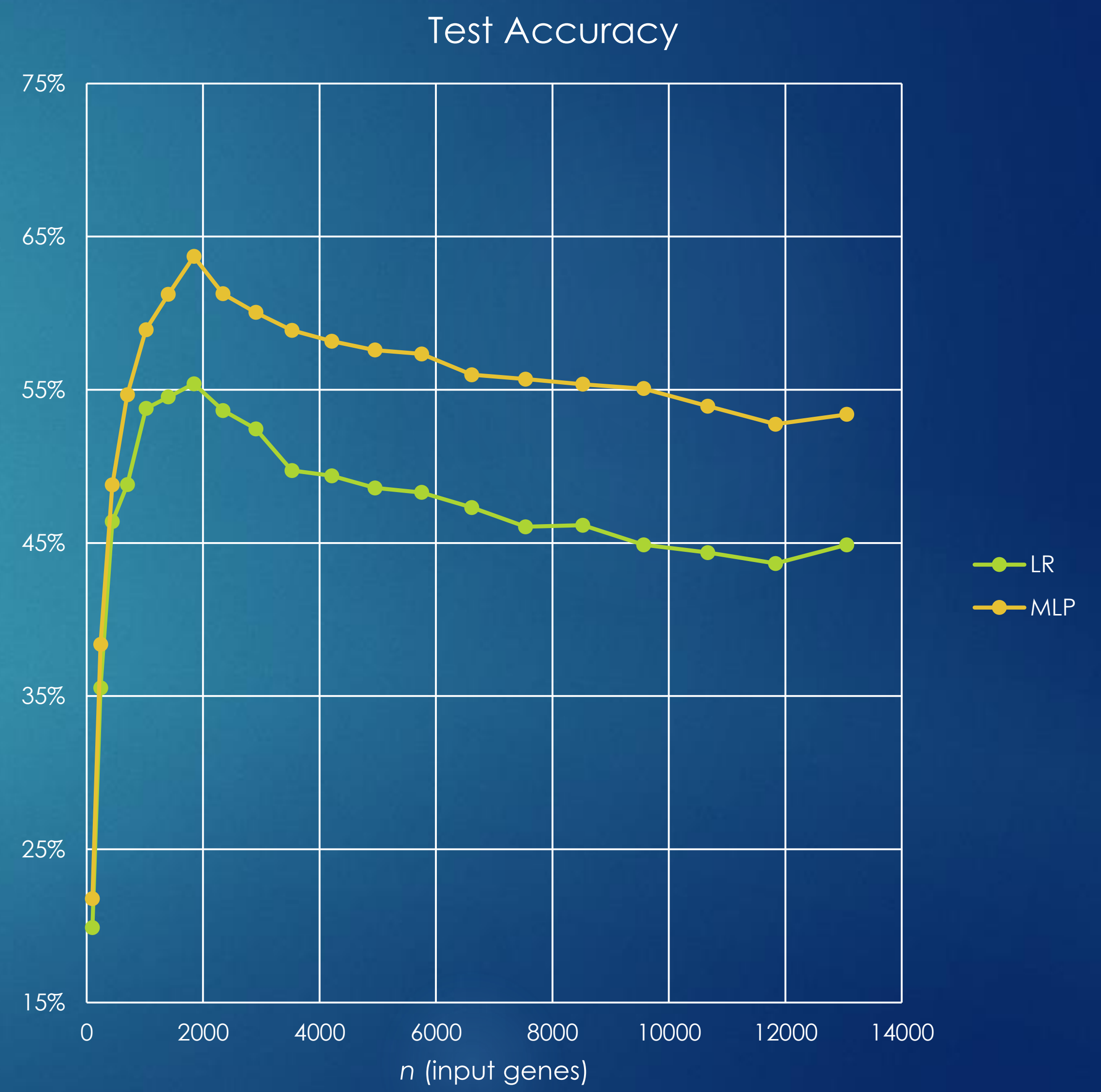
- 1 hidden layer with 64 units
- $64(n+86)$ parameters, from 12k to 0.84M



Vanilla models
Fully connected NN (MLP)

- 1 hidden layer with 64 units
- $64(n+86)$ parameters, from 12k to 0.84M
- top TA 63.7% at $n = 1800$

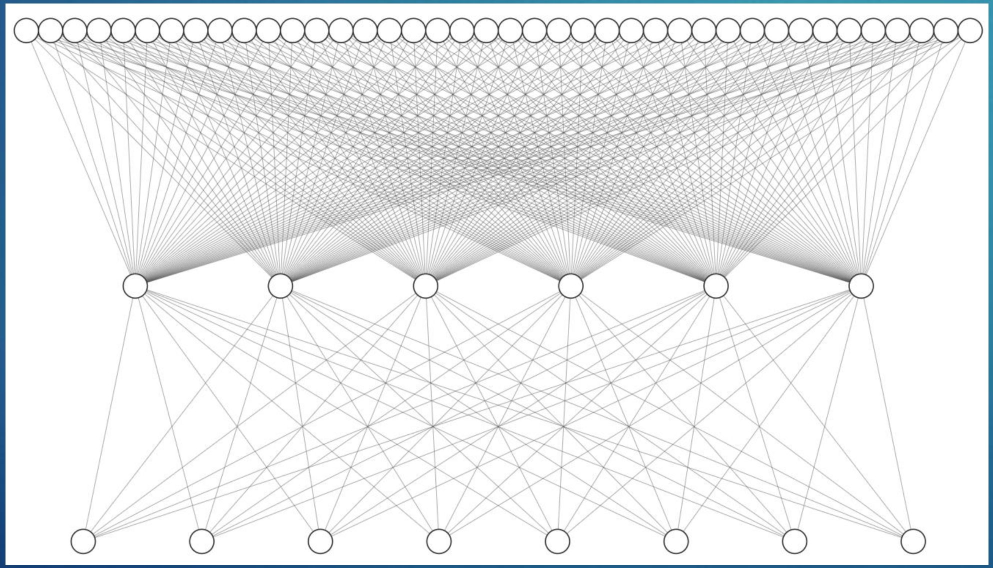


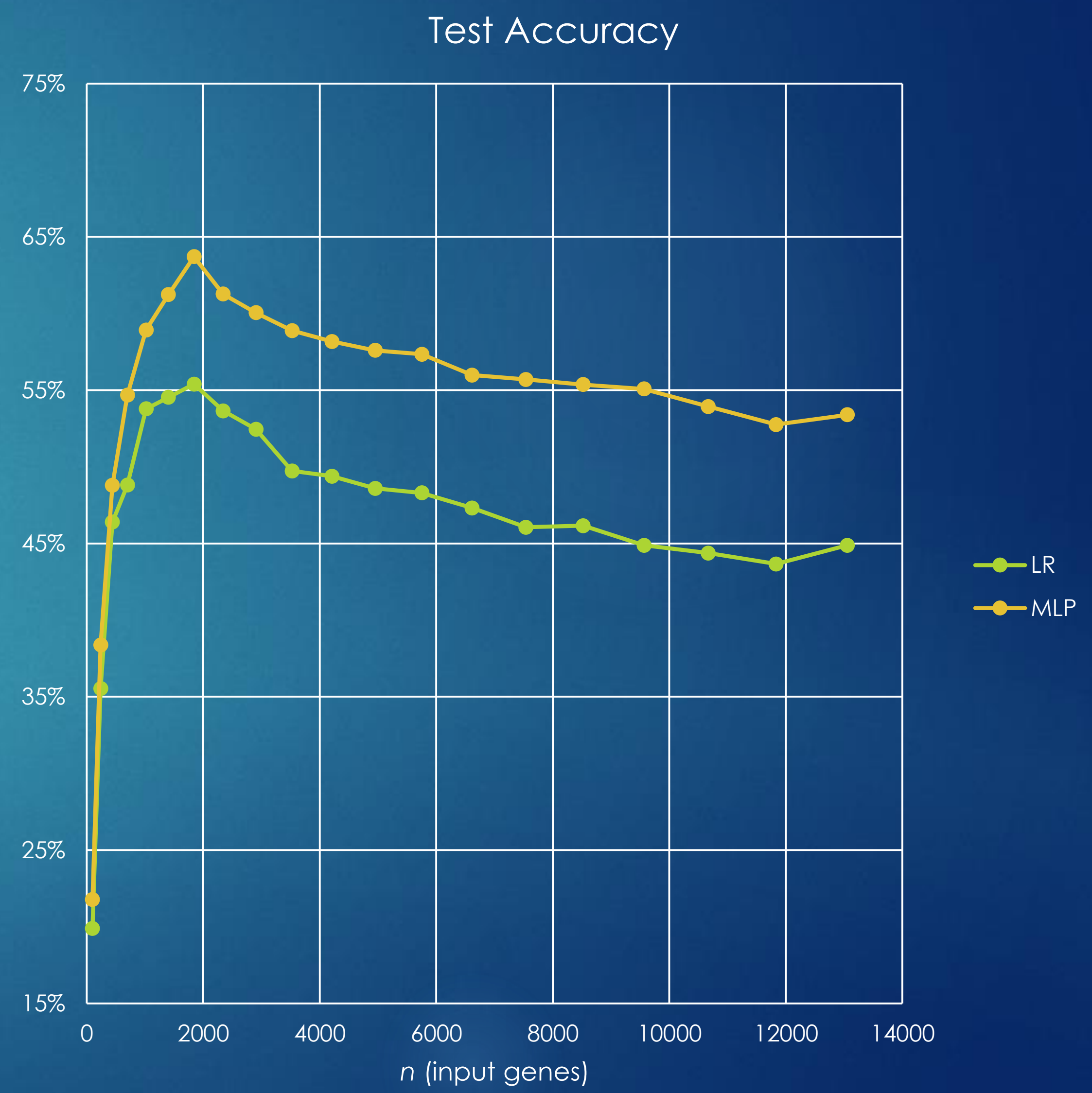


Vanilla models

Fully connected NN (MLP)

- 1 hidden layer with 64 units
- $64(n+86)$ parameters, from 12k to 0.84M
- top TA 63.7% at $n = 1800$
- for large n, TA decreases from 57% to 52%



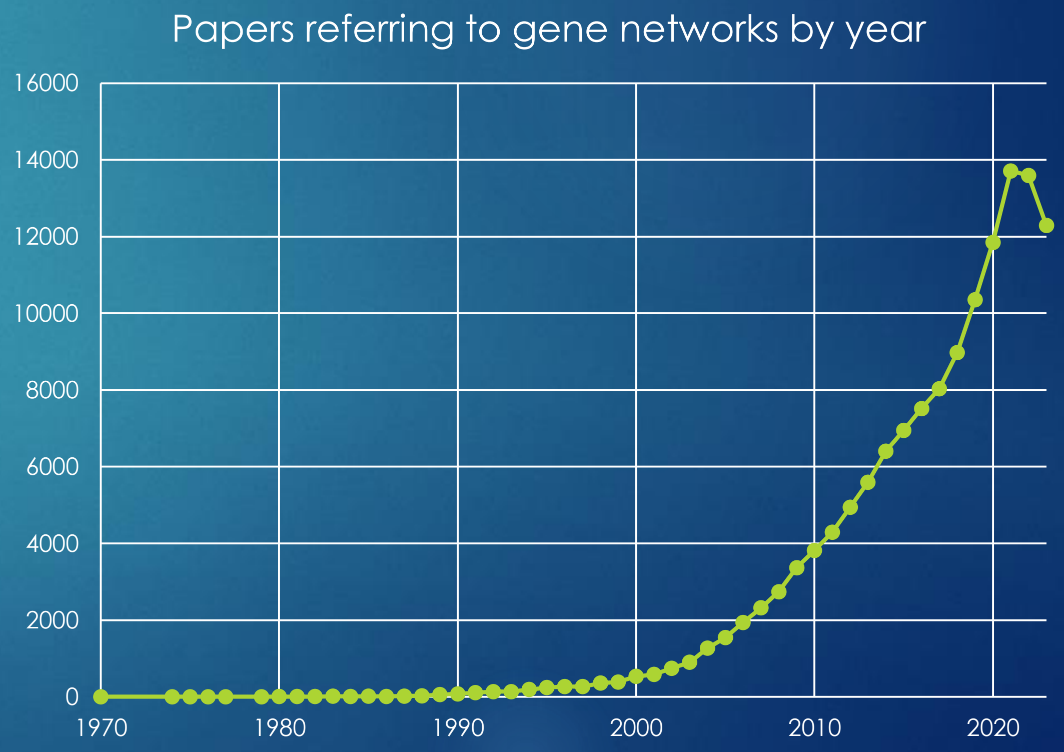


Graphs of genes and neural networks

Graphs of genes

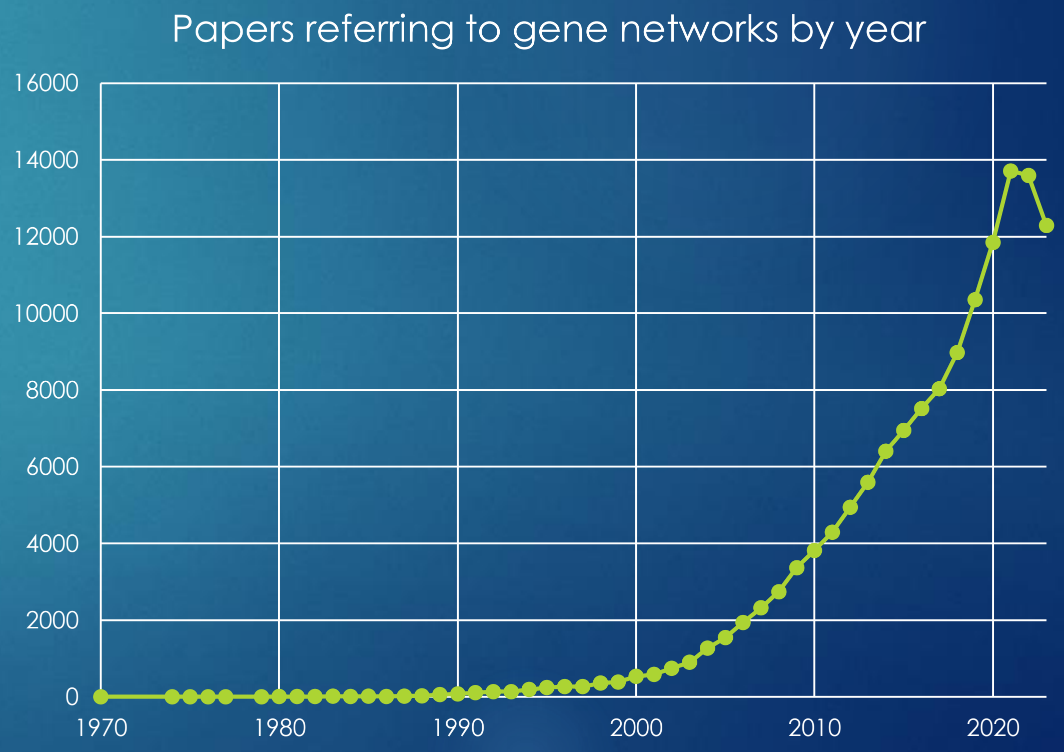
Graphs of genes

- Gene networks are graphs of genes (10k+ papers per year)



Graphs of genes

- Gene networks are graphs of genes (10k+ papers per year)
- Connected by functionality or co-expression in cell types



Graphs of genes

- Gene networks are graphs of genes (10k+ papers per year)
- Connected by functionality or co-expression in cell types
- Common to start from a correlation matrix

	TUBB2A	ZNF217	SNHG7	STK19	KIAA1324	RNF41	RASA3	ELP1	THRA	LINC01431
TUBB2A	0.00000	0.00089	-0.00036	0.00866	0.01848	0.00160	-0.00428	0.00304	0.00128	-0.00438
ZNF217	0.00089	0.00000	-0.00257	0.02519	0.00926	0.01145	0.01328	0.00838	0.01100	0.00054
SNHG7	-0.00036	-0.00257	0.00000	-0.01815	-0.00393	0.00915	-0.02798	-0.01806	-0.02084	0.00027
STK19	0.00866	0.02519	-0.01815	0.00000	-0.01672	0.00245	0.00713	0.00877	0.00178	-0.00003
KIAA1324	0.01848	0.00926	-0.00393	-0.01672	0.00000	-0.00223	0.00609	0.00391	-0.01231	-0.00165
RNF41	0.00160	0.01145	0.00915	0.00245	-0.00223	0.00000	-0.02768	-0.00850	-0.01359	0.01427
RASA3	-0.00428	0.01328	-0.02798	0.00713	0.00609	-0.02768	0.00000	-0.00533	0.01680	-0.00589
ELP1	0.00304	0.00838	-0.01806	0.00877	0.00391	-0.00850	-0.00533	0.00000	-0.01805	-0.00478
THRA	0.00128	0.01100	-0.02084	0.00178	-0.01231	-0.01359	0.01680	-0.01805	0.00000	-0.00994
LINC01431	-0.00438	0.00054	0.00027	-0.00003	-0.00165	0.01427	-0.00589	-0.00478	-0.00994	0.00000

Graphs of genes

- Gene networks are graphs of genes (10k+ papers per year)
- Connected by functionality or co-expression in cell types
- Common to start from a correlation matrix
- Get an adjacency matrix by hard threshold

	TUBB2A	ZNF217	SNHG7	STK19	KIAA1324	RNF41	RASA3	ELP1	THRA	LINC01431
TUBB2A	0.00000	0.00089	-0.00036	0.00866	0.01848	0.00160	-0.00428	0.00304	0.00128	-0.00438
ZNF217	0.00089	0.00000	-0.00257	0.02519	0.00926	0.01145	0.01328	0.00838	0.01100	0.00054
SNHG7	-0.00036	-0.00257	0.00000	-0.01815	-0.00393	0.00915	-0.02798	-0.01806	-0.02084	0.00027
STK19	0.00866	0.02519	-0.01815	0.00000	-0.01672	0.00245	0.00713	0.00877	0.00178	-0.00003
KIAA1324	0.01848	0.00926	-0.00393	-0.01672	0.00000	-0.00223	0.00609	0.00391	-0.01231	-0.00165
RNF41	0.00160	0.01145	0.00915	0.00245	-0.00223	0.00000	-0.02768	-0.00850	-0.01359	0.01427
RASA3	-0.00428	0.01328	-0.02798	0.00713	0.00609	-0.02768	0.00000	-0.00533	0.01680	-0.00589
ELP1	0.00304	0.00838	-0.01806	0.00877	0.00391	-0.00850	-0.00533	0.00000	-0.01805	-0.00478
THRA	0.00128	0.01100	-0.02084	0.00178	-0.01231	-0.01359	0.01680	-0.01805	0.00000	-0.00994
LINC01431	-0.00438	0.00054	0.00027	-0.00003	-0.00165	0.01427	-0.00589	-0.00478	-0.00994	0.00000

Graphs of genes

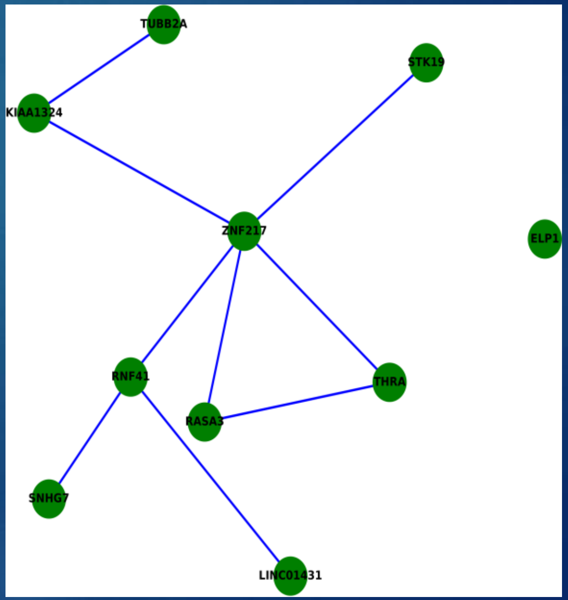
- Gene networks are graphs of genes (10k+ papers per year)
- Connected by functionality or co-expression in cell types
- Common to start from a correlation matrix
- Get an adjacency matrix by hard threshold

	TUBB2A	ZNF217	SNHG7	STK19	KIAA1324	RNF41	RASA3	ELP1	THRA	LINC01431
TUBB2A	0	0	0	0	1	0	0	0	0	0
ZNF217	0	0	0	1	1	1	1	0	1	0
SNHG7	0	0	0	0	0	1	0	0	0	0
STK19	0	1	0	0	0	0	0	0	0	0
KIAA1324	1	1	0	0	0	0	0	0	0	0
RNF41	0	1	1	0	0	0	0	0	0	1
RASA3	0	1	0	0	0	0	0	0	1	0
ELP1	0	0	0	0	0	0	0	0	0	0
THRA	0	1	0	0	0	0	1	0	0	0
LINC01431	0	0	0	0	0	1	0	0	0	0

Graphs of genes

- Gene networks are graphs of genes (10k+ papers per year)
- Connected by functionality or co-expression in cell types
- Common to start from a correlation matrix
- Get an adjacency matrix by hard threshold

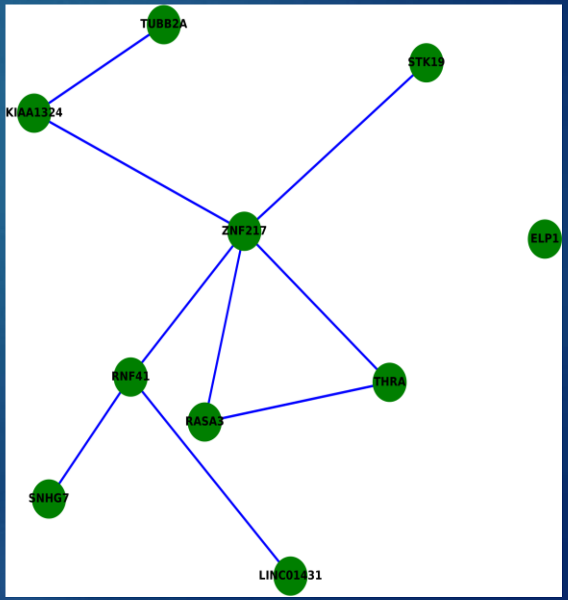
	TUBB2A	ZNF217	SNHG7	STK19	KIAA1324	RNF41	RASA3	ELP1	THRA	LINC01431
TUBB2A	0	0	0	0	1	0	0	0	0	0
ZNF217	0	0	0	1	1	1	1	0	1	0
SNHG7	0	0	0	0	0	1	0	0	0	0
STK19	0	1	0	0	0	0	0	0	0	0
KIAA1324	1	1	0	0	0	0	0	0	0	0
RNF41	0	1	1	0	0	0	0	0	0	1
RASA3	0	1	0	0	0	0	0	0	1	0
ELP1	0	0	0	0	0	0	0	0	0	0
THRA	0	1	0	0	0	0	1	0	0	0
LINC01431	0	0	0	0	0	1	0	0	0	0



Graphs of genes

- Gene networks are graphs of genes (10k+ papers per year)
- Connected by functionality or co-expression in cell types
- Common to start from a correlation matrix
- Get an adjacency matrix by hard threshold
- Can get **several graphs** with different thresholds

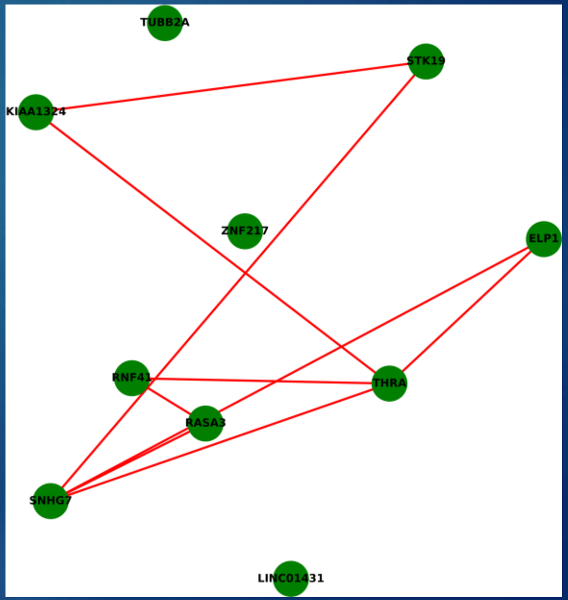
	TUBB2A	ZNF217	SNHG7	STK19	KIAA1324	RNF41	RASA3	ELP1	THRA	LINC01431
TUBB2A	0	0	0	0	1	0	0	0	0	0
ZNF217	0	0	0	1	1	1	1	0	1	0
SNHG7	0	0	0	0	0	1	0	0	0	0
STK19	0	1	0	0	0	0	0	0	0	0
KIAA1324	1	1	0	0	0	0	0	0	0	0
RNF41	0	1	1	0	0	0	0	0	0	1
RASA3	0	1	0	0	0	0	0	0	1	0
ELP1	0	0	0	0	0	0	0	0	0	0
THRA	0	1	0	0	0	0	1	0	0	0
LINC01431	0	0	0	0	0	1	0	0	0	0



Graphs of genes

- Gene networks are graphs of genes (10k+ papers per year)
- Connected by functionality or co-expression in cell types
- Common to start from a correlation matrix
- Get an adjacency matrix by hard threshold
- Can get **several graphs** with different thresholds

	TUBB2A	ZNF217	SNHG7	STK19	KIAA1324	RNF41	RASA3	ELP1	THRA	LINC01431
TUBB2A	0	0	0	0	0	0	0	0	0	0
ZNF217	0	0	0	0	0	0	0	0	0	0
SNHG7	0	0	0	1	0	0	1	1	1	0
STK19	0	0	1	0	1	0	0	0	0	0
KIAA1324	0	0	0	1	0	0	0	0	1	0
RNF41	0	0	0	0	0	0	1	0	1	0
RASA3	0	0	1	0	0	1	0	0	0	0
ELP1	0	0	1	0	0	0	0	0	1	0
THRA	0	0	1	0	1	1	0	1	0	0
LINC01431	0	0	0	0	0	0	0	0	0	0



Graphs of genes

- Gene networks are graphs of genes (10k+ papers per year)
- Connected by functionality or co-expression in cell types
- Common to start from a correlation matrix
- Get an adjacency matrix by hard threshold
- Can get **several graphs** with different thresholds
- We use the coex matrix [COTAN]

Graphs of genes

- Gene networks are graphs of genes (10k+ papers per year)
- Connected by functionality or co-expression in cell types
- Common to start from a correlation matrix
- Get an adjacency matrix by hard threshold
- Can get **several graphs** with different thresholds
- We use the coex matrix [COTAN]
- G1: $\text{corr} > 0.054$, G2: $\text{corr} < -0.06$

Graphs of genes
and neural networks

Graphs of genes and neural networks

Atypical GNN task

Graphs of genes and neural networks

Atypical GNN task

- The graph is **fixed** and independent from the samples (gene network)

Graphs of genes and neural networks

Atypical GNN task

- The graph is **fixed** and independent from the samples (gene network)
- Features at each node depend on the sample (=cell) to classify

Graphs of genes and neural networks

Atypical GNN task

- The graph is **fixed** and independent from the samples (gene network)
- Features at each node depend on the sample (=cell) to classify
- Only **one scalar feature per node** (expression levels)

Graphs of genes and neural networks

Atypical GNN task

- The graph is **fixed** and independent from the samples (gene network)
- Features at each node depend on the sample (=cell) to classify
- Only **one scalar feature per node** (expression levels)
- Graph-level task

Graphs of genes and neural networks

Atypical GNN task

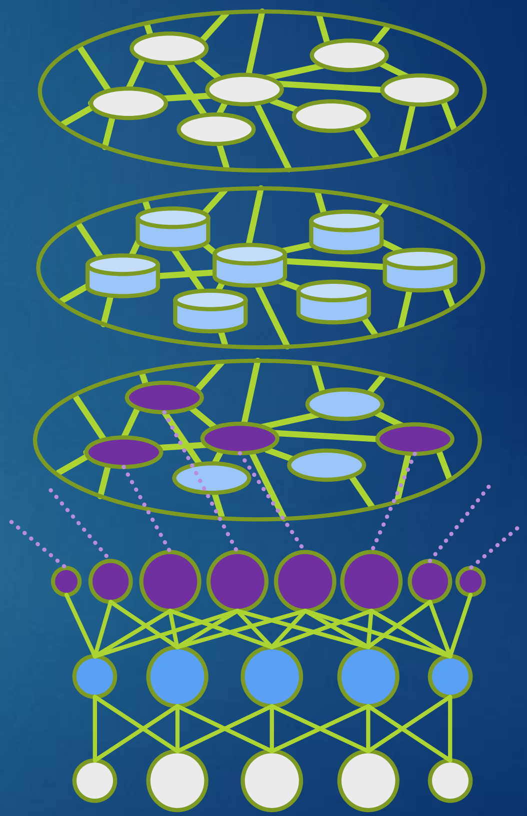
- The graph is **fixed** and independent from the samples (gene network)
- Features at each node depend on the sample (=cell) to classify
- Only **one scalar feature per node** (expression levels)
- Graph-level task
- Nodes are **not exchangeable**: global pooling does not work

Straightforward GNN models

- Nodes are **not exchangeable**: global pooling does not work

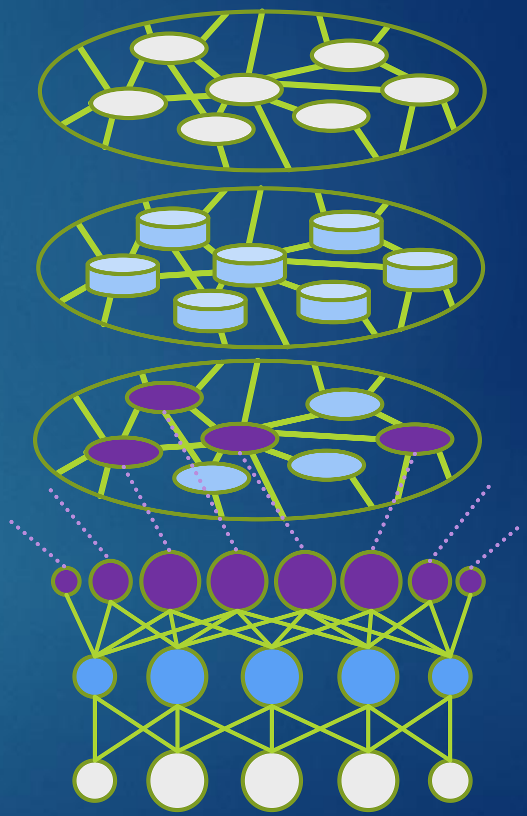
Straightforward GNN models

- Nodes are **not exchangeable**: global pooling does not work
- Basic structure:



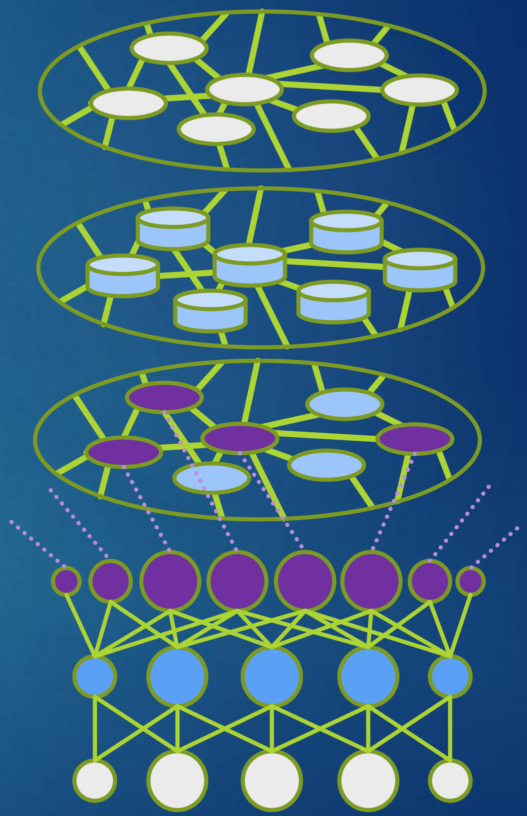
Straightforward GNN models

- Nodes are **not exchangeable**: global pooling does not work
- Basic structure:
  - start with all N genes



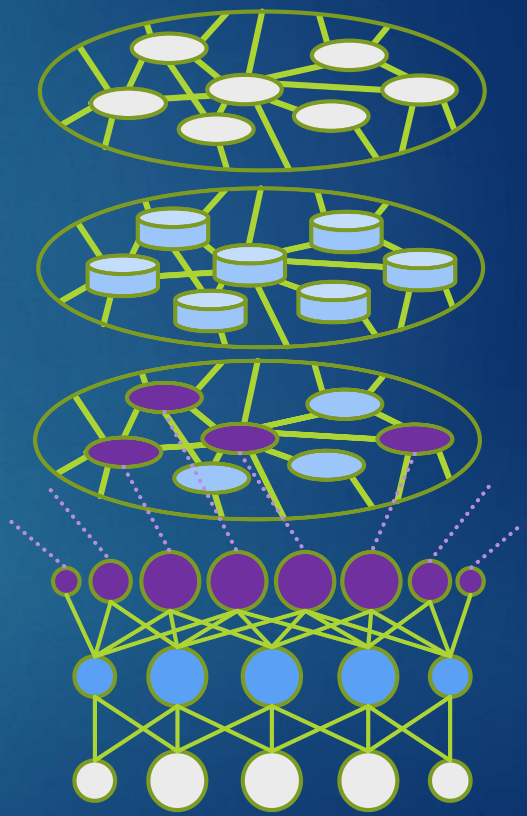
Straightforward GNN models

- Nodes are **not exchangeable**: global pooling does not work
- Basic structure:
  - start with all N genes
  - **two GNN layers** with 16 and 1 features per node



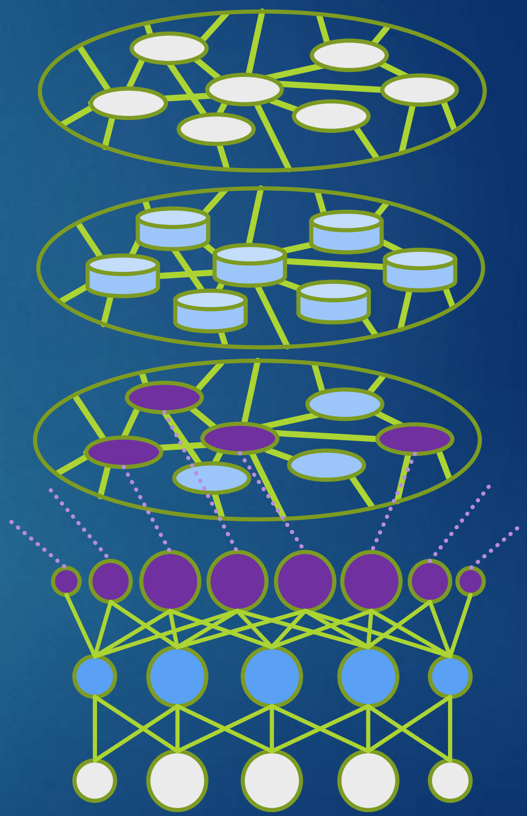
Straightforward GNN models

- Nodes are **not exchangeable**: global pooling does not work
- Basic structure:
  - start with all N genes
  - **two GNN layers** with 16 and 1 features per node
  - **restrict to n nodes** (n highest variability genes)



Straightforward GNN models

- Nodes are **not exchangeable**: global pooling does not work
- Basic structure:
  - start with all N genes
  - **two GNN layers** with 16 and 1 features per node
  - **restrict to n nodes** (n highest variability genes)
  - flatten, then MLP (as before, 1 hidden layer with 64 units)



Straightforward GNN models

- Ideas to improve:

Straightforward GNN models

- Ideas to improve:
  - skip connection (after GNN, add the input)

Straightforward GNN models

- Ideas to improve:
  - skip connection (after GNN, add the input)
  - extend 1-dim input with features of the gene population

Straightforward GNN models

- Ideas to improve:
  - skip connection (after GNN, add the input)
  - extend 1-dim input with features of the gene population
  - use several graphs in parallel for the first layer, concatenate before second layer

Common GNN layers

Common GNN layers

- **GraphSAGE** [Hamilton, Ying, Leskovec, 2017]

Common GNN layers

- **GraphSAGE** [Hamilton, Ying, Leskovec, 2017]

$$x_i' = W_1 x_i + W_2 \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} x_j$$

Common GNN layers

- **GraphSAGE** [Hamilton, Ying, Leskovec, 2017]

$$x_i' = W_1 x_i + W_2 \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} x_j$$

- **Graph Convolutional Network** [Kipf, Welling, 2017]

Common GNN layers

- **GraphSAGE** [Hamilton, Ying, Leskovec, 2017]

$$x_i' = W_1 x_i + W_2 \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} x_j$$

- **Graph Convolutional Network** [Kipf, Welling, 2017]

$$x_i' = W \sum_{j \in \mathcal{N}(i) \cup \{i\}} \frac{1}{\sqrt{\hat{d}_i \hat{d}_j}} x_j$$

Common GNN layers

- **Graph Attention Networks** [Veličković et al, 2018]
  GATv2 [Brody, Alon, Yahav, 2021]

Common GNN layers

- **Graph Attention Networks** [Veličković et al, 2018]
  GATv2 [Brody, Alon, Yahav, 2021]

$$x_i' = W_2 \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{i,j} x_j$$

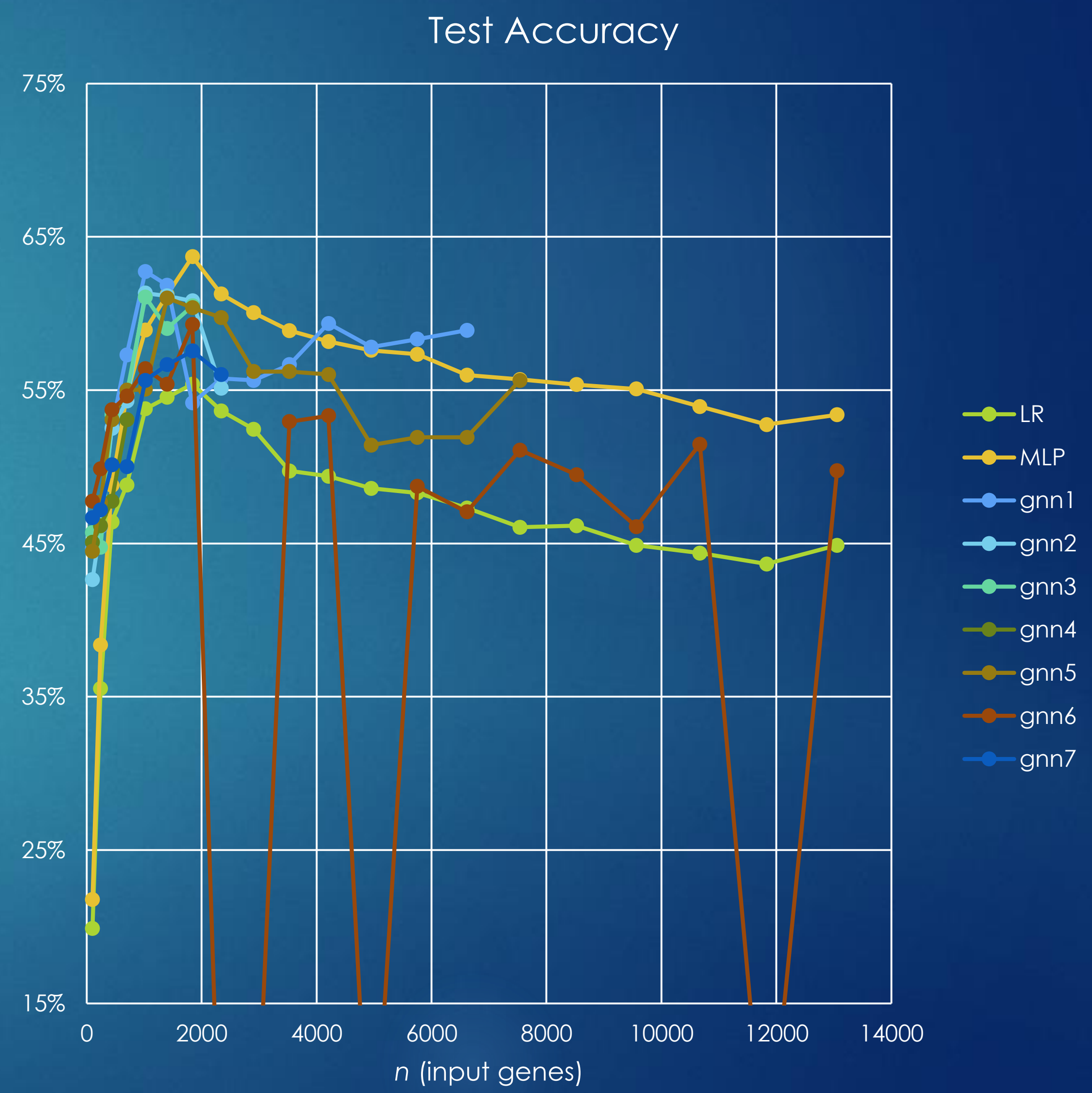
$$\alpha_{i,j} = \underset{j \in \mathcal{N}(i) \cup \{i\}}{\text{softmax}} \left(a^\intercal \varphi(W_1 x_i + W_2 x_j) \right)$$

Common GNN layers

- **Nothing works**: best performance worse than MLP

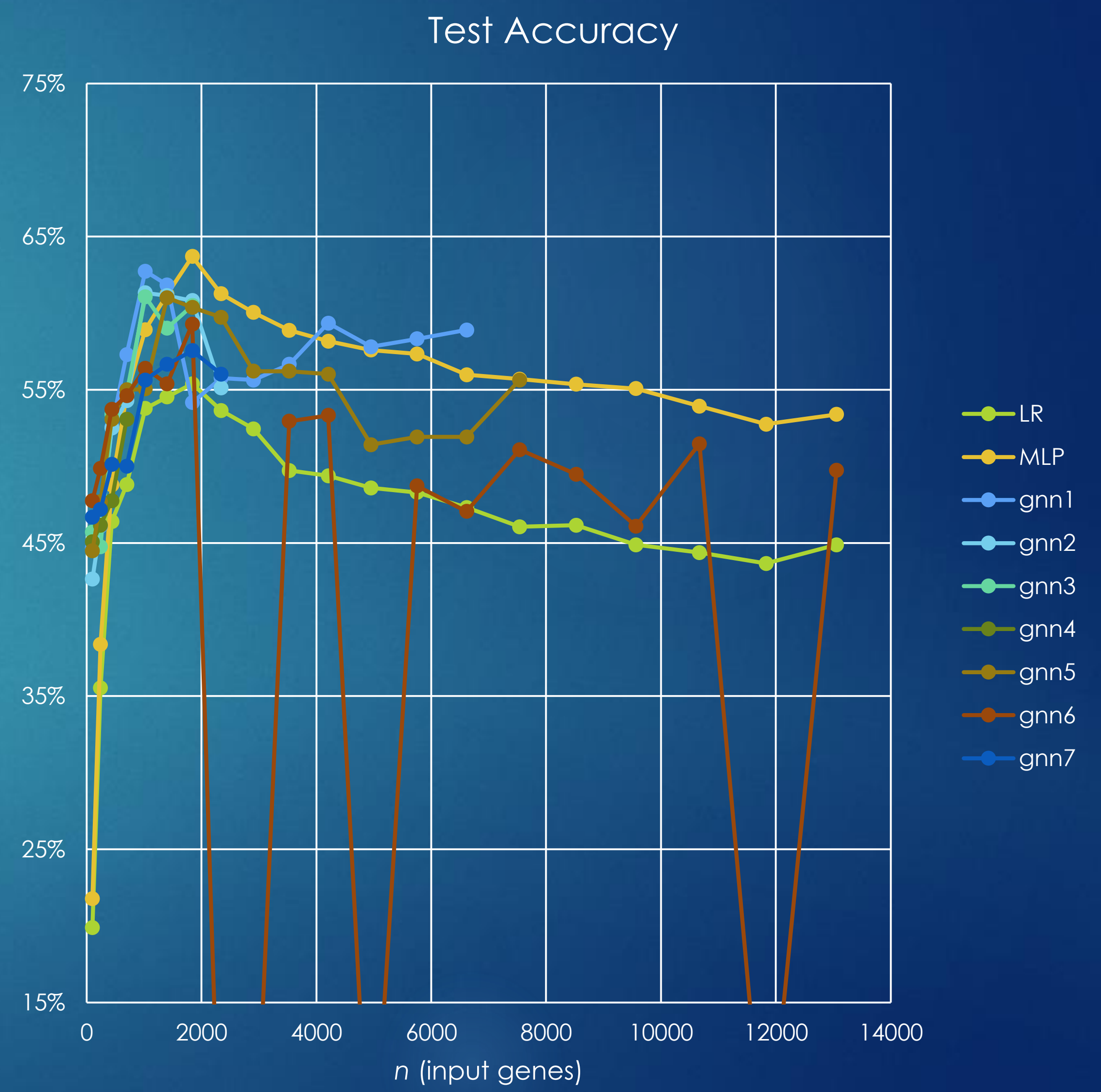
Common GNN layers

- **Nothing works**: best performance worse than MLP



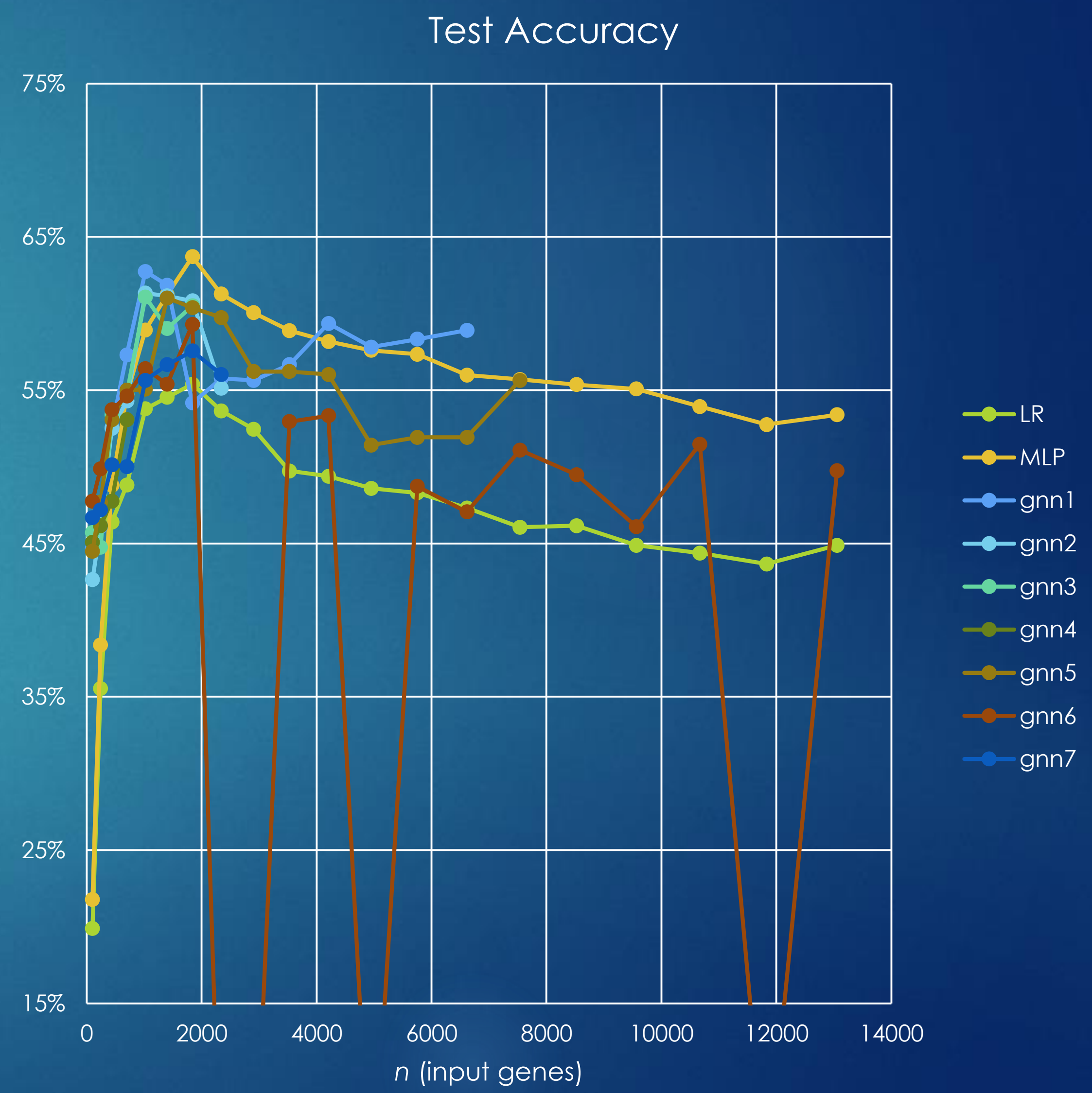
Common GNN layers

- **Nothing works**: best performance worse than MLP
- GNN layers average features among neighbours



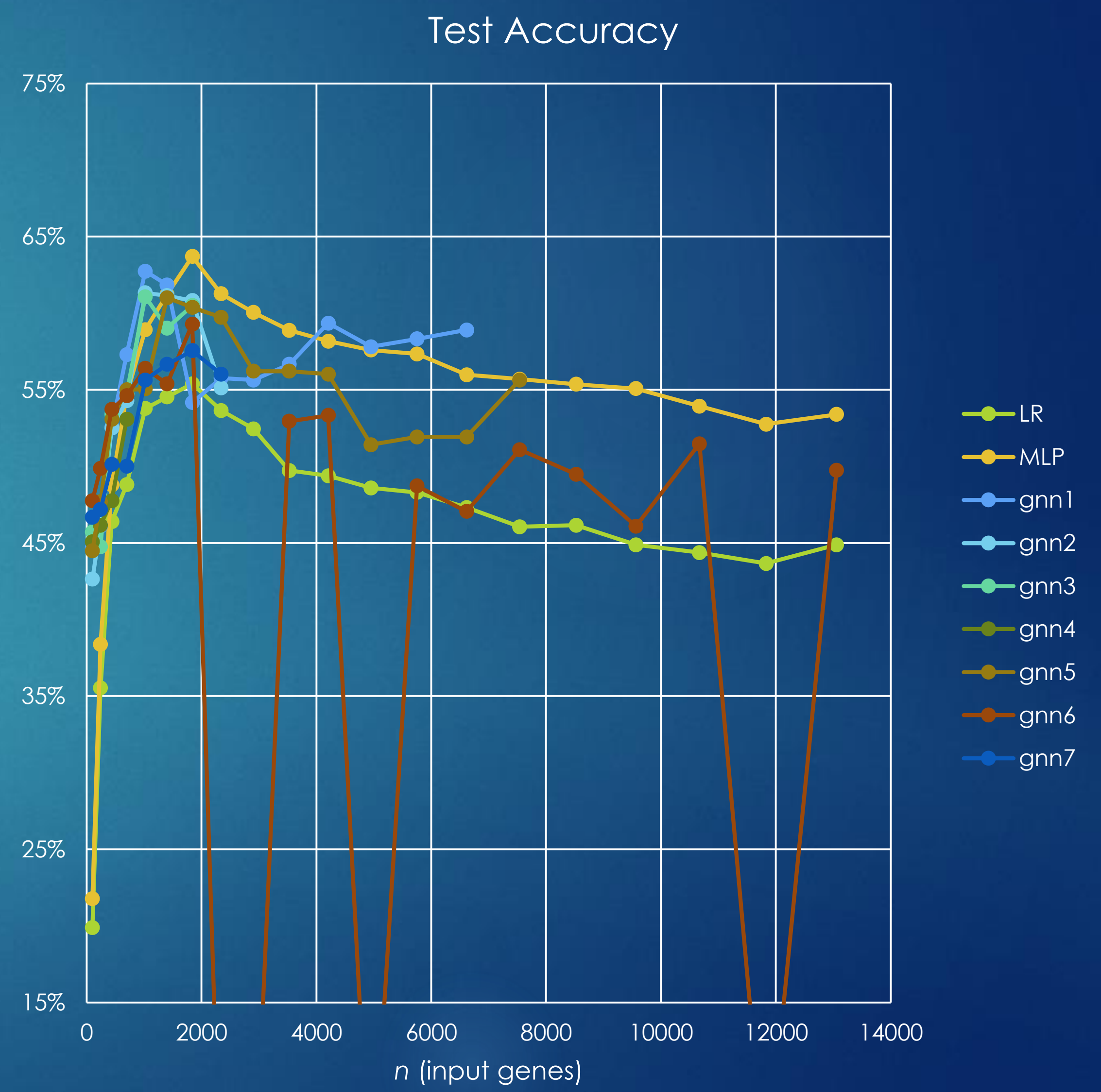
Common GNN layers

- **Nothing works**: best performance worse than MLP
- GNN layers average features among neighbours
- In gene networks, neighbours:



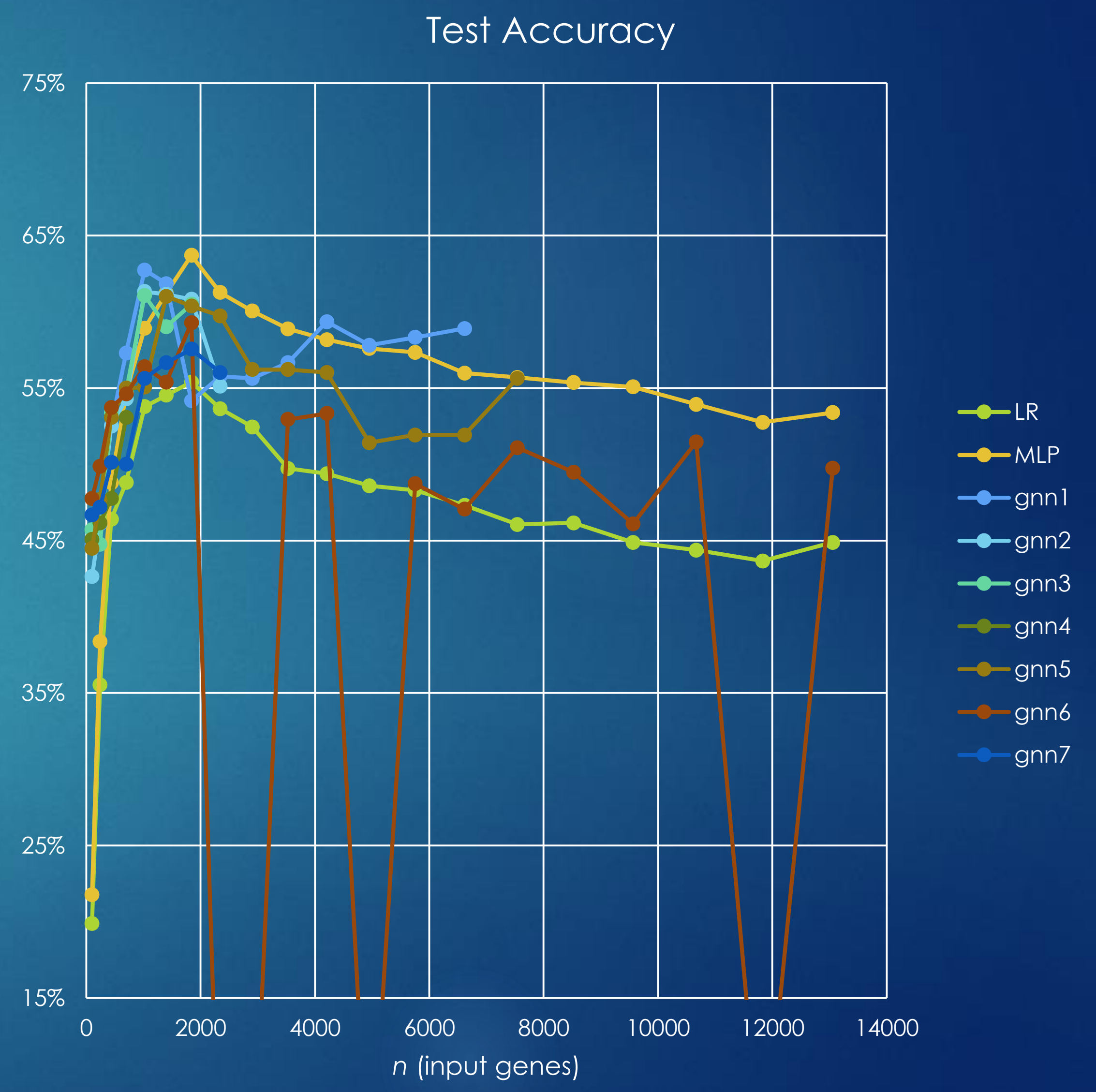
Common GNN layers

- **Nothing works**: best performance worse than MLP
- GNN layers average features among neighbours
- In gene networks, neighbours:
  - are correlated with the central node, but



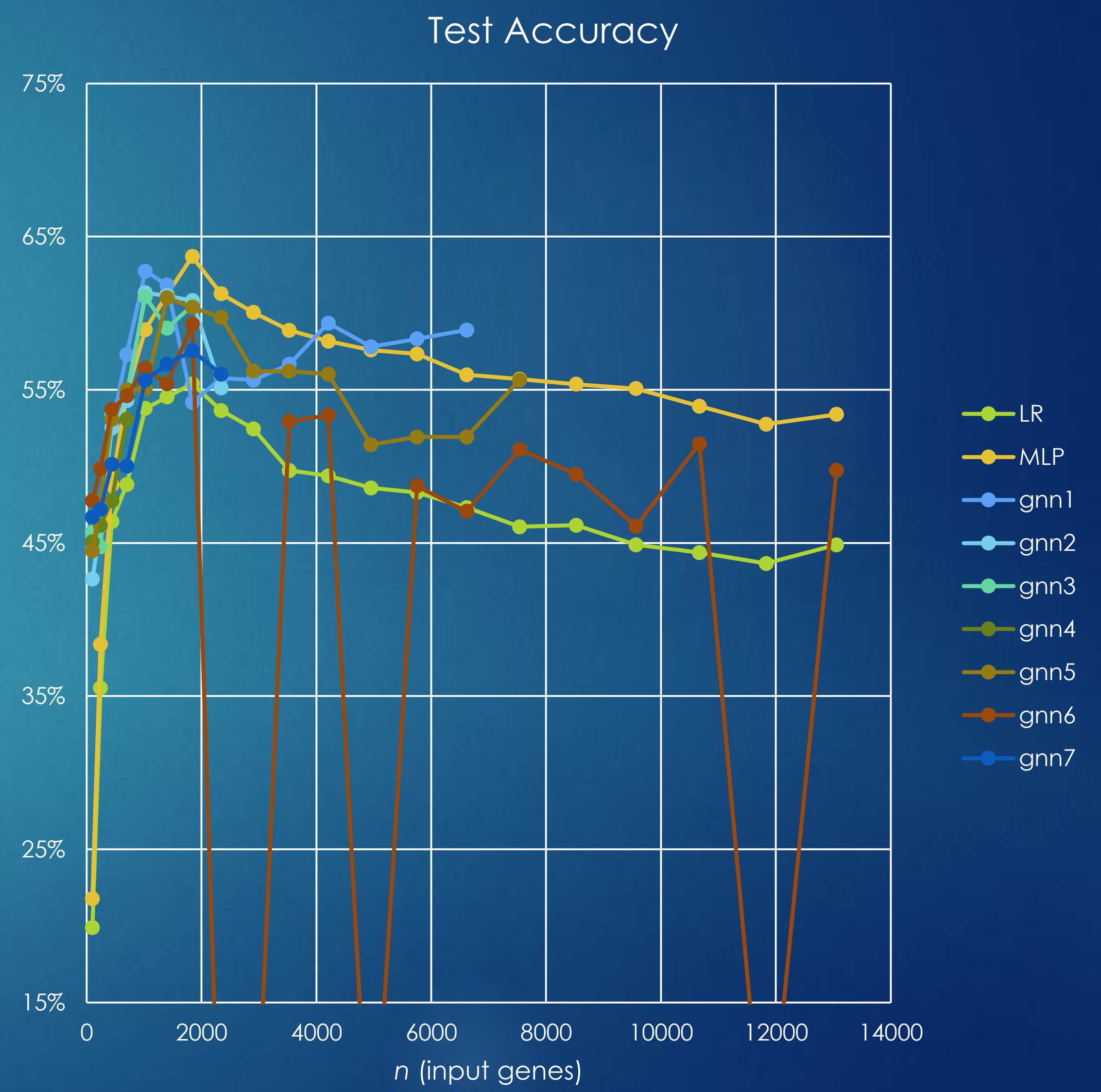
Common GNN layers

- **Nothing works**: best performance worse than MLP
- GNN layers average features among neighbours
- In gene networks, neighbours:
  - are correlated with the central node, but
  - they are not similar to each other



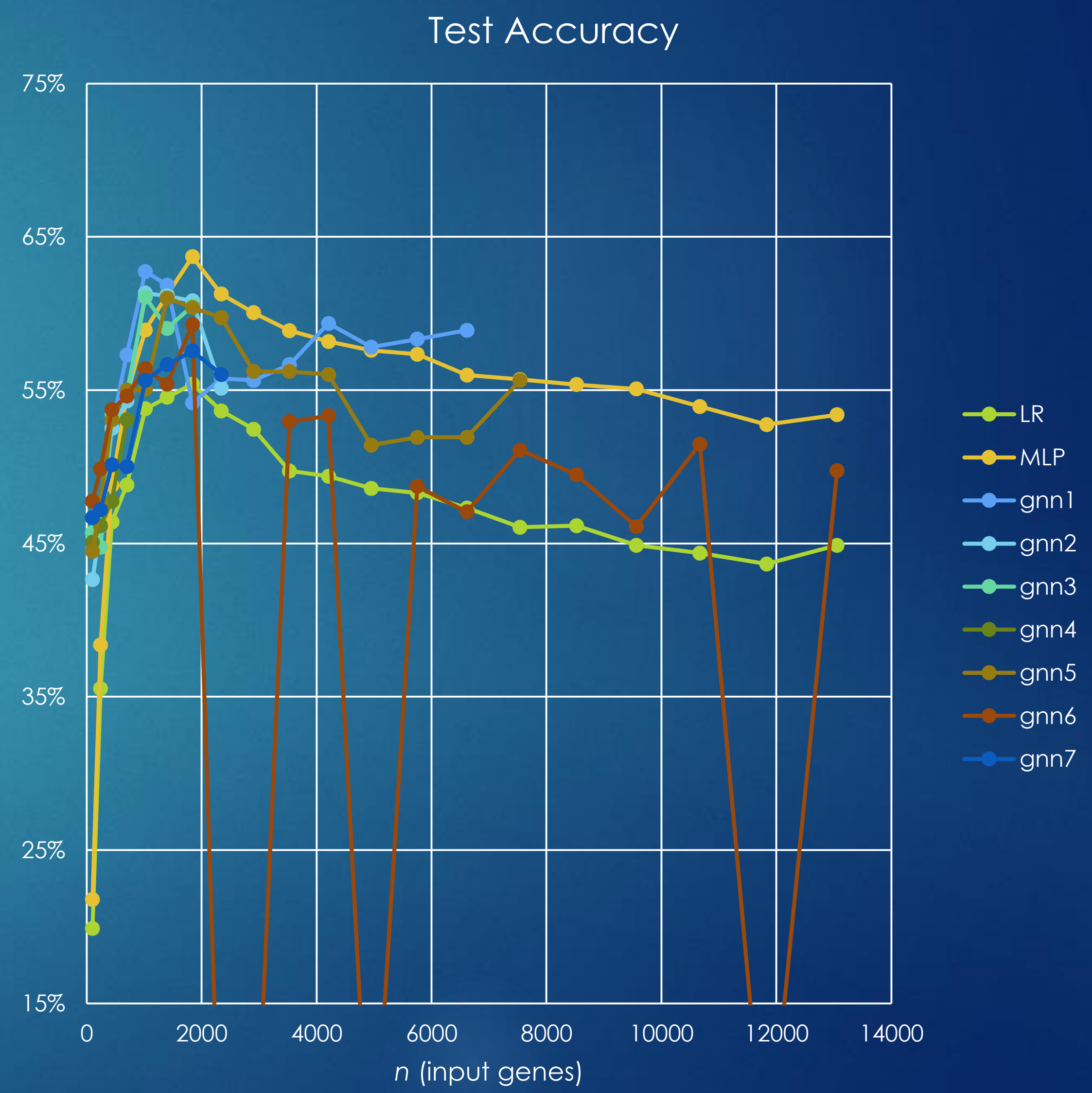
Common GNN layers

- **Nothing works**: best performance worse than MLP
- GNN layers average features among neighbours
- In gene networks, neighbours:
  - are correlated with the central node, but
  - they are not similar to each other
  - they belong to different families



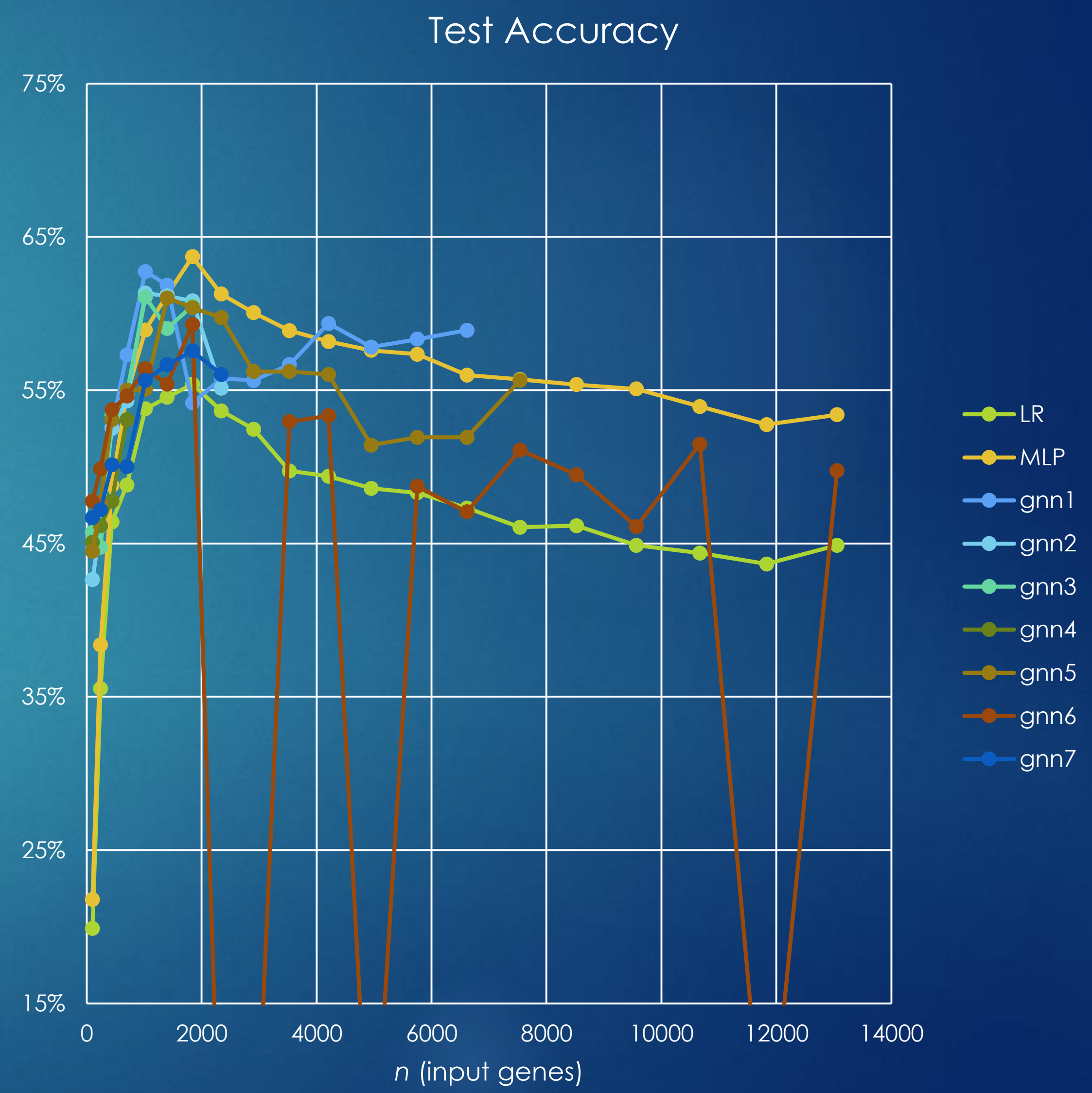
Common GNN layers

- **Nothing works**: best performance worse than MLP
- GNN layers average features among neighbours
- In gene networks, neighbours:
  - are correlated with the central node, but
  - they are not similar to each other
  - they belong to different families
  - they are correlated for different reasons



Common GNN layers

- **Nothing works**: best performance worse than MLP
- GNN layers average features among neighbours
- In gene networks, neighbours:
  - are correlated with the central node, but
  - they are not similar to each other
  - they belong to different families
  - they are correlated for different reasons
- We should make **several averages** within different families



Common GNN layers

Common GNN layers

- **Multi-head GAT**

$$x_i' = \Big\Vert_{h=1}^{K} W_2^h \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{i,j}^h x_j$$

$$\alpha_{i,j}^h = \underset{j \in \mathcal{N}(i) \cup \{i\}}{\mathrm{softmax}} \left(a_h^\intercal \varphi(W_1^h x_i + W_2^h x_j) \right)$$

Common GNN layers

- **Multi-head GAT**

$$x_i' = \big\Vert_{h=1}^{K} W_2^h \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{i,j}^h x_j$$

$$\alpha_{i,j}^h = \operatorname*{softmax}_{j \in \mathcal{N}(i) \cup \{i\}} \left(a_h^\intercal \varphi(W_1^h x_i + W_2^h x_j)\right)$$

- Still doesn't work

Common GNN layers

- **Multi-head GAT**

$$x_i' = \Big\Vert_{h=1}^{K} W_2^h \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{i,j}^h x_j$$

$$\alpha_{i,j}^h = \underset{j \in \mathcal{N}(i) \cup \{i\}}{\operatorname{softmax}} \left(a_h^\intercal \varphi(W_1^h x_i + W_2^h x_j)\right)$$

- Still doesn't work
- (Maybe also because it needs richer input features)

Graph Cheat-Attention Network

(TENTATIVE NAME)

Graph Cheat-Attention Network
(tentative name)

Graph Cheat-Attention Network
(tentative name)

- We add as input features, specific **signatures** of each gene

Graph Cheat-Attention Network
(tentative name)

- We add as input features, specific **signatures** of each gene
  - these signatures are vectors in $\mathbb{R}^D$ (we use $D = 64$)

Graph Cheat-Attention Network
(tentative name)

- We add as input features, specific **signatures** of each gene
  - these signatures are vectors in $\mathbb{R}^D$ (we use $D = 64$)
  - they are trained to be similar for genes that are similar for our task

Graph Cheat-Attention Network
(tentative name)

- We add as input features, specific **signatures** of each gene
  - these signatures are vectors in $\mathbb{R}^D$ (we use $D = 64$)
  - they are trained to be similar for genes that are similar for our task
  - to this end, we introduce an auxiliary MLP **using all genes**

Graph Cheat-Attention Network

(tentative name)

- We add as input features, specific **signatures** of each gene
  - these signatures are vectors in $\mathbb{R}^D$ (we use $D = 64$)
  - they are trained to be similar for genes that are similar for our task
  - to this end, we introduce an auxiliary MLP **using all genes**

$$y = b_1 + W_1 \varphi(b_0 + W_0\, x), \qquad W_0 \colon \mathbb{R}^N \to \mathbb{R}^D, \quad W_1 \colon \mathbb{R}^D \to \mathbb{R}^C$$

Graph Cheat-Attention Network
(tentative name)

- We add as input features, specific **signatures** of each gene
  - these signatures are vectors in $\mathbb{R}^D$ (we use $D = 64$)
  - they are trained to be similar for genes that are similar for our task
  - to this end, we introduce an auxiliary MLP **using all genes**

$$y = b_1 + W_1 \varphi(b_0 + W_0\, x), \qquad W_0 \colon \mathbb{R}^N \to \mathbb{R}^D, \quad W_1 \colon \mathbb{R}^D \to \mathbb{R}^C$$

- the **signatures** are the **weights** of the first fully connected layer

the column $W_0^{(i)}$ is the signature of gene i

Graph Cheat-Attention Network
(tentative name)

Graph Cheat-Attention Network
(tentative name)

- The signatures are used only for the attention

Graph Cheat-Attention Network
(tentative name)

- The signatures are used only for the attention

$$\alpha_i = \sigma\left(b + A\, W_0^{(i)}\right), \qquad A : \mathbb{R}^D \rightarrow \mathbb{R}^K$$

Graph Cheat-Attention Network
(tentative name)

- The signatures are used only for the attention

$$\alpha_i = \sigma\left(b + A\,W_0^{(i)}\right), \qquad A : \mathbb{R}^D \to \mathbb{R}^K$$

- The attention is an **attribute of the node**, not the edge

Graph Cheat-Attention Network
(tentative name)

- The signatures are used only for the attention

$$\alpha_i = \sigma\left(b + A\,W_0^{(i)}\right), \qquad A: \mathbb{R}^D \to \mathbb{R}^K$$

- The attention is an **attribute of the node**, not the edge
- We use multi-head attention, with $K = 64$ heads

Graph Cheat-Attention Network
(tentative name)

- The signatures are used only for the attention

$$\alpha_i = \sigma\big(b + A\,W_0^{(i)}\big), \qquad A : \mathbb{R}^D \to \mathbb{R}^K$$

- The attention is an **attribute of the node**, not the edge
- We use multi-head attention, with $K = 64$ heads

$$x_i' = \Big\Vert_{h=1}^{K} \left[W_2^h\, \alpha_i^h\, x_i + W_3^h \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \alpha_j^h\, x_j \right]$$

Graph Cheat-Attention Network
(tentative name)

Graph Cheat-Attention Network
(tentative name)

- Compare GCAT and GAT

Graph Cheat-Attention Network
(tentative name)

- Compare GCAT and GAT

$$x_i' = \Big\Vert_{h=1}^{K} \left[W_2^h \, \alpha_i^h \, x_i + W_3^h \, \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \alpha_j^h \, x_j \right]$$

Graph Cheat-Attention Network
(tentative name)

- Compare GCAT and GAT

$$x_i' = \Big\Vert_{h=1}^{K} \left[W_2^h \, \alpha_i^h \, x_i + W_3^h \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \alpha_j^h \, x_j \right]$$

$$x_i' = \Big\Vert_{h=1}^{K} W_2^h \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{i,j}^h \, x_j$$

Graph Cheat-Attention Network

(tentative name)

- Compare GCAT and GAT
  - W is different for i
  - either K or $\dim(Wx)$ is large

$$x_i' = \Big\Vert_{h=1}^{K} \left[W_2^h \, \alpha_i^h \, x_i + W_3^h \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \alpha_j^h \, x_j \right]$$

$$x_i' = \Big\Vert_{h=1}^{K} W_2^h \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{i,j}^h \, x_j$$

Graph Cheat-Attention Network
(tentative name)

- Compare GCAT and GAT
  - W is different for i
  - either K or $\dim(Wx)$ is large

$$x_i' = \big\Vert_{h=1}^{K} \left[W_2^h \, \alpha_i^h \, x_i + W_3^h \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \alpha_j^h \, x_j \right]$$

$$x_i' = \big\Vert_{h=1}^{K} W_2^h \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{i,j}^h \, x_j$$

$$\alpha_i^h = \sigma\left(b^h + a_h^\intercal W_0^{(i)}\right)$$

Graph Cheat-Attention Network
(tentative name)

- Compare GCAT and GAT
  - W is different for i
  - either K or $\dim(Wx)$ is large

$$x_i' = \Big\Vert_{h=1}^{K} \left[W_2^h\, \alpha_i^h\, x_i + W_3^h \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \alpha_j^h\, x_j \right]$$

$$x_i' = \Big\Vert_{h=1}^{K} W_2^h \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{i,j}^h\, x_j$$

$$\alpha_i^h = \sigma\left(b^h + a_h^\intercal W_0^{(i)}\right)$$

$$\alpha_{i,j}^h = \operatorname*{softmax}_{j \in \mathcal{N}(i) \cup \{i\}} \left(a_h^\intercal \varphi(W_1^h\, x_i + W_2^h\, x_j)\right)$$

Graph Cheat-Attention Network
(tentative name)

- Compare GCAT and GAT
  - W is different for i
  - either K or $\dim(Wx)$ is large

$$x_i' = \Big\Vert_{h=1}^{K} \left[W_2^h \, \alpha_i^h \, x_i + W_3^h \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \alpha_j^h \, x_j \right]$$

$$x_i' = \Big\Vert_{h=1}^{K} W_2^h \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{i,j}^h \, x_j$$

  - α does not depend on x...
    ...only on the signatures

$$\alpha_i^h = \sigma\left(b^h + a_h^\intercal W_0^{(i)}\right)$$

$$\alpha_{i,j}^h = \operatorname*{softmax}_{j \in \mathcal{N}(i) \cup \{i\}} \left(a_h^\intercal \varphi(W_1^h \, x_i + W_2^h \, x_j)\right)$$

Graph Cheat-Attention Network
(tentative name)

- Compare GCAT and GAT
  - W is different for i
  - either K or $\dim(Wx)$ is large

$$x_i' = \Big\Vert_{h=1}^{K} \left[W_2^h \, \alpha_i^h \, x_i + W_3^h \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \alpha_j^h \, x_j \right]$$

$$x_i' = \Big\Vert_{h=1}^{K} W_2^h \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{i,j}^h \, x_j$$

  - α does not depend on x...
    ...only on the signatures
  - sum of α's can be less than 1

$$\alpha_i^h = \sigma\left(b^h + a_h^\intercal \, W_0^{(i)}\right)$$

$$\alpha_{i,j}^h = \operatorname*{softmax}_{j \in \mathcal{N}(i) \cup \{i\}} \left(a_h^\intercal \varphi(W_1^h \, x_i + W_2^h \, x_j)\right)$$

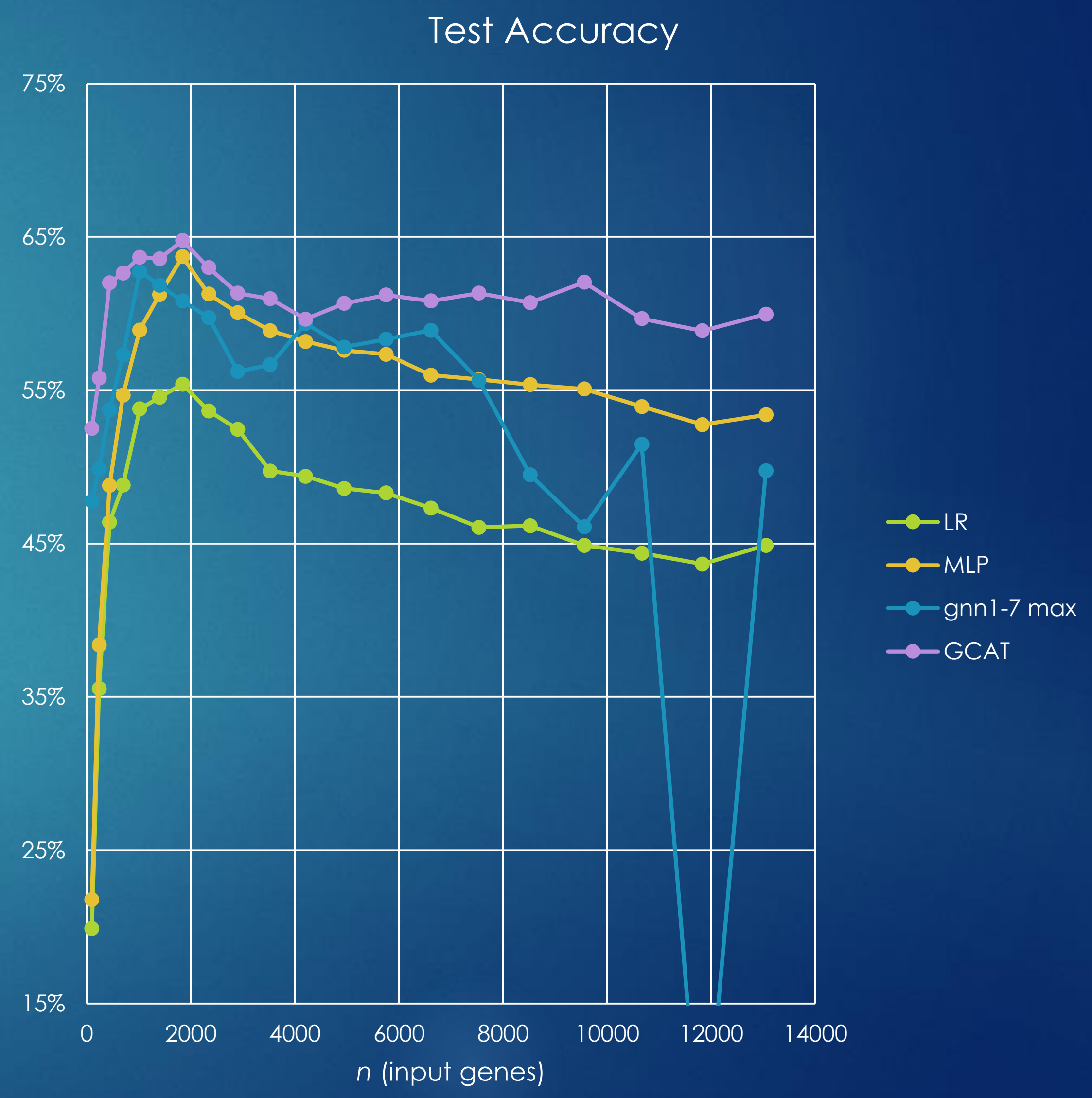
Graph Cheat-Attention Network
(tentative name)

Graph Cheat-Attention Network
(tentative name)

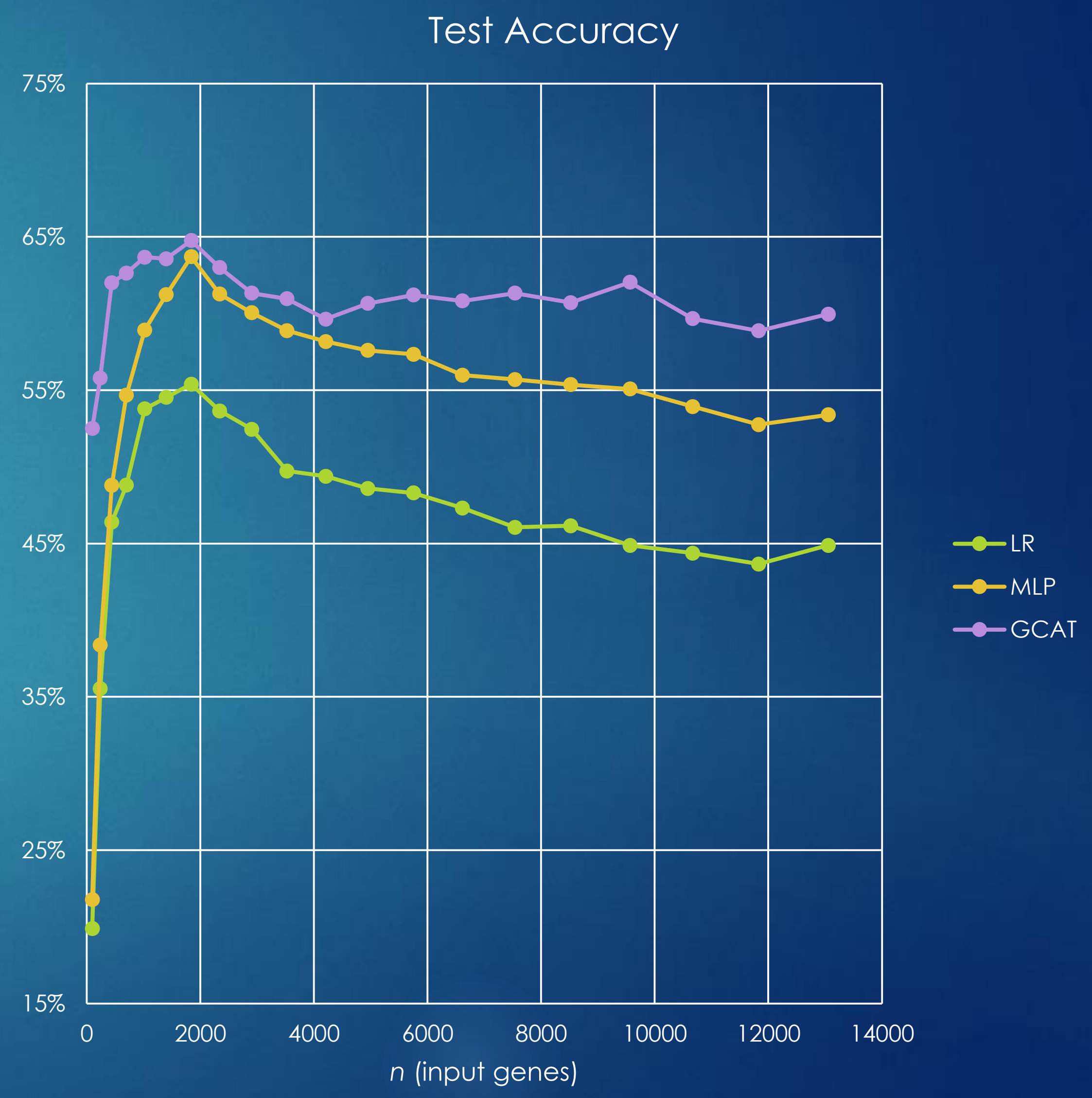
- GCAT works

Graph Cheat-Attention Network
(tentative name)

- GCAT works



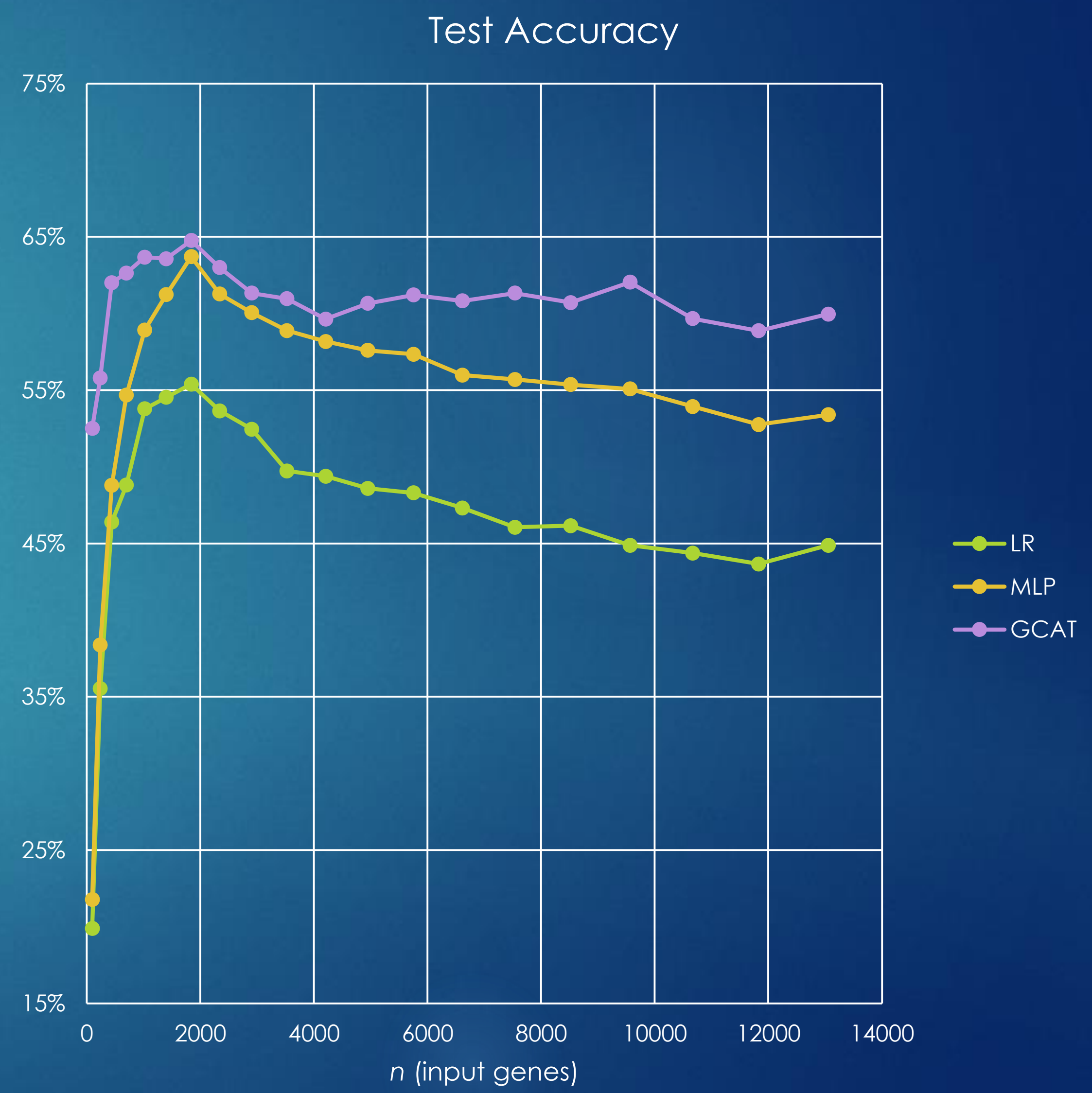
Graph Cheat-Attention Network
(tentative name)



Graph Cheat-Attention Network

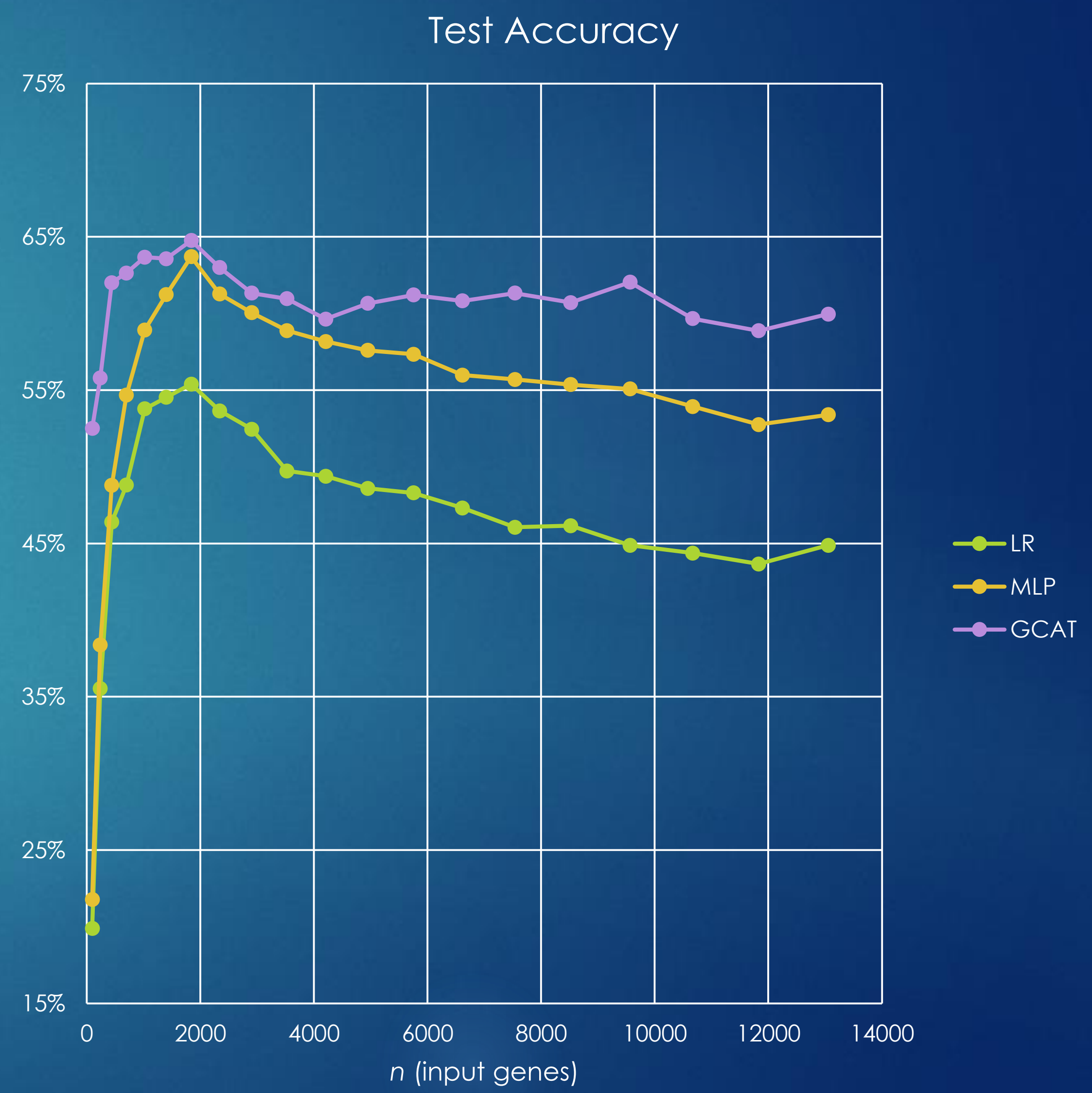
(tentative name)

- **auxiliary MLP** (1 hidden layer with 64 units)



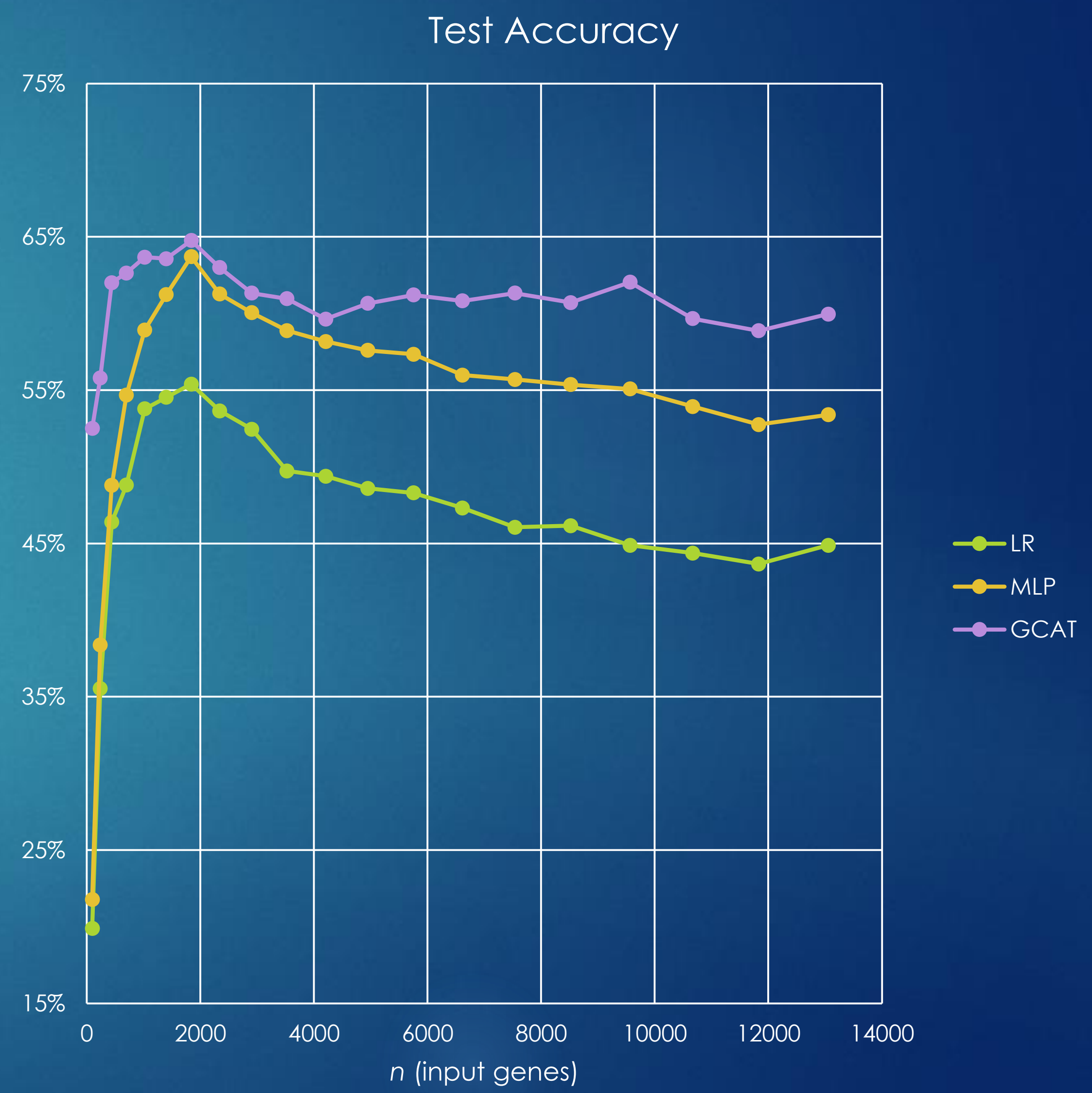
Graph Cheat-Attention Network
(tentative name)

- **auxiliary MLP** (1 hidden layer with 64 units)
- 64-dim **gene signatures**



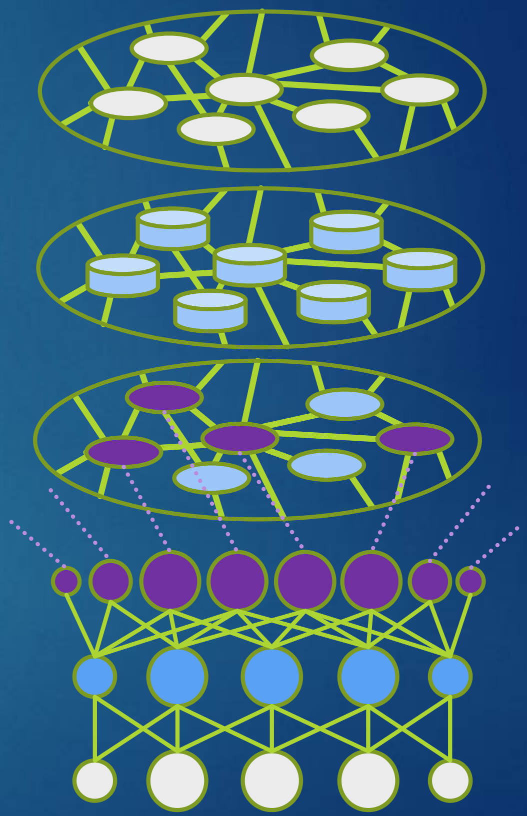
Graph Cheat-Attention Network
(tentative name)

- **auxiliary MLP** (1 hidden layer with 64 units)
- 64-dim **gene signatures**
- **cheat-attention** with 64 heads



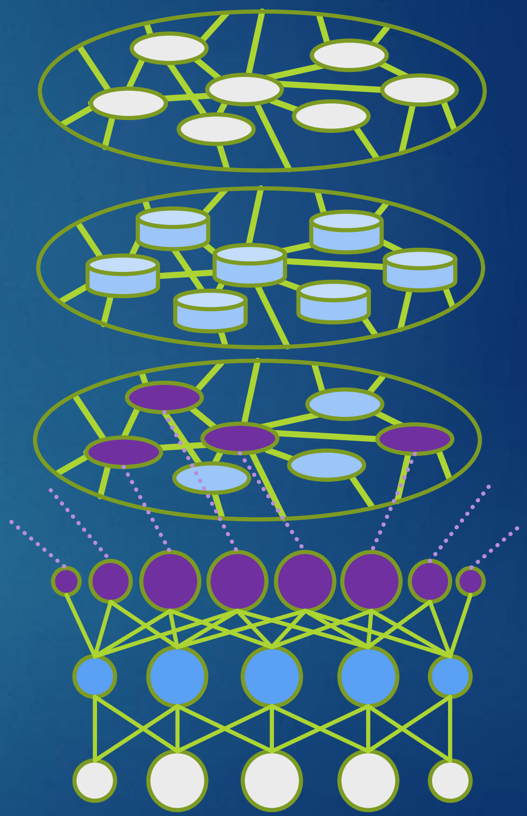
Graph Cheat-Attention Network
(tentative name)

- **auxiliary MLP** (1 hidden layer with 64 units)
- 64-dim **gene signatures**
- **cheat-attention** with 64 heads
- 1 **GCAT** layer using two graphs



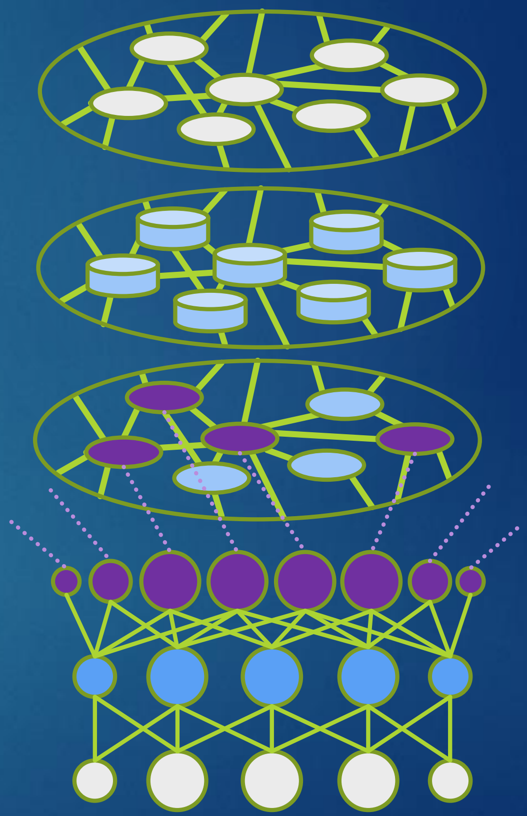
Graph Cheat-Attention Network
(tentative name)

- **auxiliary MLP** (1 hidden layer with 64 units)
- 64-dim **gene signatures**
- **cheat-attention** with 64 heads
- 1 **GCAT** layer using two graphs
- 1 graph **convolutional** layer with 1 output



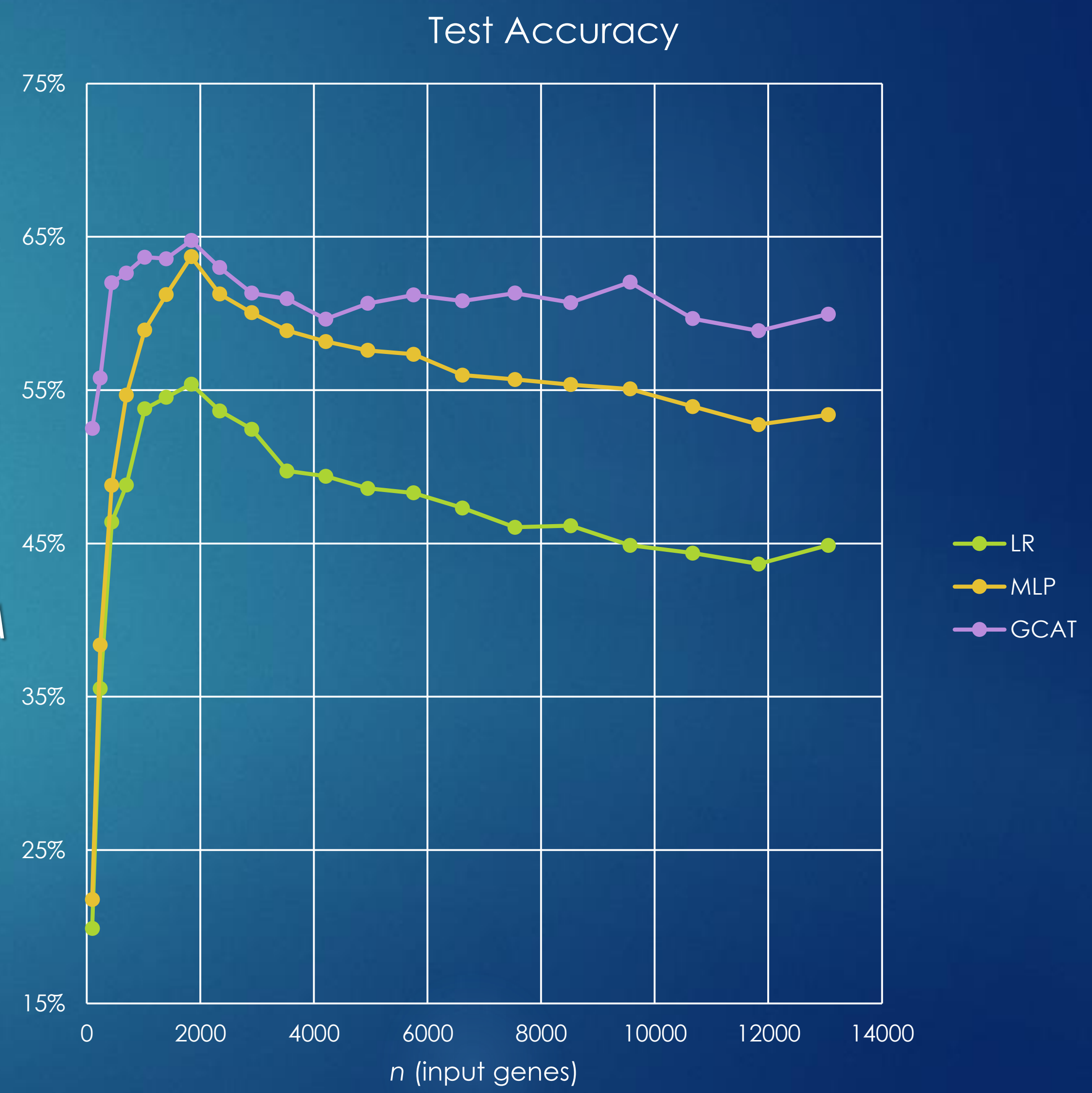
Graph Cheat-Attention Network
(tentative name)

- **auxiliary MLP** (1 hidden layer with 64 units)
- 64-dim **gene signatures**
- **cheat-attention** with 64 heads
- 1 **GCAT** layer using two graphs
- 1 graph **convolutional** layer with 1 output
- $\sim 64(N+n)$ parameters, from 0.85M to 1.7M



Graph Cheat-Attention Network
(tentative name)

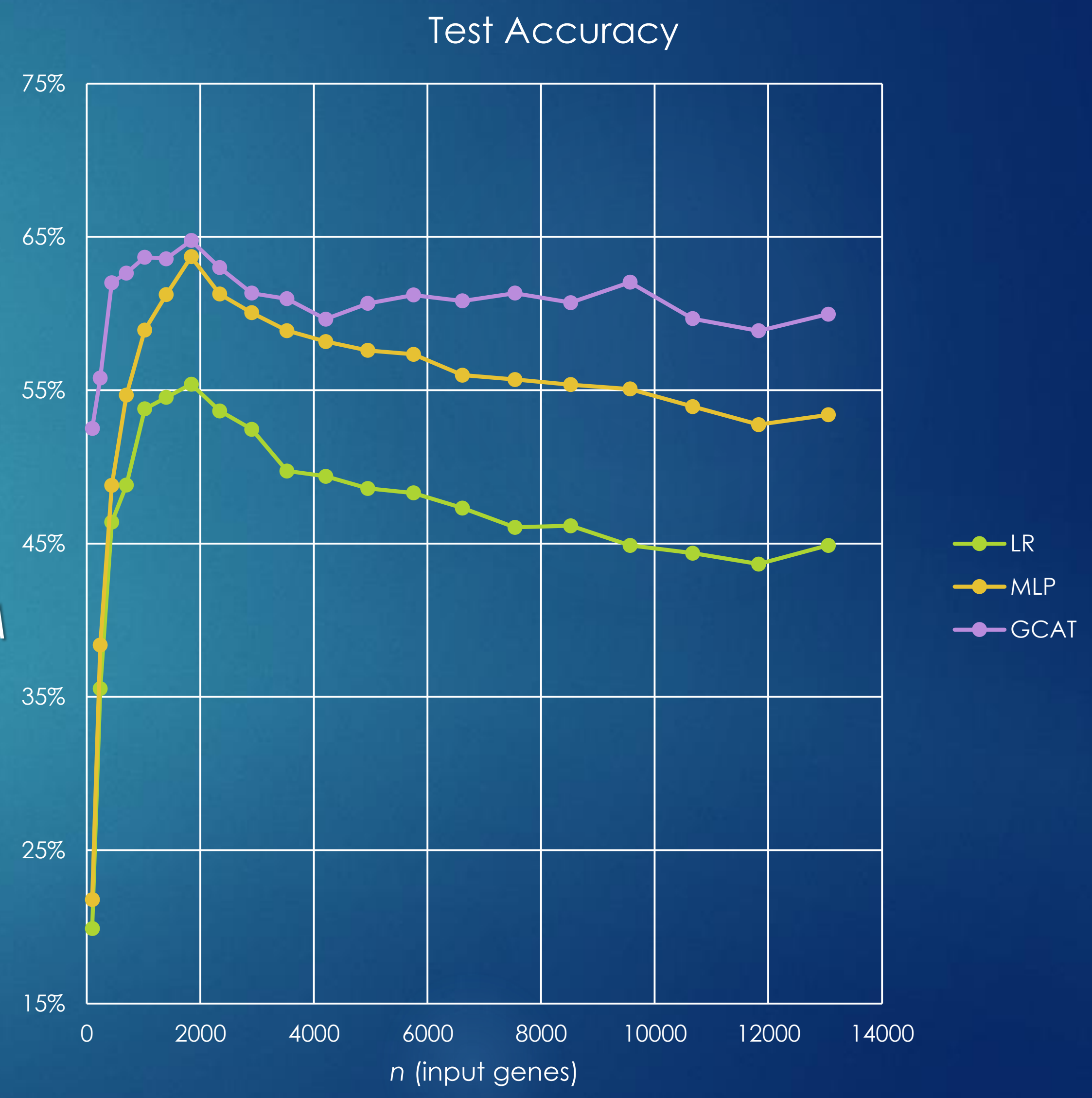
- **auxiliary MLP** (1 hidden layer with 64 units)
- 64-dim **gene signatures**
- **cheat-attention** with 64 heads
- 1 **GCAT** layer using two graphs
- 1 graph **convolutional** layer with 1 output
- $\sim 64(N+n)$ parameters, from 0.85M to 1.7M
- top TA 64.8% at $n = 1800$



Graph Cheat-Attention Network

(tentative name)

- **auxiliary MLP** (1 hidden layer with 64 units)
- 64-dim **gene signatures**
- **cheat-attention** with 64 heads
- 1 **GCAT** layer using two graphs
- 1 graph **convolutional** layer with 1 output
- $\sim 64(N + n)$ parameters, from 0.85M to 1.7M
- top TA 64.8% at $n = 1800$
- for large n, TA stays at about 60%



Thanks
for the attention