Comp683: Computational Biology

Lecture 22

April 15, 2025

Today

- Conclude lecture on optimization and multiomics for ADNI
- Technical Writing in Computational Biology
- Conclusion and summary of major themes from the semester.

Important announcements

- Homework 2, due Friday
- Presentations are next Monday, Wednesday, and Monday. Final writeups will be due by noon on our final exam day (April 30)

A Joint Model of Cognitive Scores and Diagnosis

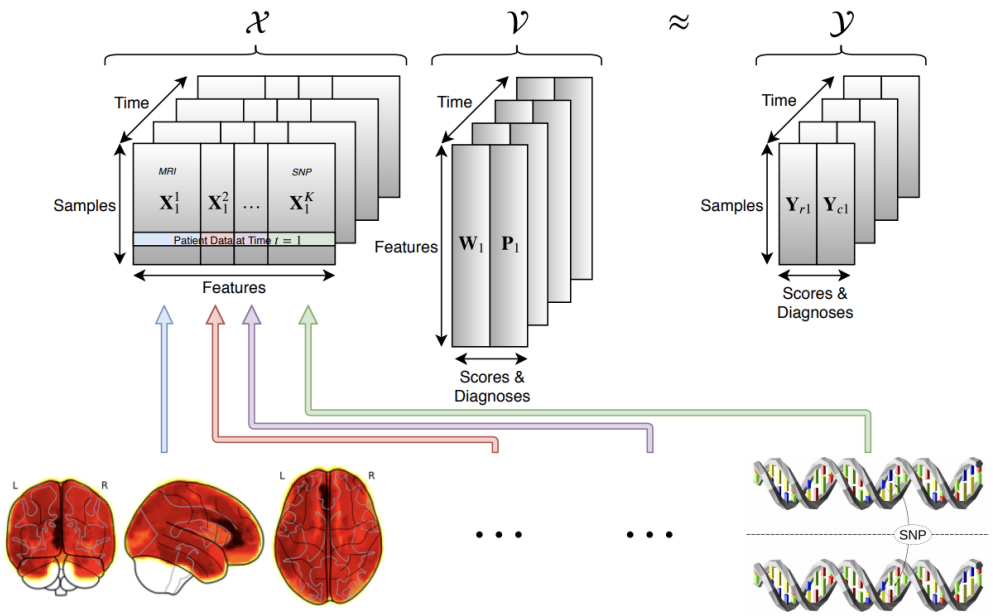


Figure: from Brand, Wang, *et al.* PacSym Biocomputing. 2020.

Notation and Problem Formulation

- **Input Features:** $\mathcal{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_T\} \in \mathbb{R}^{n \times d \times T}$ This represents patients $\times$ features $\times$ timepoints.
- Note that each $\mathbf{X}_t$ can be broken down across the K modalities as, $\{\mathbf{X}_t\}_{j=1}^{K}$
- The output diagnoses and cognitive scores are represented by another tensor, $\mathcal{Y} = \{\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_T\} \in \mathbb{R}^{n \times c \times T}$
- Each $\mathbf{Y}_t = [\mathbf{Y}_{rt}, \mathbf{Y}_{ct}]$ represents concatenated response variables for regression (r) and classification (c).
- The goal is to learn a tensor, $\mathcal{V} = \{[\mathbf{W}_1, \mathbf{P}_1], [\mathbf{W}_2, \mathbf{P}_2], [\mathbf{W}_T, \mathbf{P}_T]\}$ which represents the coefficients for each feature for regression ($\mathbf{W}$s) and classification ($\mathbf{P}$s) across the T timepoints.

A Regularized Joint Learning Model

$$\min_{\mathcal{W},\mathcal{P}} \mathcal{L}_r(\mathcal{W}) + \mathcal{L}_c(\mathcal{P}) + \mathcal{R}(\mathcal{V})$$

- Here, $\mathcal{L}_r(\mathcal{W})$ and $\mathcal{L}_c(\mathcal{P})$ are the loss functions for the regression and classification tasks, respectively.
- Regression and classification coefficient matrices are $\mathbf{W}_t \in \mathbb{R}^{d \times c_r}$ and $\mathbf{P}_t \in \mathbb{R}^{d \times c_c}$. This yields c total coefficients.
- $\mathcal{R}(\mathcal{V})$ is a regularization function applied to the matrix unfolded from the tensor, $\mathcal{V} \rightarrow \mathbf{V}^{d \times cT}$. Here, $\mathbf{V}^{d \times cT}$ is constructed by taking the $(\mathbf{W}_t, \mathbf{P}_t)$ matrix pairs and joining by the columns.

Regularization, $\mathcal{R}(\mathcal{V})$

To associate image and genomic features to cognitive scores and diagnoses over time, apply an $\ell_{2,1}$ norm to unfolded coefficient matrix as,

$$\mathbf{V} : ||\mathbf{V}||_{2,1} = \sum_{d=1}^{d} ||\mathbf{v}^i||_2$$

Next, to capture the impact of each modality (e.g. MRI, SNP, etc), the authors use the group ℓ_1-norm (G_1 norm) on the rows of $\mathbf{V}$ corresponding to modality j as,

$$||\mathbf{V}||_{G_1} = \sum_{j=1}^{K} ||\mathbf{V}^j||_2$$

Regularization, $\mathcal{R}(\mathcal{V}$, Continued

Finally, to account for inter-modal relationships (or relatedness of features within a modality to cognitive outcome), they use trace norm regularization of $\mathbf{V}$ as,

$$\mathbf{V} : ||\mathbf{V}||_* = \sum \sigma_i(\mathbf{V})$$

. Here, $\sigma_i(\mathbf{V})$ are the singular values of $\mathbf{V}$

Objective

Incorporating the three regularizations, the objective can be specified as follows,

$$\min_{\mathbf{V}} J = \sum_{t=1}^{T} \left[\|\mathbf{X}_t \mathbf{W}_t - \mathbf{Y}_{rt}\|_F^2 \right] + \sum_{t=1}^{T} \sum_{i=1}^{n} \sum_{k=1}^{c_c} \left[\left(1 - \left(\mathbf{x}^{it} \mathbf{p}_{kt} + b_{kt}\right) y_{ikt}\right)_+ \right] + \gamma_1 \|\mathbf{V}\|_{2,1} + \gamma_2 \|\mathbf{V}\|_{G_1} + \gamma_3 \|\mathbf{V}\|_* \quad ,$$

- The second term is the loss of $c_c \times T$ one-vs-all multi-class SVM
- $y_{ikt} \in \{-1, 1\}$ is the class label associated with the i-th patient at time t
- b_{kt} is the bias associated with the $(k \times t)$-th SVM
- $(\cdot)_+$ is defined as $(a)_+ = \max(0, a)$

Technical Writing Motivation

- In Comp Bio we write for two different audiences.
- Notation, figure presentation, publicly available code goes a long way.
- **Communication is Your Job!**
  - Good writing through simple language and organization
  - Well-documented publicly available code

Question

What part of technical writing do you find the most challenging?

Abstract: A Self Contained Story

- An elevator pitch of the main points
- Someone should read this and know exactly what your paper is about.
- Sentence breakdown
  - 1 sentence background
  - 1 sentence about what is still missing
  - 1 sentence about what you did
  - 1 sentence about what results suggest
  - 1 inspirational sentence about how this advances the field.

Introduction

General sections of an introduction.

- Problem motivation- what are we even talking about?
- Description of previous approaches to the problem.
  - Always highlight the work of others in a positive way
- A paragraph where you compare and contrast previous solutions. You can still discuss limitations by spinning them in relation to all of the positive things that the other authors have done.
- Paragraph giving an overview of your contributions. Someone might only read this section of your paper. You need to sell your contribution in a human-readable way.

Methods: First Defining Your Notation

- Notation needs to be clearly defined. There should never be a symbol in an equation that has not been properly defined.
- Keep bolding, italics, upper-case and lower-case consistent
- Dimensions of matrices need to be consistent represented with the same letter (usually p, d, or m)
- Indices should always map the same thing throughout the paper (for example i referring to cells and j referring to a feature of a cell)

Example Defining Notation

We start with some notation. We assume that we have an undirected graph $G = (V, E)$, where there are $n = |V|$ nodes with features on each node represented by a matrix $X \in \mathbb{R}^{n \times p}$. Let A be the adjacency matrix of the graph, D be the diagonal degree matrix, and S be the normalized adjacency matrix $D^{-1/2}AD^{-1/2}$. For the prediction problem, the node set V is split into a disjoint set of unlabeled nodes U and labeled nodes L, which are subsets of the indices $\{1, \ldots, n\}$. We will further split the labeled nodes into a training set L_t and validation set L_v. We represent the labels by a one-hot-encoding matrix $Y \in \mathbb{R}^{n \times c}$, where c is the number of classes (i.e., $Y_{ij} = 1$ if $i \in L$ is known to be in class j, and 0 otherwise, where the ith row of Y is all zero if $i \in U$), Our problem is transductive node classification: assign each node $j \in U$ a label in $\{1, \ldots, c\}$, given G, X, and Y.

Figure: from Huang *et al.* ICLR 2021.

Methods : Problem Formulation

- A section where you mathematically define your problem with the notation you introduced.
- What are your inputs and outputs? What are the dimensions of the inputs and outputs and what do they represent?
- Even if you write out your problem in text format, reference the variables that you defined in the text.

For example: 'For each cell, $\mathbf{x}_i \in \mathbb{R}^d$, we wish to learn its label, y_i through the use of the graph, $\mathcal{G}$.

Tip: Give Reminders

- It is good to keep reminding readers what notations and abstractions represent.
- For example, defining a graph? It doesn't hurt to remind them that nodes are cells and edges represent sufficient similarity between cells.
- Connect problem formulation to 'Figure 1'. In defining the overview of your problem, reference sub-panels of figure 1 of interest.

Example of a Comprehensive Figure 1

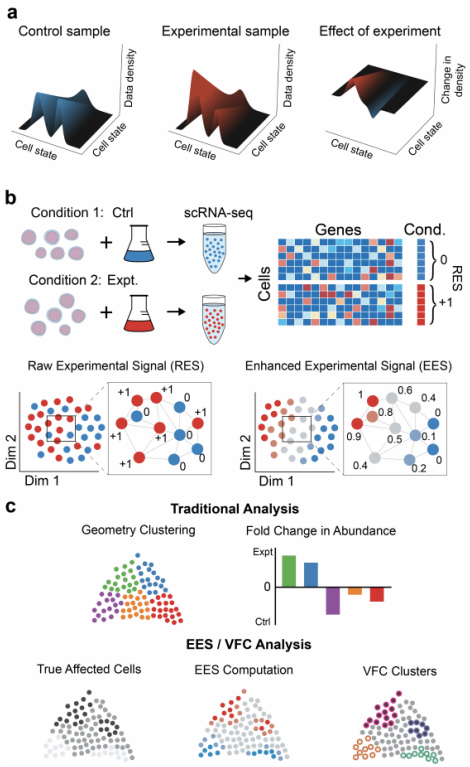


Figure: from Burkhardt *et al.* Nature Biotech. 2021.

Schematic Illustrations

If you draw cells, or patients, make sure these are carried through the entire figure.

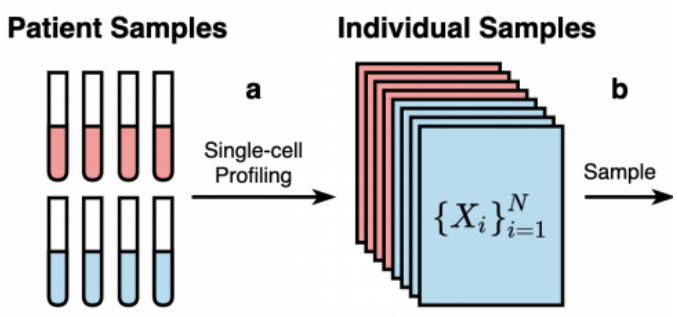


Figure: from Haidong Yi. `https://www.biorxiv.org/content/biorxiv/early/2021/04/14/2021.04.13.439702.full.pdf`

Pseudo-Code

Writing good pseudo code is extremely helpful. It can often by more helpful than the entire methods section.

Algorithm 2 xNetMF ($G_1, G_2, p, K, \gamma_s, \gamma_a$)

STEP 1. Node Identity Extraction
for node u in $\mathcal{V}_1 \cup \mathcal{V}_2$ **do**
  for hop k up to K **do** ▷ counts of node degrees of k-hop neighbors of u
    $\mathbf{d}_u^k$ = CountDegreeDistributions($\mathcal{R}_u^k$) ▷ $1 \leq K \leq$ graph diameter
  end for
  $\mathbf{d}_u = \sum_{k=1}^{K} \delta^{k-1} \mathbf{d}_u^k$ ▷ discount factor $\delta \in (0, 1]$
end for

STEP 2. Efficient Similarity-based Representation
STEP 2a. Reduced $n \times p$ Similarity Computation
$\mathcal{L}$ = ChooseLandmarks(G_1, G_2,p) ▷ choose p nodes from G_1, G_2
for node u in $\mathcal{V}$ **do**
  for node v in $\mathcal{L}$ **do**
    $c_{uv} = e^{-\gamma_s \cdot ||\mathbf{d}_u - \mathbf{d}_v||_2^2 - \gamma_a \cdot \text{dist}(\mathbf{f}_u, \mathbf{f}_v)}$
  end for
end for ▷ Used in low-rank approx. of similarity graph (not constructed)

STEP 2b. From Similarity to Representation
$\mathbf{W} = \mathbf{C}[\mathcal{L}, \mathcal{L}]$ ▷ Rows of $\mathbf{C}$ corresponding to landmark nodes
$[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \text{SVD}(\mathbf{W}^\dagger)$
$\tilde{\mathbf{Y}} = \mathbf{C}\mathbf{U}\mathbf{\Sigma}^{-\frac{1}{2}}$ ▷ **Embedding**: *implicit* factorization of similarity graph
$\tilde{\mathbf{Y}} = Normalize(\tilde{\mathbf{Y}})$ ▷ **Postprocessing**: make embeddings have magnitude 1
$\tilde{\mathbf{Y}}_1, \tilde{\mathbf{Y}}_2$ = Split($\tilde{\mathbf{Y}}$) ▷ **Separate** representations for nodes in G_1, G_2
return $\tilde{\mathbf{Y}}_1, \tilde{\mathbf{Y}}_2$

Figure: from `https://arxiv.org/pdf/1802.06257.pdf`

Results

- Figure/table legends should be self-contained. For example, if there is some kind of confidence interval around your curve, tell us what it represents
- Plotting: try to choose appropriate axis to capture all of the datapoints. Don't just plot for example between 0 and 1 on the y-axis by default.
- Make sure that each panel of your results figures are clearly referenced in the text.
- Avoid sloppiness. Don't let a table flow over the margin. Try to avoid different fonts and font sizes between figures.
- Colors: choose them well. Try changing default colors and removing grids from plots, etc.

Information to Include in Results

- **Baselines:** How were the baseline methods used? Did you use default parameters?
- (In real life...) you should be testing your method on several datasets (3 in biology is good).
- **Dataset description:** Describe these datasets, any pre-processing you did, and where the information can be accessed.
- **Description of Experiments:** Experiments need to be clearly described, including small details like the number of times you repeated such experiment. Always reference the figure or table where the results appear wrt a given experiment.

Discussion

- Recap what you have done with an overall summary
- Explain how your work complements or addresses some unmet need in the field
- Summarize your results again
- Discuss limitations and future work
- **Inspirational Parting Thought:** What is the main reason people should care and why does your work advance the field?

Publishing in Comp Bio

- Conferences
  - ISMB
  - RECOMB
  - ACM BCB
- Journals
  - Bioinformatics
  - Cell Systems
  - Nature Journals (Nature Methods, Nature Biotech, Nature Communications)

Writing a Conference Paper

- Self-contained, well-structured, making it easy to read and write
- Much faster in terms of review, revision
- Appealing to CS audience.

Writing a Journal Article

- The main text is selling an algorithm to a broad audience.
- Heavily relies on supplemental text to get all of the relevant details.
- Very slow process. From initial submission to publication can take 1 year.
- Not as appealing to a CS audience.
- More appealing to biology audience.
- Very expensive to publish

Providing Code

- It is good to provide code with your paper starting at the time of submission
- Repository should contain a pre-processed version of the data and instructions about how to run code on these data.

From the Point of View of a Paper Consumer

- It is great to publish in fancy interdisciplinary journals
- It becomes less valuable to us on the CS side if the method is scattered over 100 pages of supplement
- Writing a version of your paper with all of the technical details for ArXiv is very good practice.

A Word of Advice for Being a PhD Student in Comp Bio

Protect your expertise and your time. You are not a core facility.

- Prioritize collaborations that are mutually beneficial
- Make sure you publish your own papers without too many distractions of analyzing random datasets.
- Check where your potential collaborators put their comp bio people in the author list.

Communicating Between Fields

- People will care about different things, between biology and computer science- tailor your details accordingly.
- You need to translate your complex model to a series of steps that don't involve mathematical phrases that we all take for granted. For example, don't say phrases like 'L1 penalty'

Choosing What to Work On

Inspired by the talk of Quaid Morris
`https://www.youtube.com/watch?v=xueh6WnpRDQ`

- Don't be the state-of-the-art, be the benchmark (aka ask a new question)
- Choose hard problems rooted in biology that other people wouldn't have thought to ask because they don't read the biological literature.
- Watch the superstars who speak both languages. Watch how they publish and what they choose to work on.

Transitioning and Summarizing What we Have Covered

We have focused on representing data as graphs and using the graphs to help us to answer questions.

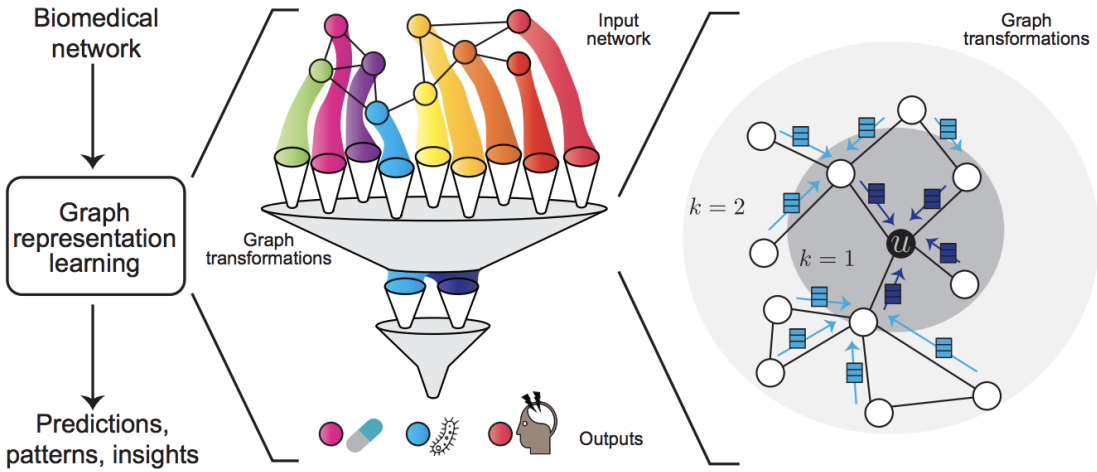
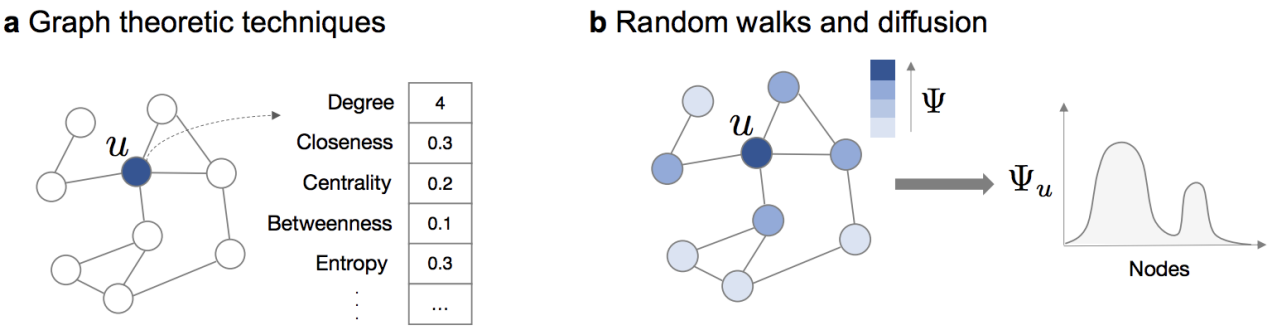


Figure: From `https://arxiv.org/abs/2104.04883`. For example. Assigning proteins to groups or people to outcomes.

Class 1: Graph Summary Statistics and Diffusion

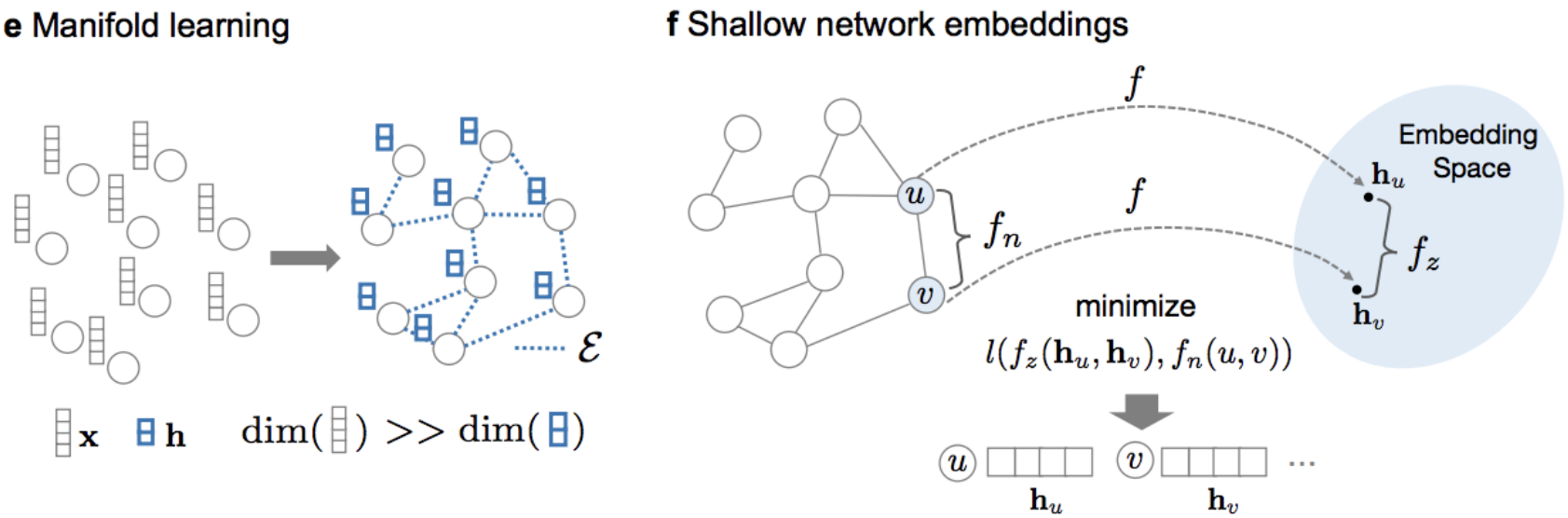
Summary statistics and diffusion can describe patterns in the graph, importance of nodes,



Graph Structure and Diffusion and Papers

- **PhenoGraph:** Partition cells to cell clusters
- **BigClam:** For overlapping clustering
- **MAGIC:** for imputation in single cell data.
- **MELD:** for predicting the specificity of each cell to each condition.
- **Conos:** Combining multiple single cell datasets
- **REGAL:** graph alignment based on structural properties

Node Embedding Theme



Class 2: Node Embedding Theme

- Node2Vec for node embedding (embedding)
- SUGAR for data augmentation in single-cell analysis (manifold)
- SLICER for trajectory inference (manifold)
- Grassmann Embedding for combining multiple datasets (manifold)
- Mashup for embedding nodes according to multiple relational definitions (embedding)

Class 3: Machine Learning on Graphs

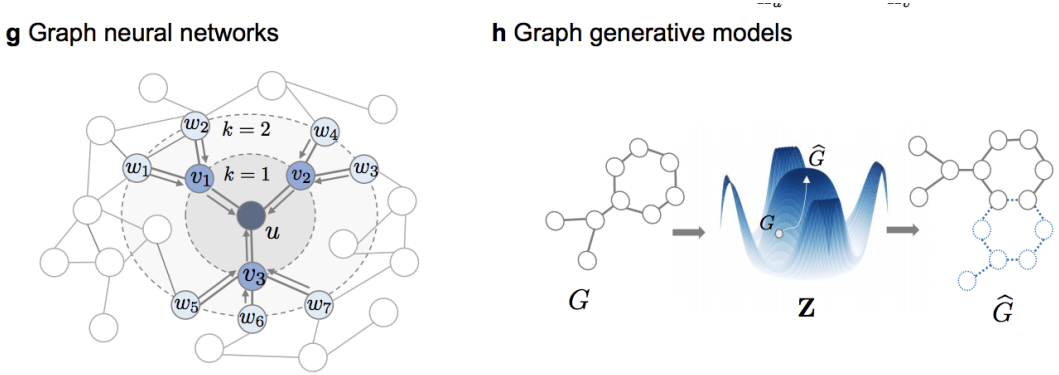


Figure: We haven't seen so much here.....

Seen in ML on Graphs

- Correct and Smooth for predicting labels of nodes based on simple base predictor for node features.