Manifold Learning and Artificial Intelligence Lecture 8

A Path to Protein Design and Disease Mechanism (1)

Protein Language Model

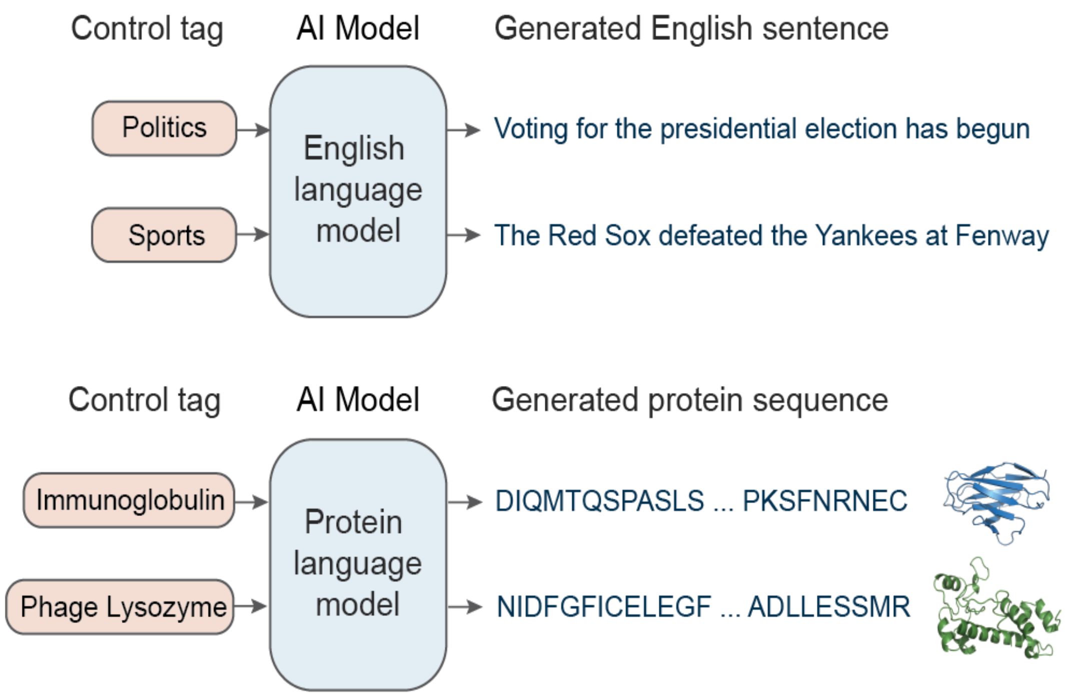
Momiao Xiong, University of Texas School of Public Health

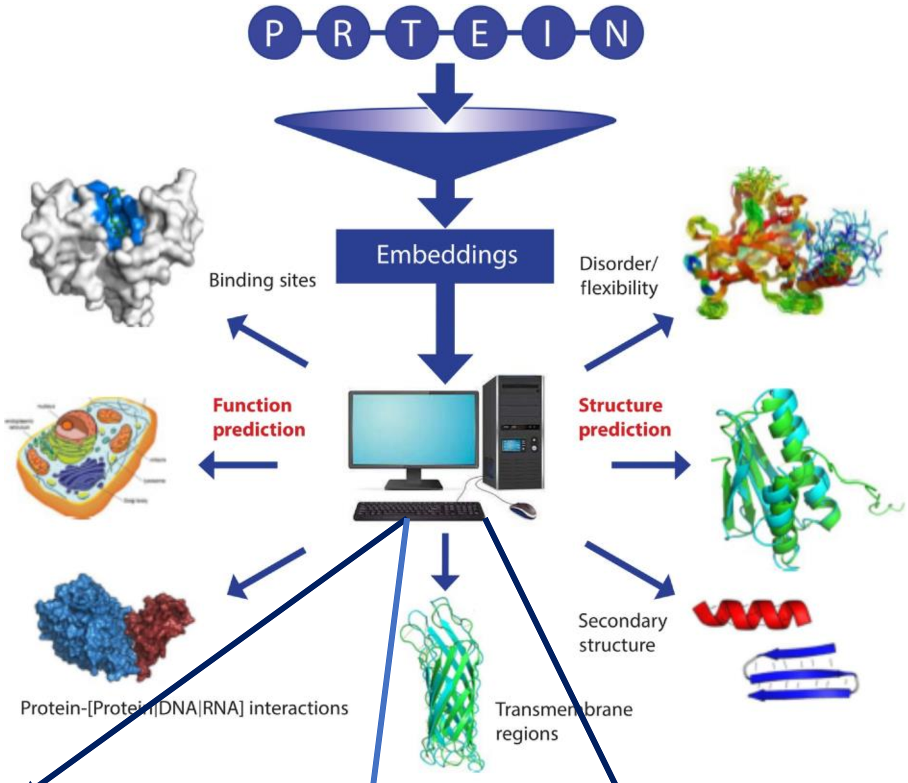
- Time: 9:00 pm, US East Time, 01/14/2023
- 10:00 am, Beijing Time. 01/15/2023
- Zoom

https://uwmadison.zoom.us/w/93316139423?tk=wfbmsTfN2fgERto_HI1WKtBzh94d3HO02XV
Cexqd8.DQMAAAAVuhNJnxZ2dGVCblllYlR3ZWJBNHl5LXlYMjBnAAAAAAAAAAAAAAAAAAAAAA
AAAAAA&pwd=Q0NVWFYvRFg5RmxCNkwxMmYrbW41dz09

Github Address: https://ai2healthcare.github.io/

Introduction

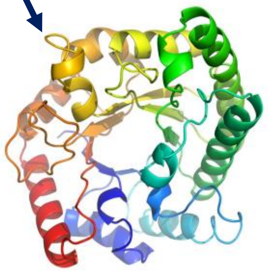

Control tag
AI Model
Generated English sentence
Politics
Sports
English language model
Voting for the presidential election has begun
The Red Sox defeated the Yankees at Fenway
Control tag
AI Model
Generated protein sequence
Immunoglobulin
Phage Lysozyme
Protein language model
DIQMTQSPASLS ... PKSFNRNEC
NIDFGFICELEGF ... ADLLESSMR



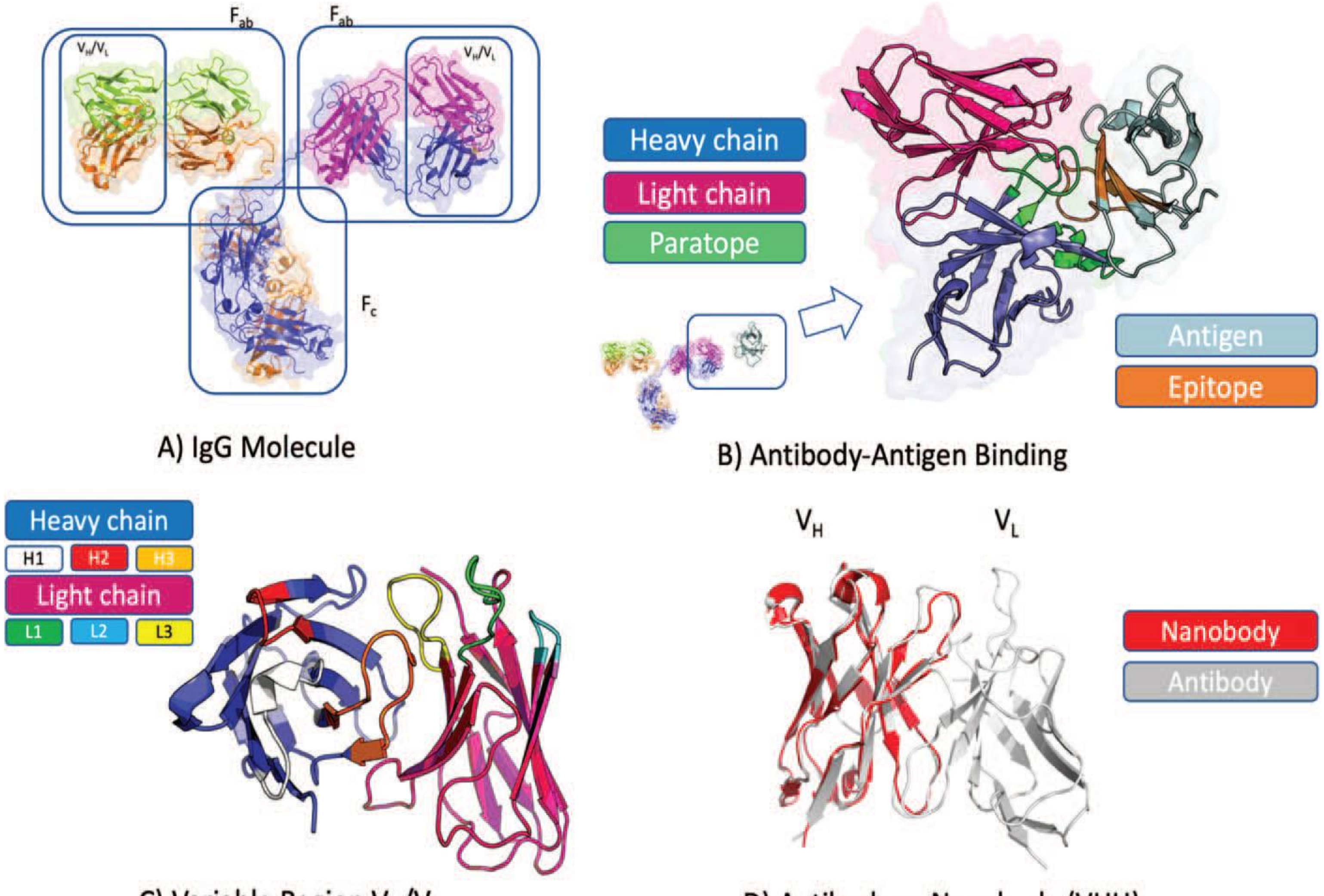
Trends in Biochemical Sciences

Function Annotation
Stability Prediction
Fitness Prediction

MASLSCV



Antibody



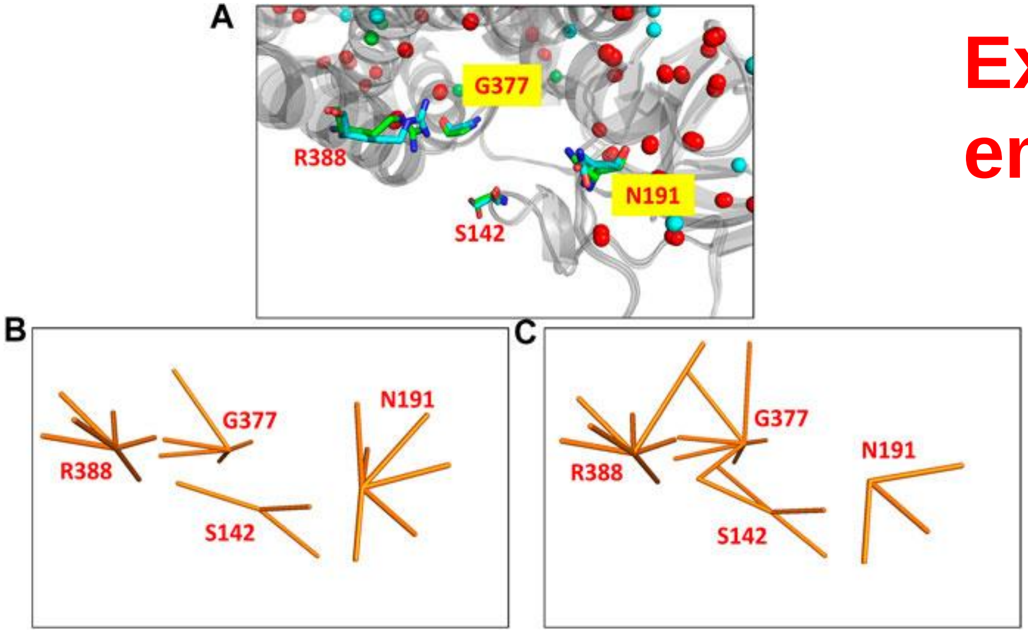
A) IgG Molecule

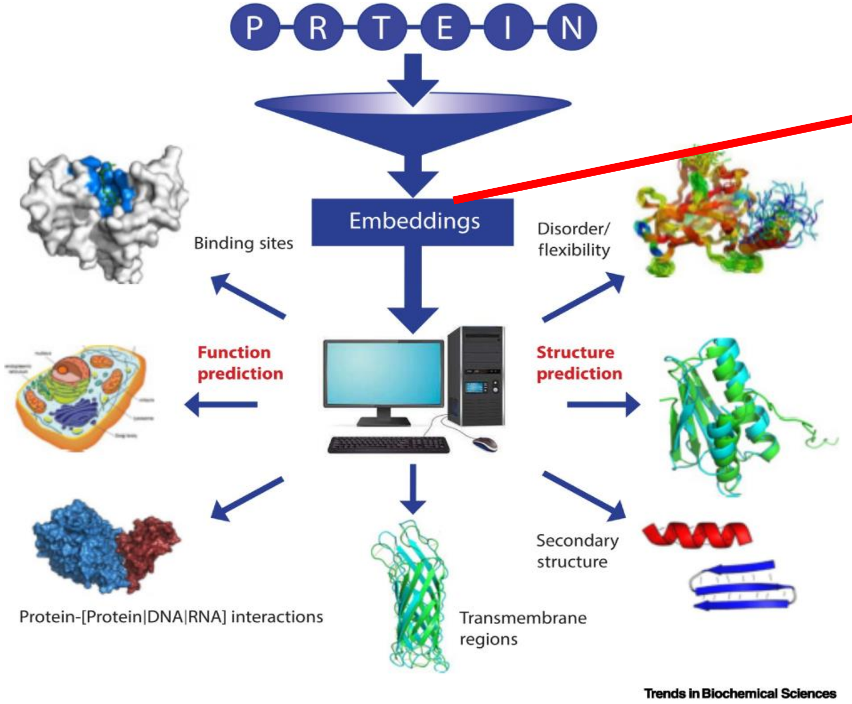
B) Antibody-Antigen Binding

C) Variable Region V_H/V_L

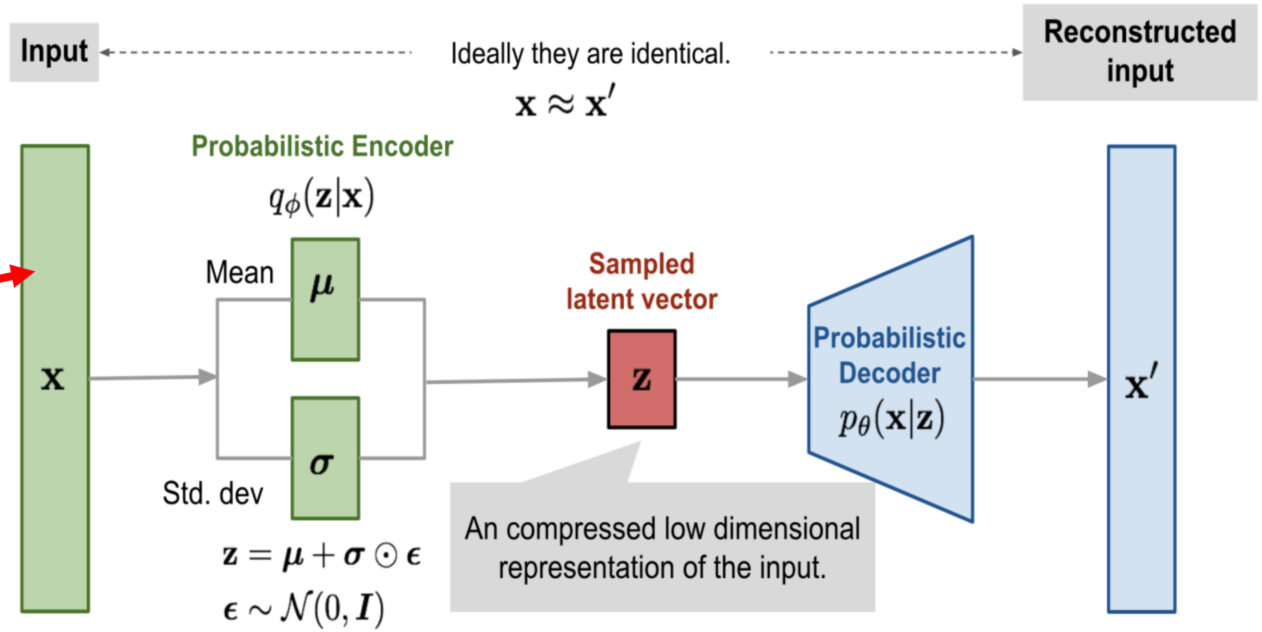
D) Antibody vs Nanobody (VHH)

Extract Protein Variation into embedding and reduce it via VAE





Trends in Biochemical Sciences



Ideally they are identical.

$\mathbf{x} \approx \mathbf{x}'$

$q_\phi(\mathbf{z}|\mathbf{x})$

$\boldsymbol{\mu}$, $\boldsymbol{\sigma}$, $\mathbf{x}$, $\mathbf{z}$, $\mathbf{x}'$

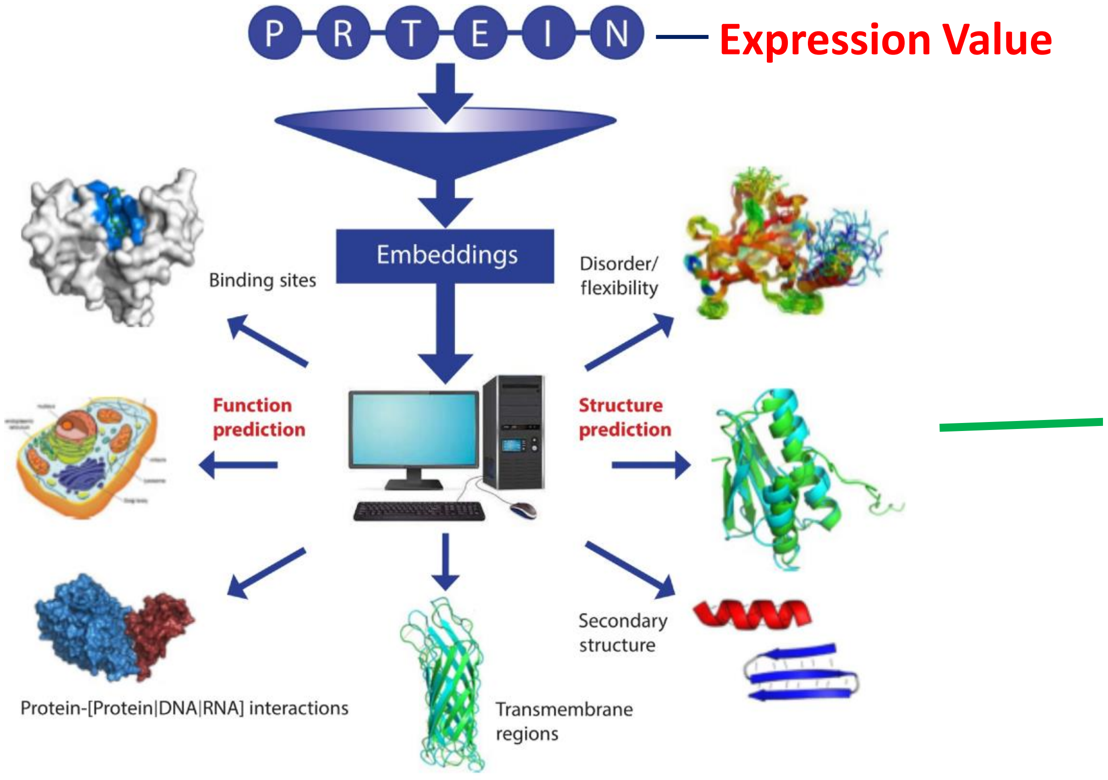
$p_\theta(\mathbf{x}|\mathbf{z})$

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$$

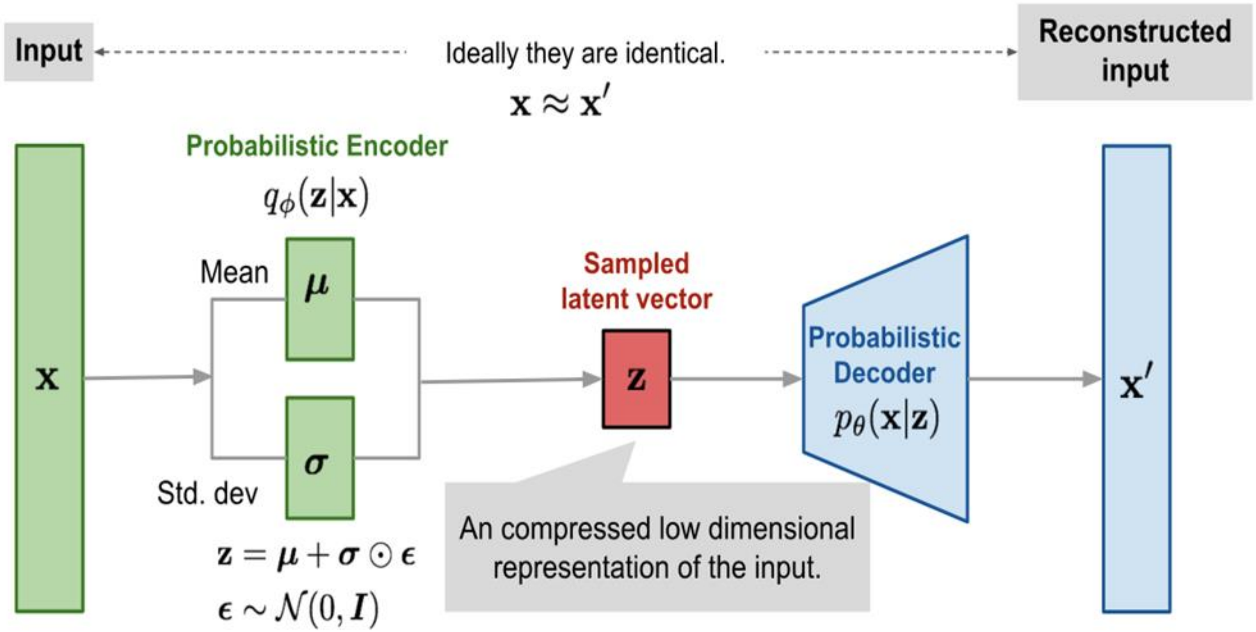
$$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$$

An compressed low dimensional representation of the input.

Joint Amino Acid Variation and Expression Embedding

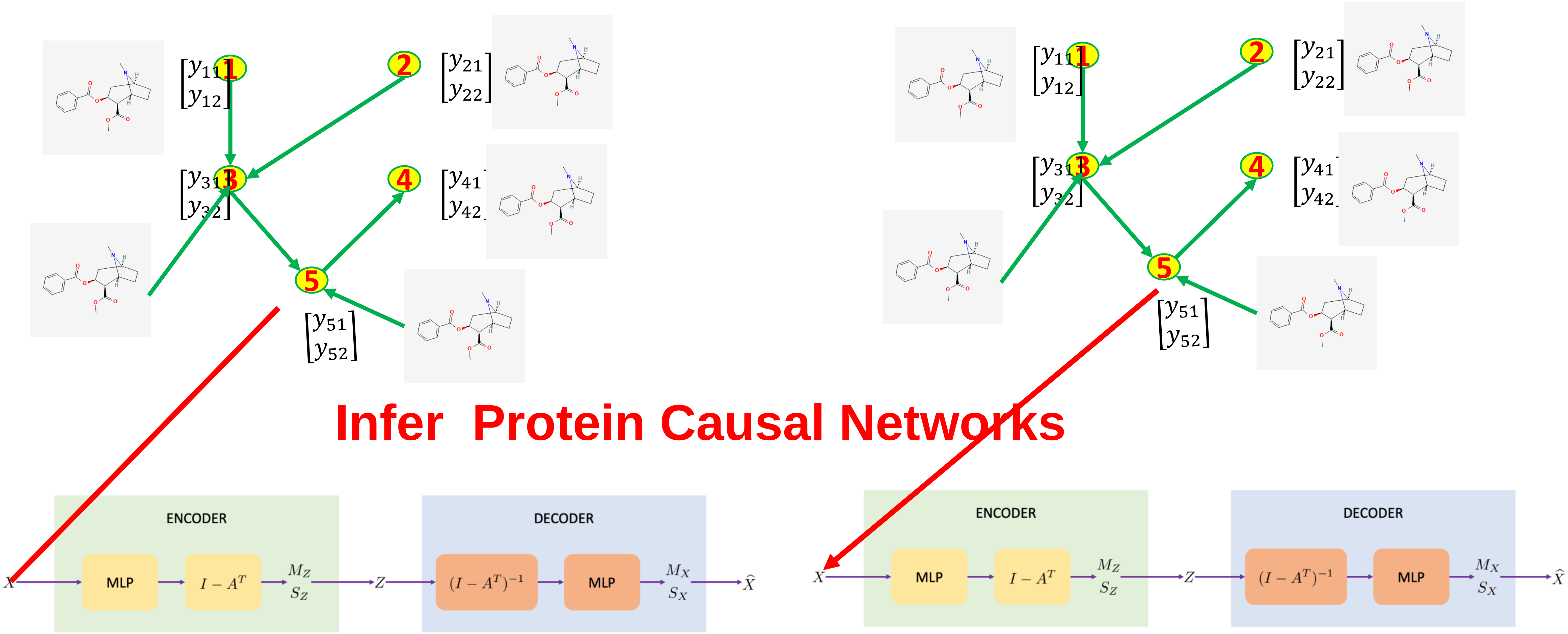


Trends in Biochemical Sciences


Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
Mean
$\boldsymbol{\mu}$
$\boldsymbol{\sigma}$
Std. dev
$\mathbf{x}$
Sampled latent vector
$\mathbf{z}$
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \epsilon$
$\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$
An compressed low dimensional representation of the input.

Normal

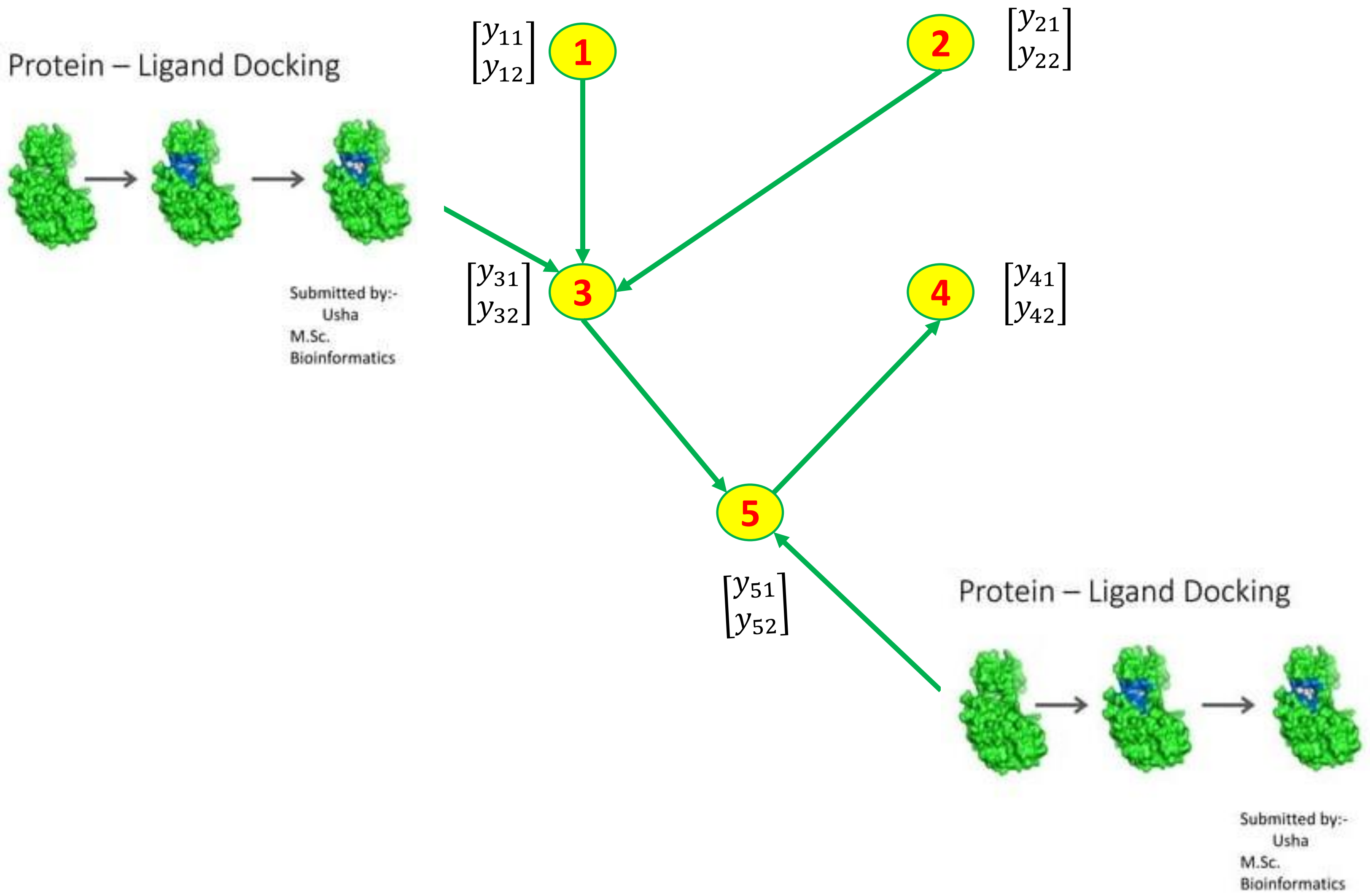
Disease



1
2
3
4
5
$\begin{bmatrix} y_{11} \\ y_{12} \end{bmatrix}$
$\begin{bmatrix} y_{21} \\ y_{22} \end{bmatrix}$
$\begin{bmatrix} y_{31} \\ y_{32} \end{bmatrix}$
$\begin{bmatrix} y_{41} \\ y_{42} \end{bmatrix}$
$\begin{bmatrix} y_{51} \\ y_{52} \end{bmatrix}$
ENCODER
DECODER
MLP
$I - A^T$
M_Z
S_Z
Z
$(I - A^T)^{-1}$
MLP
M_X
S_X
X
$\hat{X}$

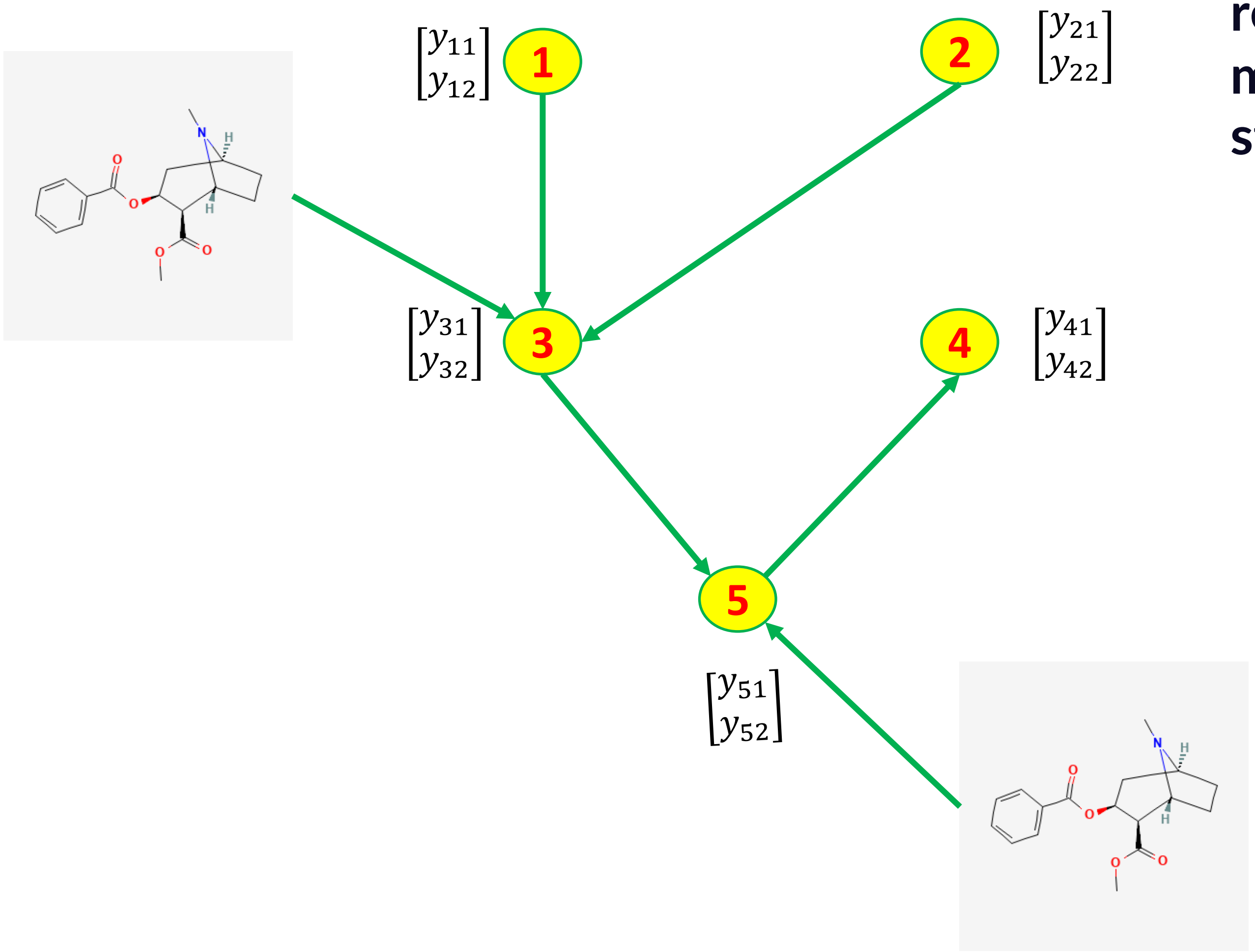
Infer Protein Causal Networks

$$X = A^T X + Z$$

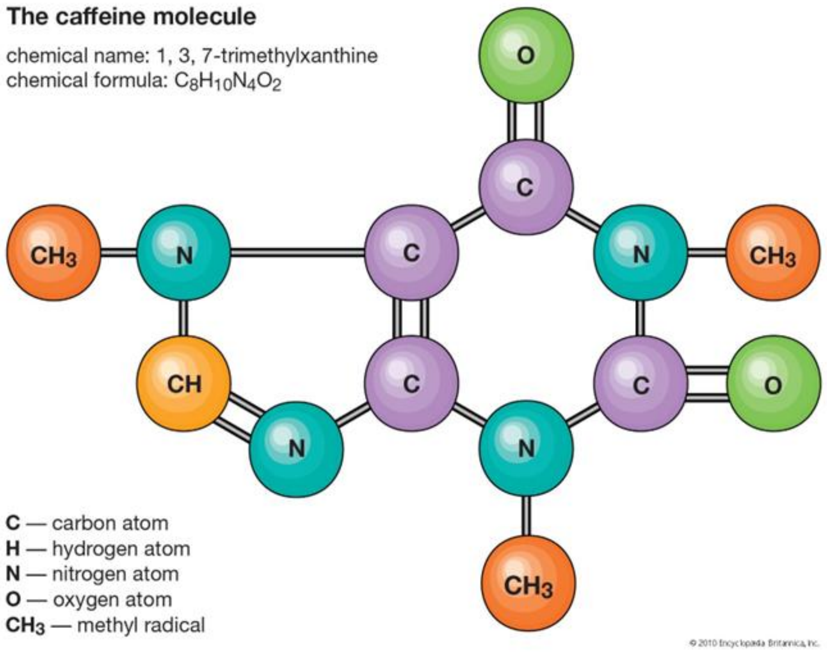

Protein – Ligand Docking
Submitted by:-
Usha
M.Sc.
Bioinformatics
$\begin{bmatrix} y_{11} \\ y_{12} \end{bmatrix}$
1
2
$\begin{bmatrix} y_{21} \\ y_{22} \end{bmatrix}$
$\begin{bmatrix} y_{31} \\ y_{32} \end{bmatrix}$
3
4
$\begin{bmatrix} y_{41} \\ y_{42} \end{bmatrix}$
5
$\begin{bmatrix} y_{51} \\ y_{52} \end{bmatrix}$
Protein – Ligand Docking
Submitted by:-
Usha
M.Sc.
Bioinformatics

Drug Target Causal Networks

Graphic Neural Networks (GNN)


$\begin{bmatrix} y_{11} \\ y_{12} \end{bmatrix}$
1
2
$\begin{bmatrix} y_{21} \\ y_{22} \end{bmatrix}$
$\begin{bmatrix} y_{31} \\ y_{32} \end{bmatrix}$
3
4
$\begin{bmatrix} y_{41} \\ y_{42} \end{bmatrix}$
5
$\begin{bmatrix} y_{51} \\ y_{52} \end{bmatrix}$
N
H
O
O
H
O
O

Probably the most common application of representing data with graphs is using molecular graphs to represent chemical structures



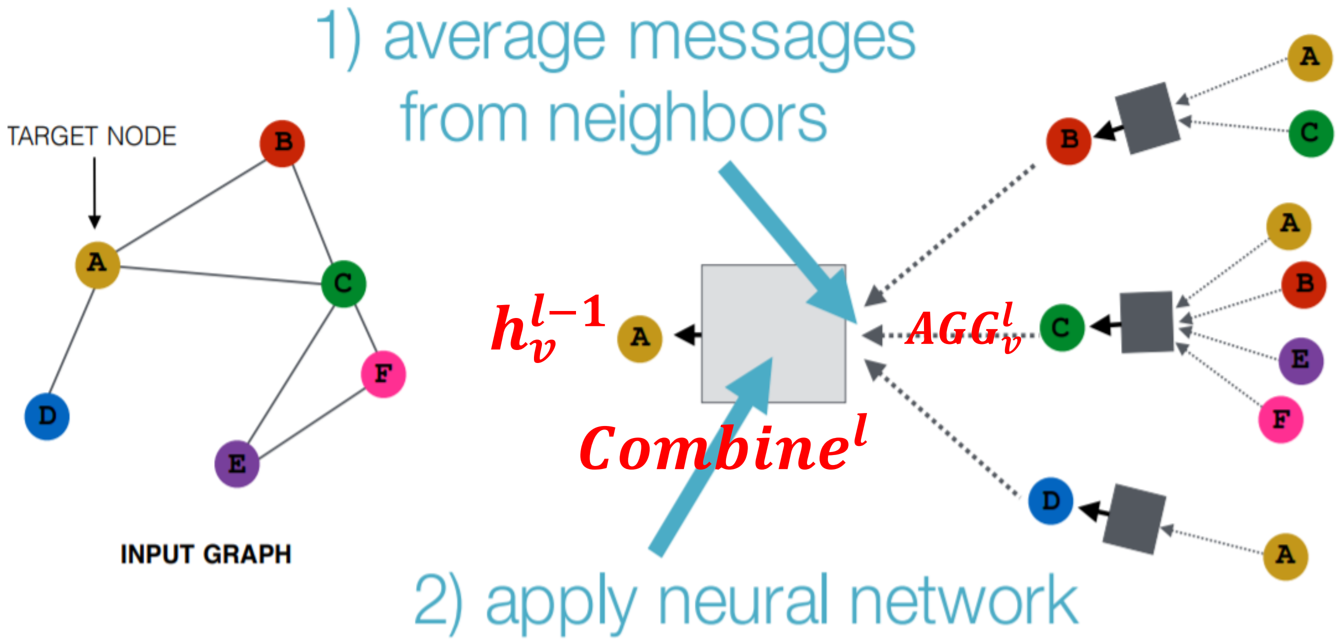
Graphic Neural Network (GNN)

- **Tasks of GNN: Graph level, node level and edge level**
  - **Aggregation**

$$M_{jv}^{l-1} = MSG(h_j^{l-1}, h_v^{l-1}, e_{jv}^{l-1})$$

$$AGG_v^l = AGG^l(\left\{ M_{jv}^{l-1} | j \in \mathcal{N}(v) \right\})$$

$$h_v^l = Combine^l(h_v^{l-1}, AGG_v^l)$$



1) average messages from neighbors

h_v^{l-1}

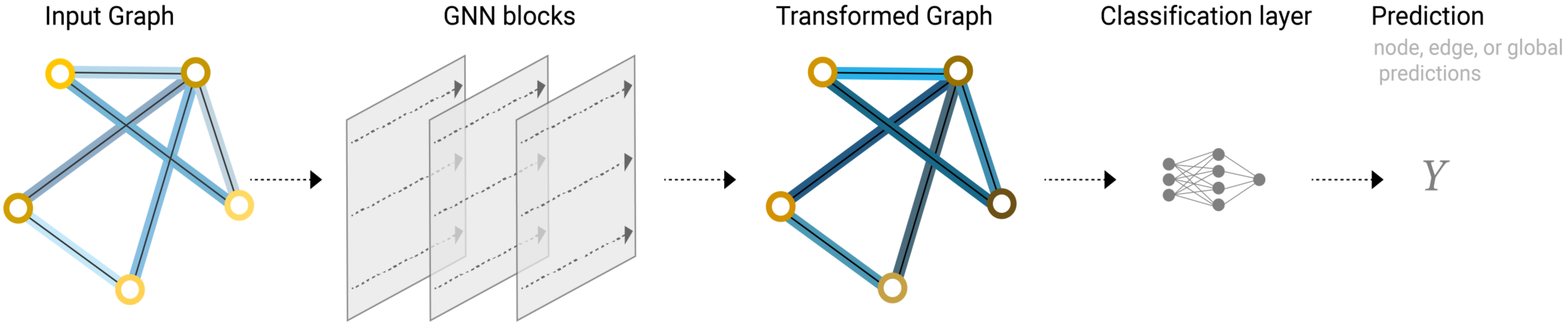
AGG_v^l

$Combine^l$

2) apply neural network

Sharma 2020, Introduction to Graph Neural Networks

Pipelines of Graphic Neural Networks



$$h_G = READOUT(\{h_v^L, v \in \mathcal{V}\})$$

Distill, 2021; A Gentle Introduction to Graph Neural Networks. https://distill.pub/2021/gnn-intro

Directed Acyclic Graph Neural Networks

- **A DAG is a directed graph without cycles**
- **updating node representations based on those of all their predecessors sequentially, such that nodes without successors digest the information of the entire graph**.

$$AGG_v^l = \sum\nolimits_{u \in \mathcal{P}(v)} \alpha_{vu}^l(h_v^{l-1}, h_u^l) h_u^l$$

$$\alpha_{vu}^l\left(h_v^{l-1}, h_u^l\right) = \underset{u \in \mathcal{P}(v)}{\operatorname{softmax}}\left(\left(w_1^l\right)^T h_v^{l-1}\right.$$

$$\left. + \left(w_2^l\right)^T h_u^l + \left(w_3^l\right)^T y(u,v)\right)$$

$$h_v^l = F^l\left(h_v^{l-1}, AGG_v^l\right) = GRU^l\left(h_v^{l-1}, AGG_v^l\right)$$

$$h_G = FC(\underset{v \in \mathcal{T}}{\max - \operatorname{pool}}(\|_0^L\, h_v^l,)$$

$$\| \underset{u \in S}{\max - \operatorname{pool}}(\|_0^L\, \tilde{h}_u^l))$$

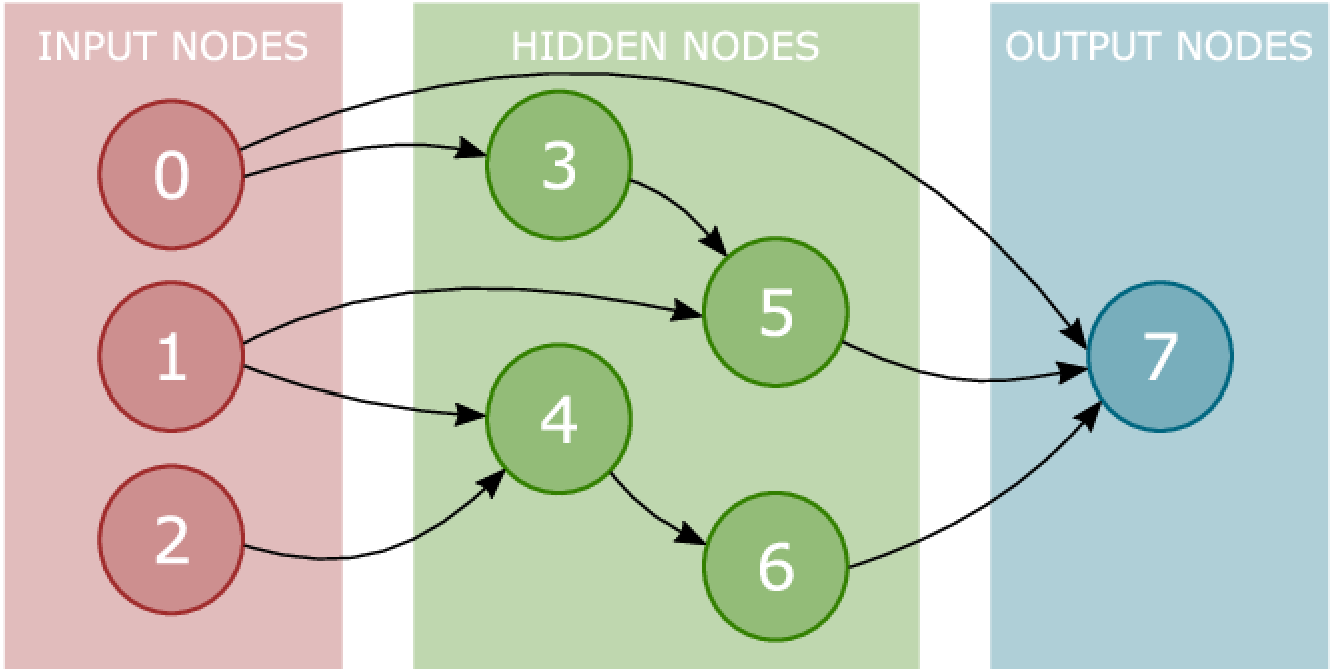
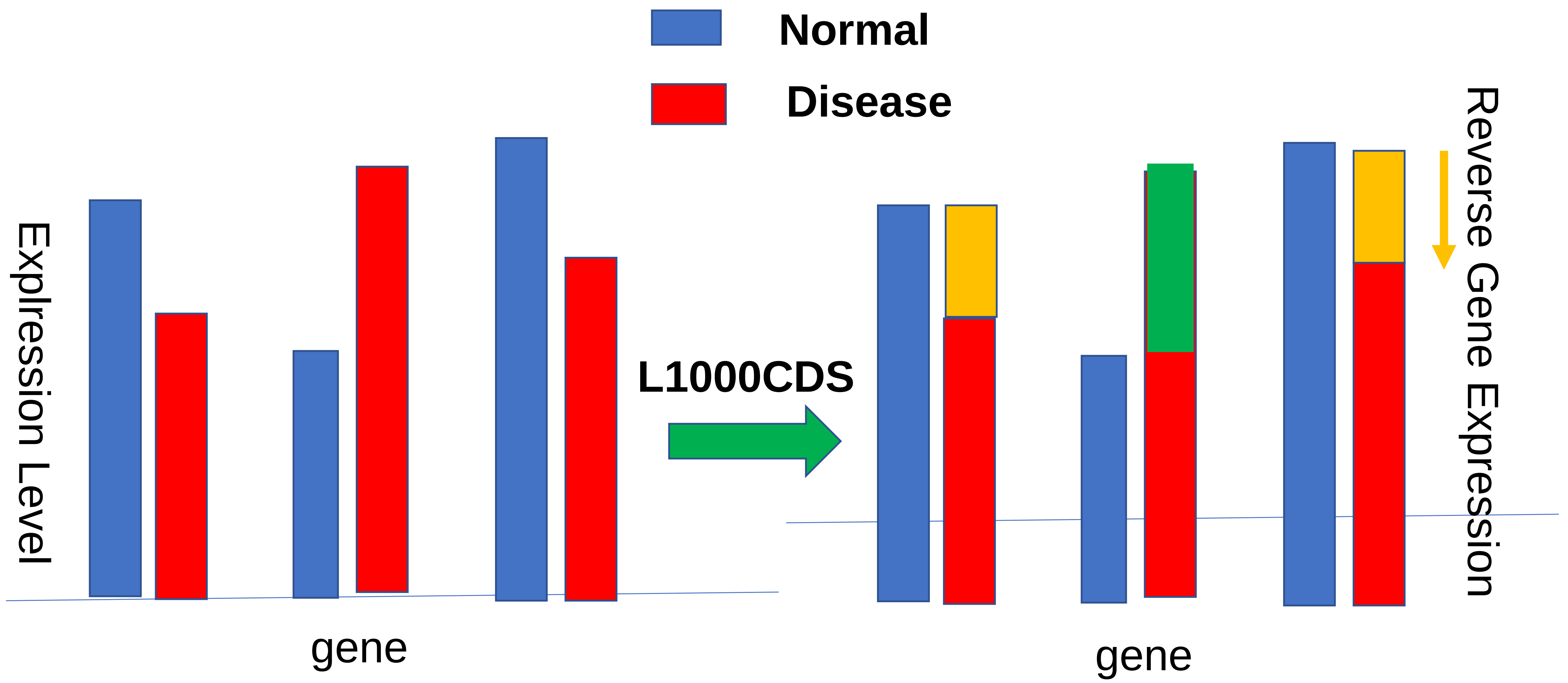


FIGURE 1. **An example phenotype for a Directed Acyclic Graph Neural Network (DAG-NN).**

$$\mathcal{P}(v) = set\ of preceeding\ nodes$$

This also allows producing a single output for the whole graph

Drug efficacy is determined based on the ability to reverse altered gene expression



Embedding Test Statistics

$$T = \left(\bar{Z}_m - \bar{Z}_w\right)^T \Lambda^{-1} \left(\bar{Z}_m - \bar{Z}_w\right), \mathrm{T} \sim \chi^2_{(k)}$$

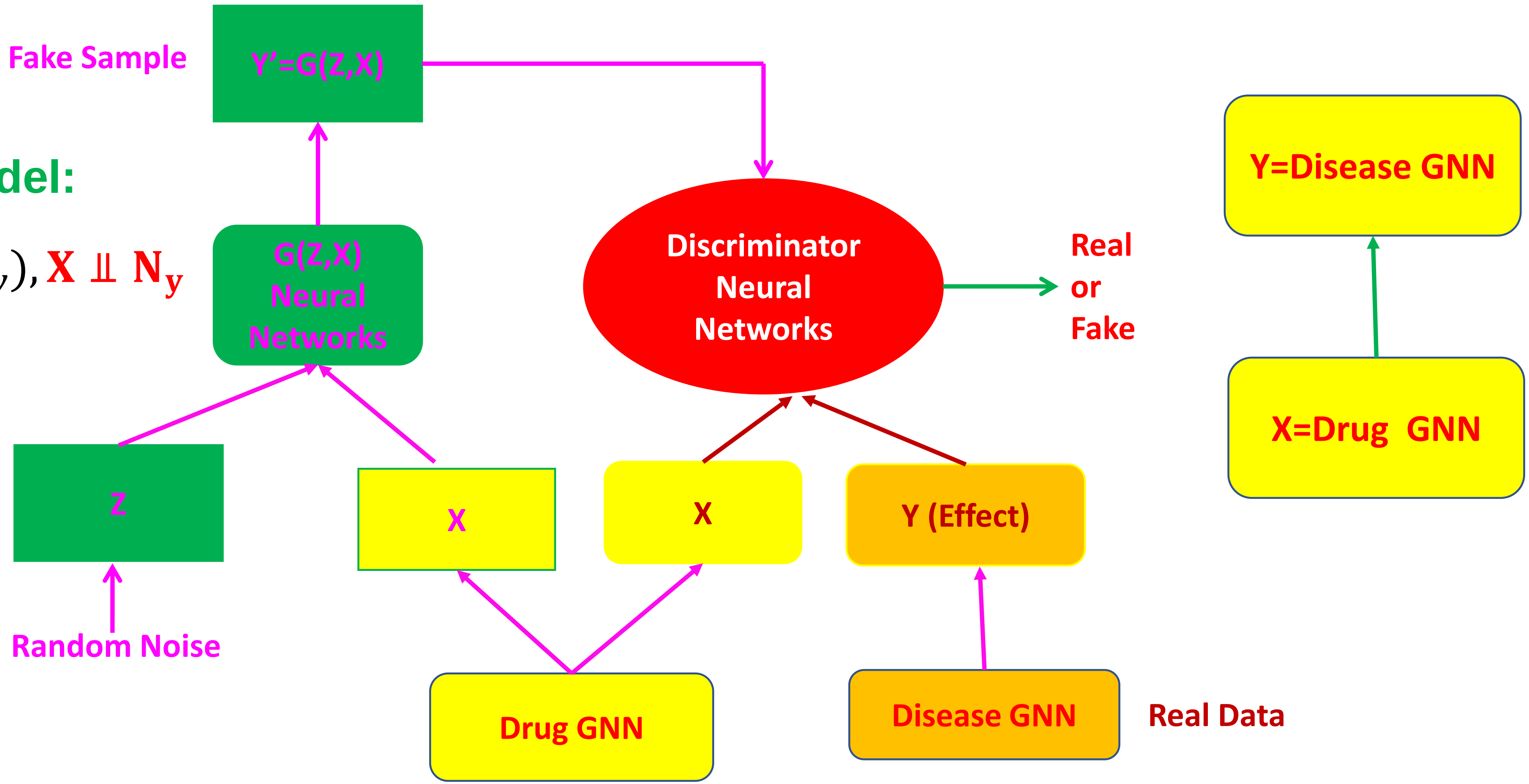
K: dimension of Embedding

Causal Test for Embedding

Test Causal Relationship between Drug and Disease GNNs

Causal Model:

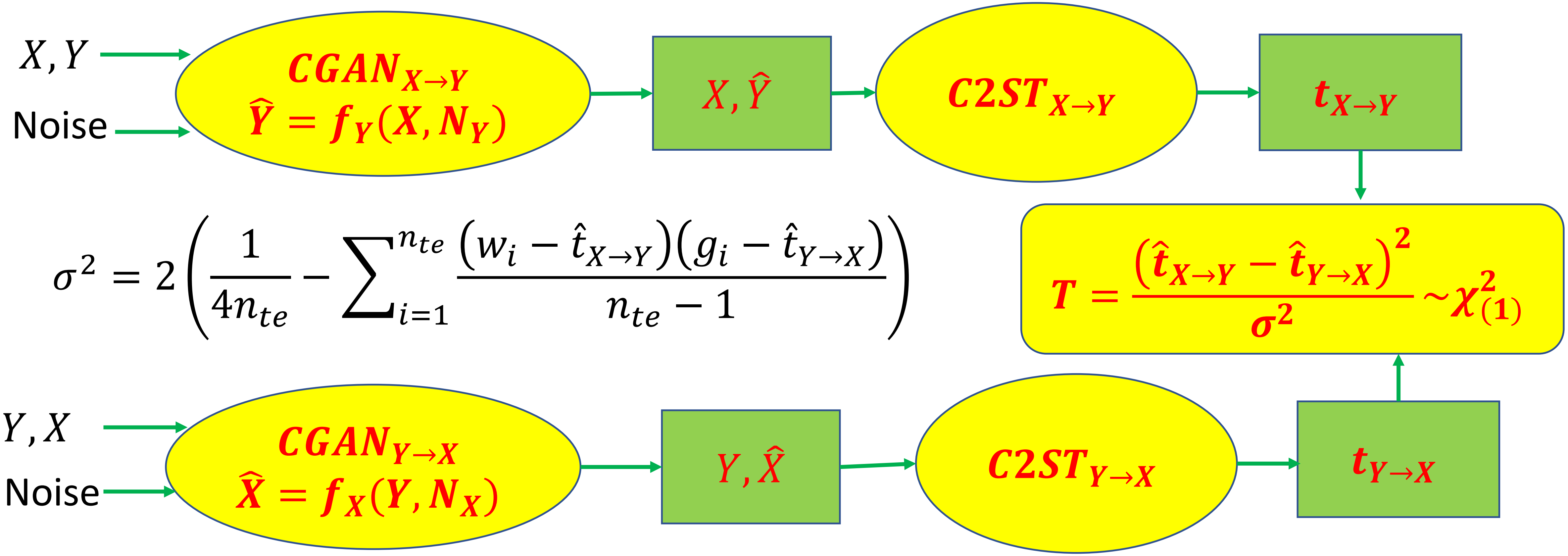
$$\mathbf{Y} = \mathbf{f_y}(\mathbf{X}, \boldsymbol{N_y}), \mathbf{X} \perp\!\!\!\perp \mathbf{N_y}$$



Test for Causation between Drug and Disease Networks

- **Define**

Y =output of Disease GNN,
X =output of drug GNN



X, Y

Noise

$CGAN_{X\to Y}$

$\hat{Y} = f_Y(X, N_Y)$

$X, \hat{Y}$

$C2ST_{X\to Y}$

$t_{X\to Y}$

$$\sigma^2 = 2\left(\frac{1}{4n_{te}} - \sum_{i=1}^{n_{te}} \frac{(w_i - \hat{t}_{X\to Y})(g_i - \hat{t}_{Y\to X})}{n_{te} - 1}\right)$$

$$T = \frac{\left(\hat{t}_{X\to Y} - \hat{t}_{Y\to X}\right)^2}{\sigma^2} \sim \chi^2_{(1)}$$

Y, X

Noise

$CGAN_{Y\to X}$

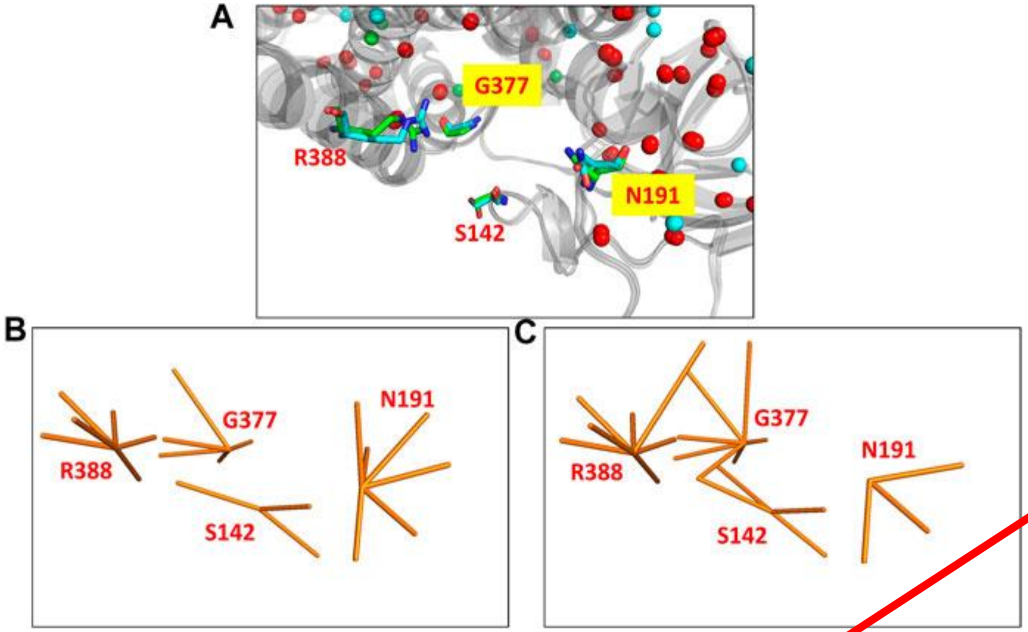
$\hat{X} = f_X(Y, N_X)$

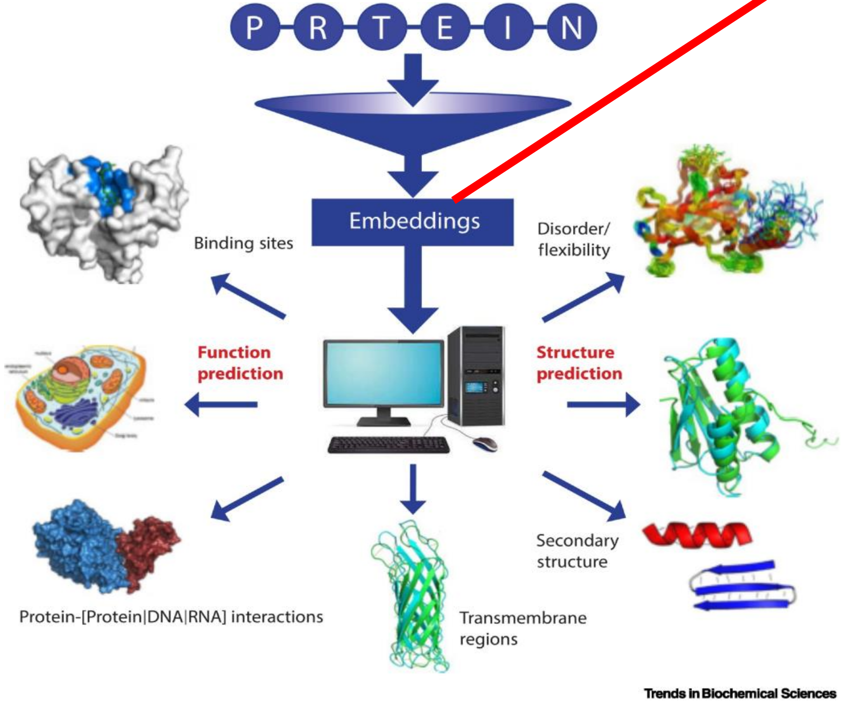
$Y, \hat{X}$

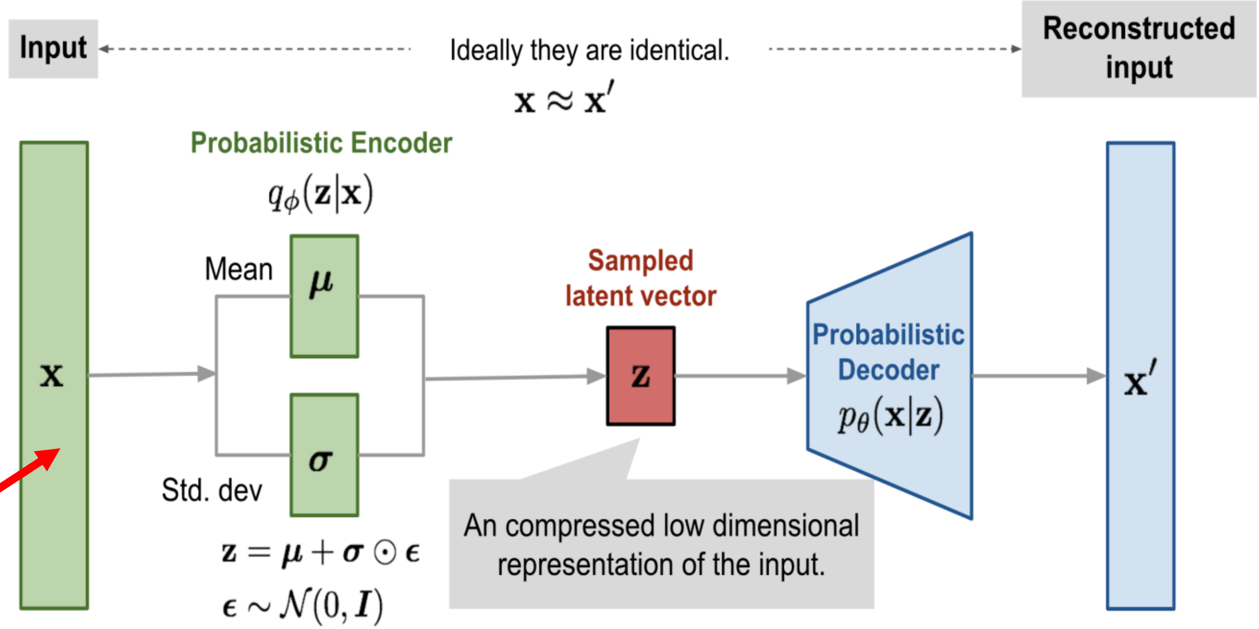
$C2ST_{Y\to X}$

$t_{Y\to X}$

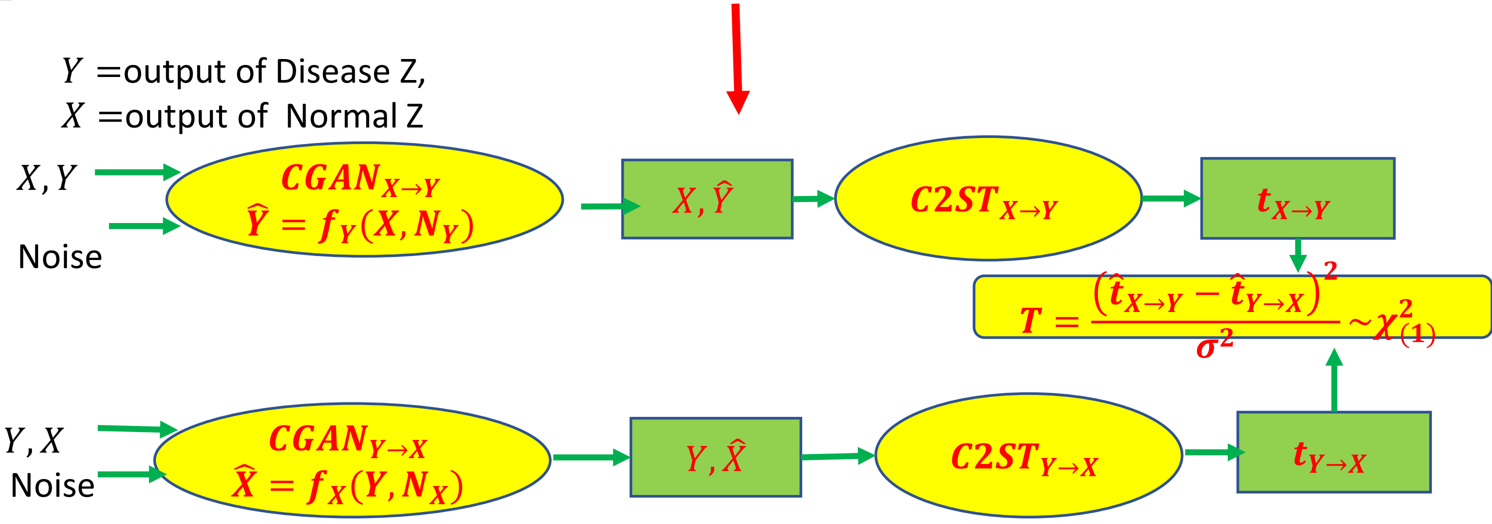
Xiong MM (2022) Artificial Intelligence and Causal Inference. CRC Press






Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
Mean
$\boldsymbol{\mu}$
Std. dev
$\boldsymbol{\sigma}$
$\mathbf{x}$
Sampled latent vector
$\mathbf{z}$
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$
An compressed low dimensional representation of the input.

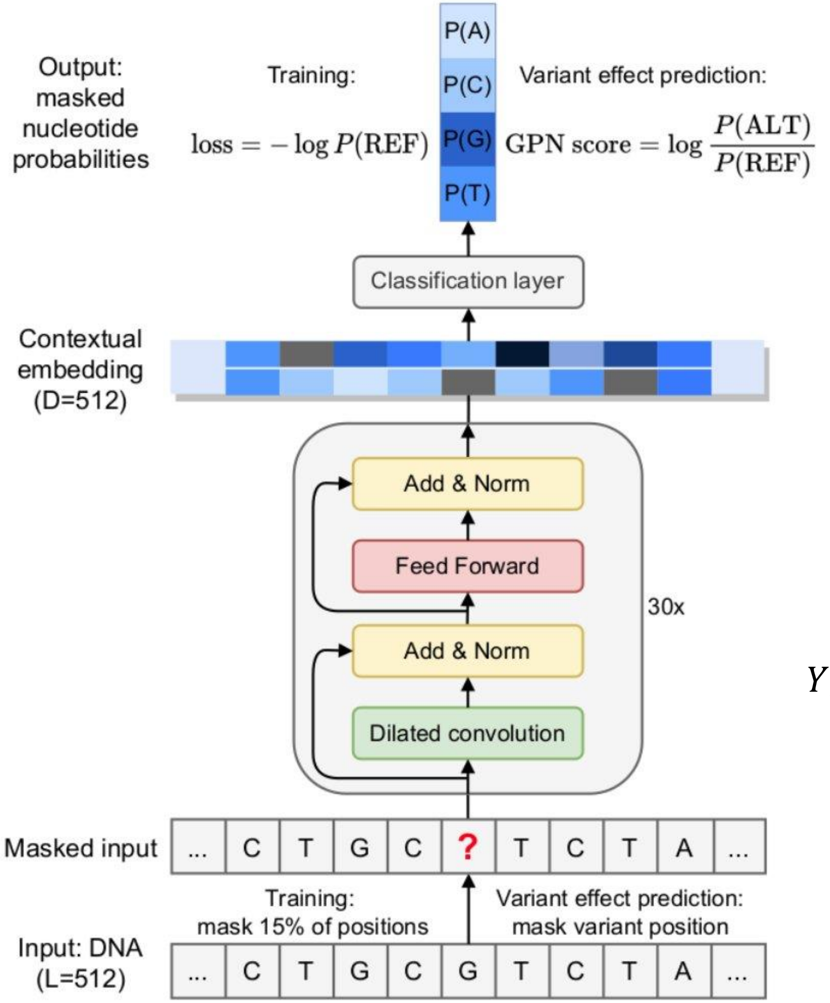
Y =output of Disease Z,
X =output of Normal Z



Test Causation of Protein on Disease (or Trait)

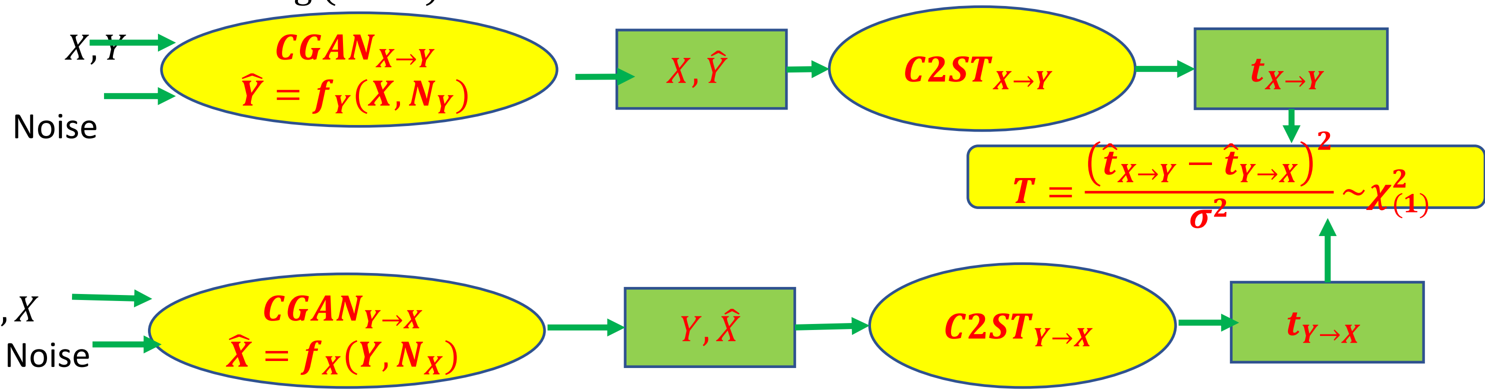
Test Causation of Gene with Trait

GPN (Genomic Pre-trained Network)


Output: masked nucleotide probabilities
Training:
$\text{loss} = -\log P(\text{REF})$
P(A)
P(C)
P(G)
P(T)
Variant effect prediction:
$\text{GPN score} = \log \frac{P(\text{ALT})}{P(\text{REF})}$
Classification layer
Contextual embedding (D=512)
Add & Norm
Feed Forward
Add & Norm
Dilated convolution
30x
Masked input
... C T G C ? T C T A ...
Training: mask 15% of positions
Variant effect prediction: mask variant position
Input: DNA (L=512)
... C T G C G T C T A ...

Y = Trait,

X = Embedding (DNAs)



X, Y

Noise

$CGAN_{X\to Y}$

$\hat{Y} = f_Y(X, N_Y)$

$X, \hat{Y}$

$C2ST_{X\to Y}$

$t_{X\to Y}$

$T = \frac{(\hat{t}_{X\to Y} - \hat{t}_{Y\to X})^2}{\sigma^2} \sim \chi^2_{(1)}$

Y, X

Noise

$CGAN_{Y\to X}$

$\hat{X} = f_X(Y, N_X)$

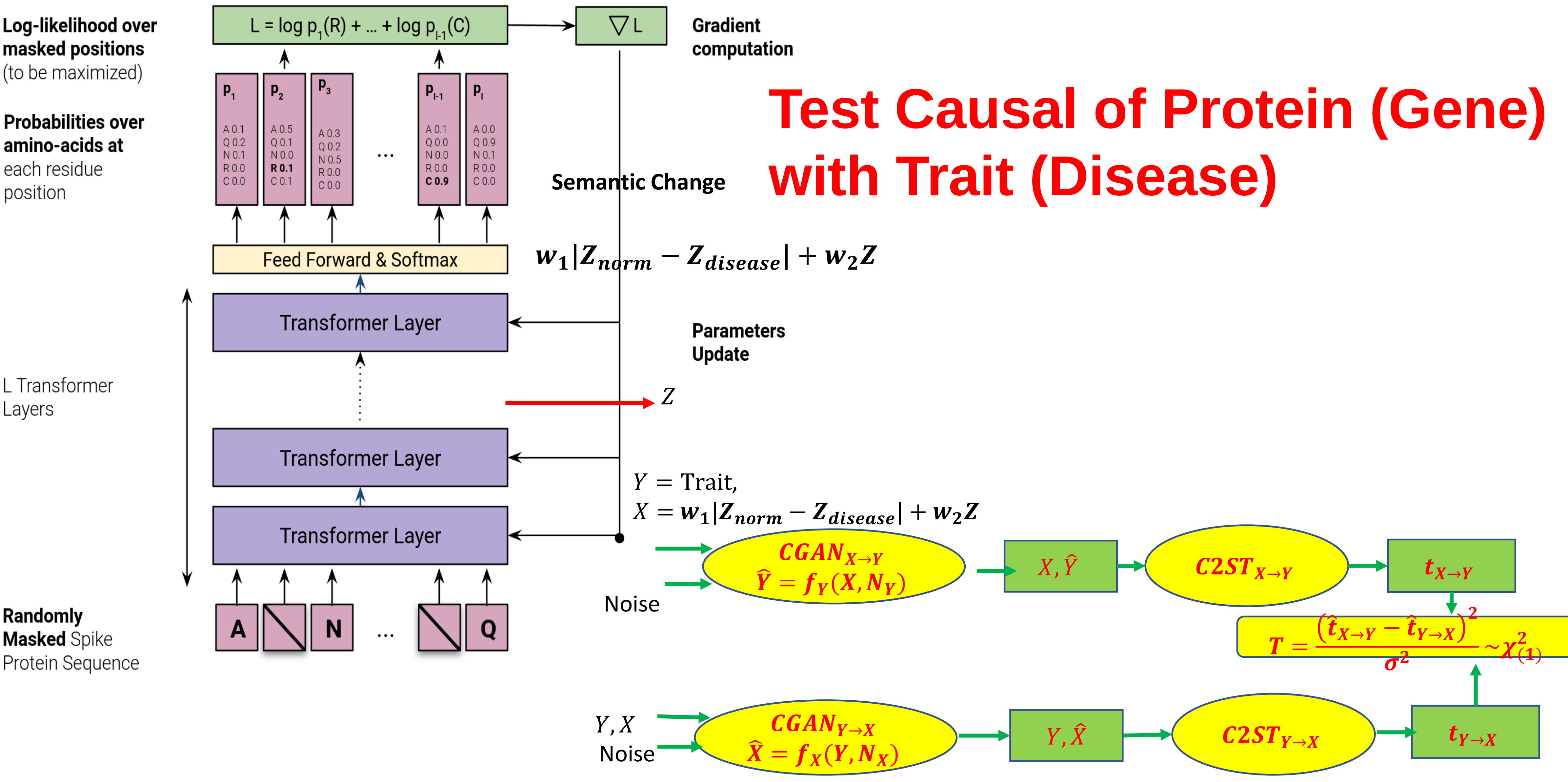
$Y, \hat{X}$

$C2ST_{Y\to X}$

$t_{Y\to X}$

Test Causal of Protein (Gene) with Trait (Disease)

A. Training



Log-likelihood over masked positions (to be maximized)

L = log p_1(R) + ... + log p_{l-1}(C)

∇ L

Gradient computation

Probabilities over amino-acids at each residue position

p_1: A 0.1, Q 0.2, N 0.1, R 0.0, C 0.0

p_2: A 0.5, Q 0.1, N 0.0, **R 0.1**, C 0.1

p_3: A 0.3, Q 0.2, N 0.5, R 0.0, C 0.0

...

p_{l-1}: A 0.1, Q 0.0, N 0.0, R 0.0, **C 0.9**

p_l: A 0.0, Q 0.9, N 0.1, R 0.0, C 0.0

Semantic Change

$$w_1|Z_{norm} - Z_{disease}| + w_2 Z$$

Feed Forward & Softmax

Transformer Layer

Parameters Update

L Transformer Layers

Z

Transformer Layer

Transformer Layer

Randomly Masked Spike Protein Sequence

A, N, ..., Q

Y = Trait,

$X = w_1|Z_{norm} - Z_{disease}| + w_2 Z$

Noise

$CGAN_{X\to Y}$

$\hat{Y} = f_Y(X, N_Y)$

$X, \hat{Y}$

$C2ST_{X\to Y}$

$t_{X\to Y}$

$$T = \frac{(\hat{t}_{X\to Y} - \hat{t}_{Y\to X})^2}{\sigma^2} \sim \chi^2_{(1)}$$

Y, X

Noise

$CGAN_{Y\to X}$

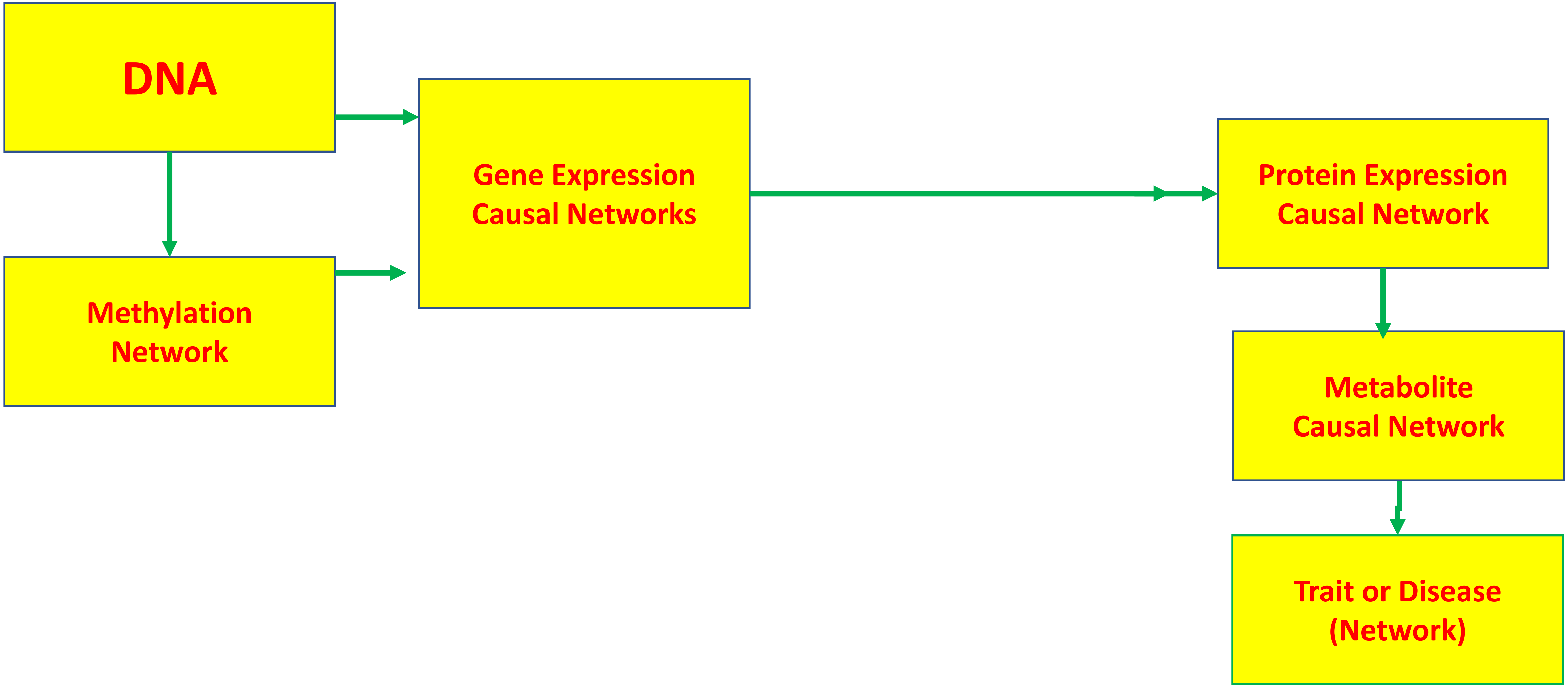
$\hat{X} = f_X(Y, N_X)$

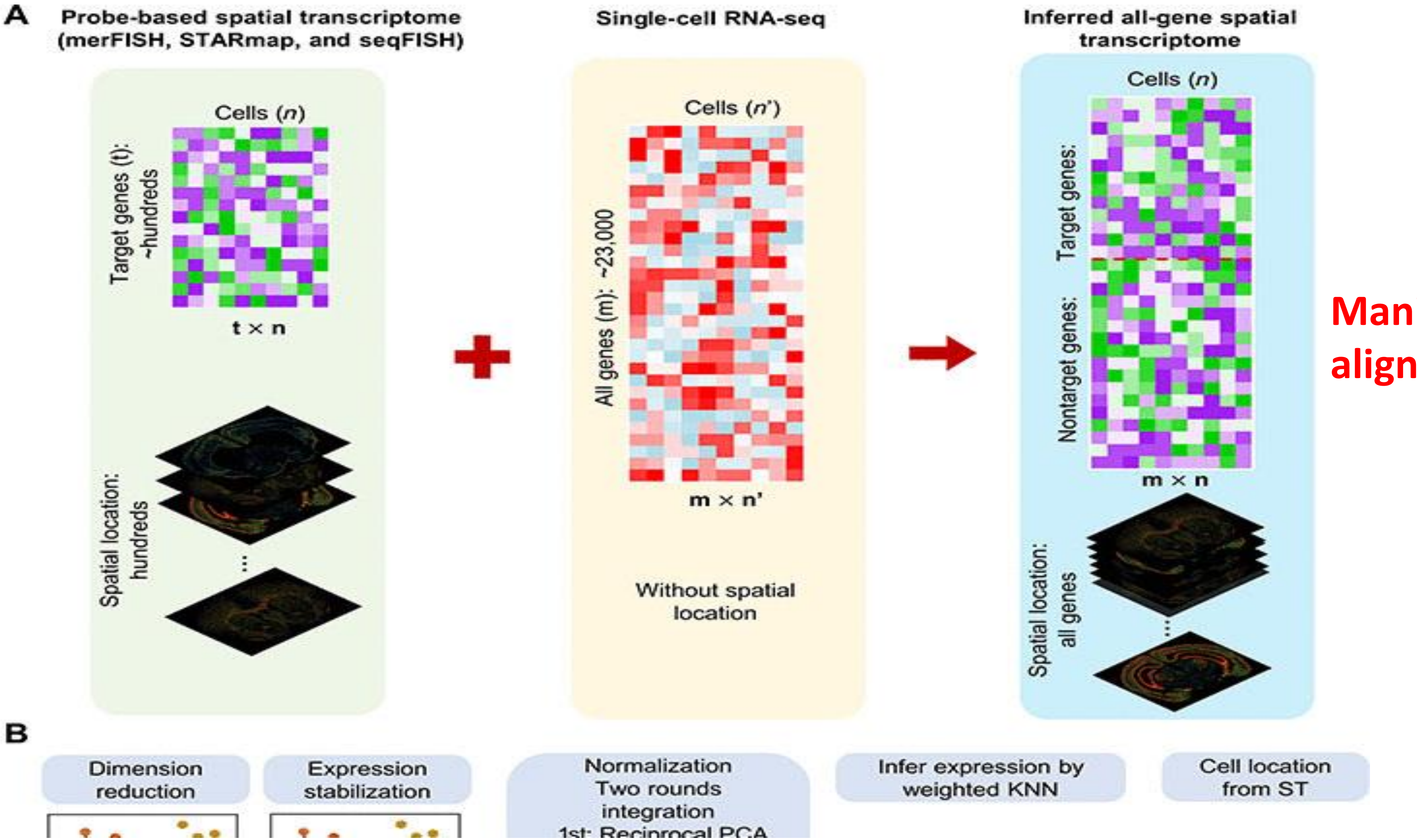
$Y, \hat{X}$

$C2ST_{Y\to X}$

$t_{Y\to X}$

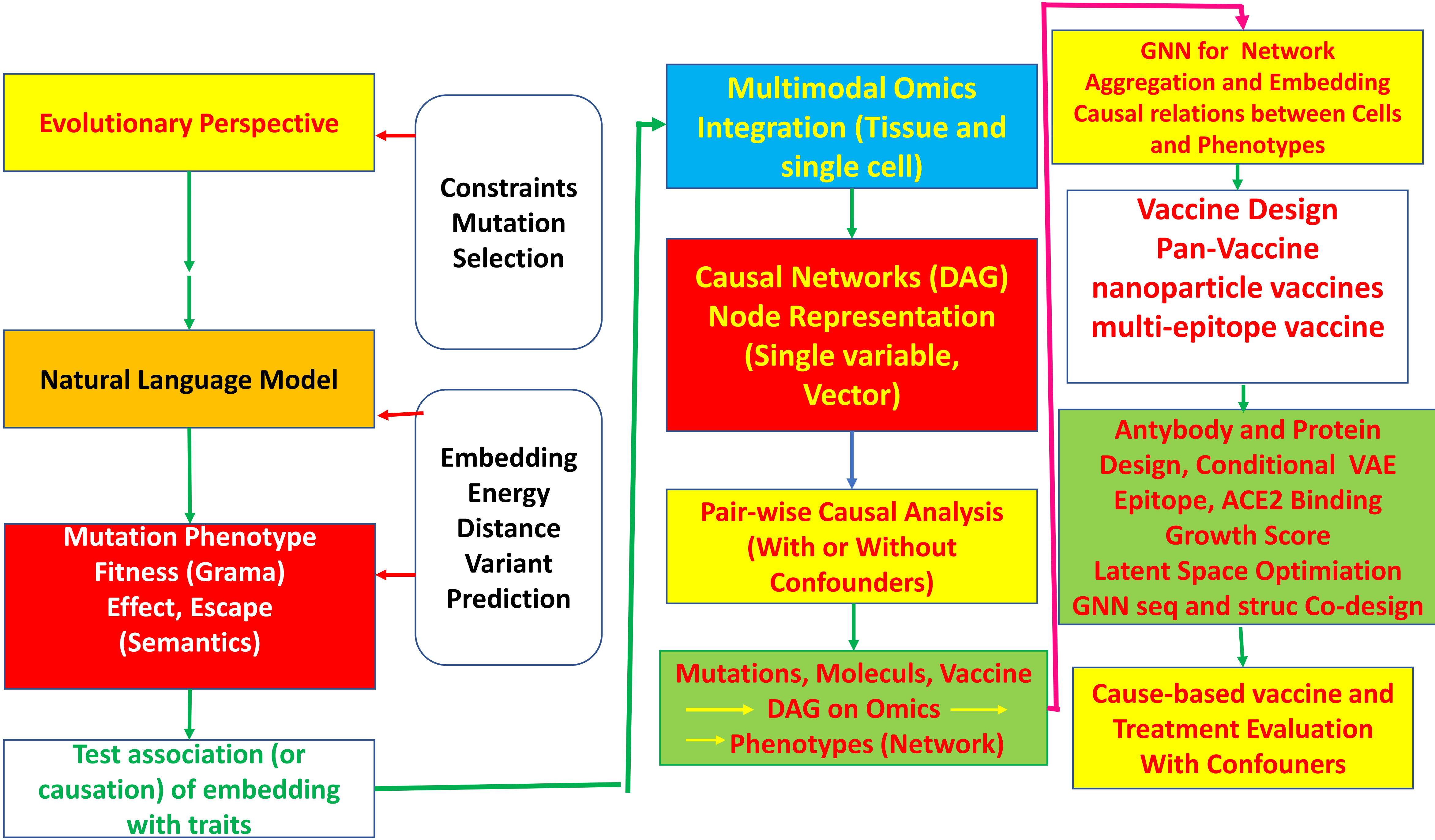
A Path to Uncovering Mechanism of Complex Trait


DNA
Gene Expression Causal Networks
Methylation Network
Protein Expression Causal Network
Metabolite Causal Network
Trait or Disease (Network)



Manifold alignment

Accurate inference of genome-wide spatial expression with iSpatial



Evolutionary Perspective

Natural Language Model

Mutation Phenotype
Fitness (Grama)
Effect, Escape
(Semantics)

Test association (or
causation) of embedding
with traits

Constraints
Mutation
Selection

Embedding
Energy
Distance
Variant
Prediction

Multimodal Omics
Integration (Tissue and
single cell)

Causal Networks (DAG)
Node Representation
(Single variable,
Vector)

Pair-wise Causal Analysis
(With or Without
Confounders)

Mutations, Moleculs, Vaccine
→ DAG on Omics →
→ Phenotypes (Network)

GNN for Network
Aggregation and Embedding
Causal relations between Cells
and Phenotypes

Vaccine Design
Pan-Vaccine
nanoparticle vaccines
multi-epitope vaccine

Antybody and Protein
Design, Conditional VAE
Epitope, ACE2 Binding
Growth Score
Latent Space Optimiation
GNN seq and struc Co-design

Cause-based vaccine and
Treatment Evaluation
With Confouners

18. Protein Language Model

ESM2:Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences

- **the UniParc database**
- **Number of amino acid sequences**: 250 millions

ProGen: Language Modeling for Protein Generation

Ali Madani et al. 2020

- **1.2 billion parameter conditional language**
- **Number of amino acid sequence:** 280 millions

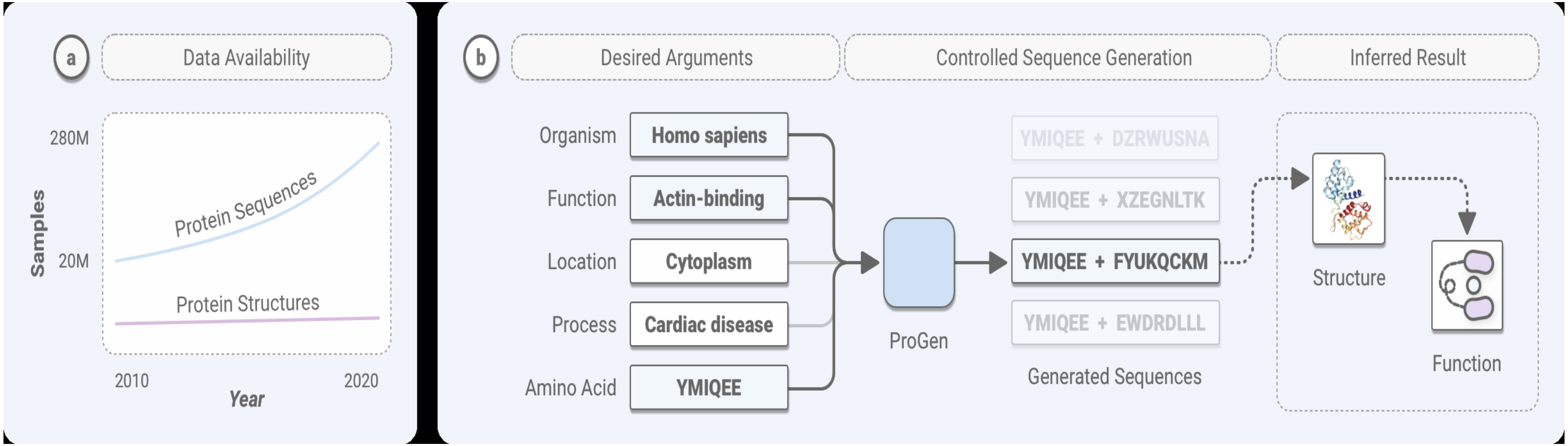
Generative modeling for protein engineering is key to solving fundamental problems in synthetic biology, medicine, and material science

18.1. ProGene

18.1.1. Generating proteins with desired properties

- the development of new enzymes, antibody, therapies, and sensors
- However, leading experimental techniques for protein engineering such as directed evolution (Arnold, 1998) still rely on heuristics and random mutations to select initial sequences for rounds of evolution.
- The raw amino acid sequence encodes a protein. This chain of amino acids folds in ways that exhibit local (secondary) and global (tertiary) structure, which in turn determines unique functions.
- Unfortunately, obtaining three-dimensional structural information for proteins is expensive and time consuming. Consequently, there are three orders of magnitude more raw sequences than there are sequences with structural annotations, and protein sequence data grow exponentially.

- **By conditioning on these tags, ProGen provides a new method for protein generation that can be tailored for desired properties**



a) Protein sequence data is growing exponentially as compared to structural data. b) We utilize protein sequence data along with and keyword tags to develop a conditional language model: ProGen.

Ali Madani et al. 2020 ProGen: Language Modeling for Protein Generation

18.1.2. Methods

- **Notations**

Amino Acid Sequence: $\boldsymbol{a} = \left(\boldsymbol{a}_1, \boldsymbol{a}_2, \quad , \boldsymbol{a}_{n_a}\right)$

Conditional Tag: $\boldsymbol{c} = \left(\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_{n_c}\right),\ \boldsymbol{n} = \boldsymbol{n}_a + \boldsymbol{n}_c$

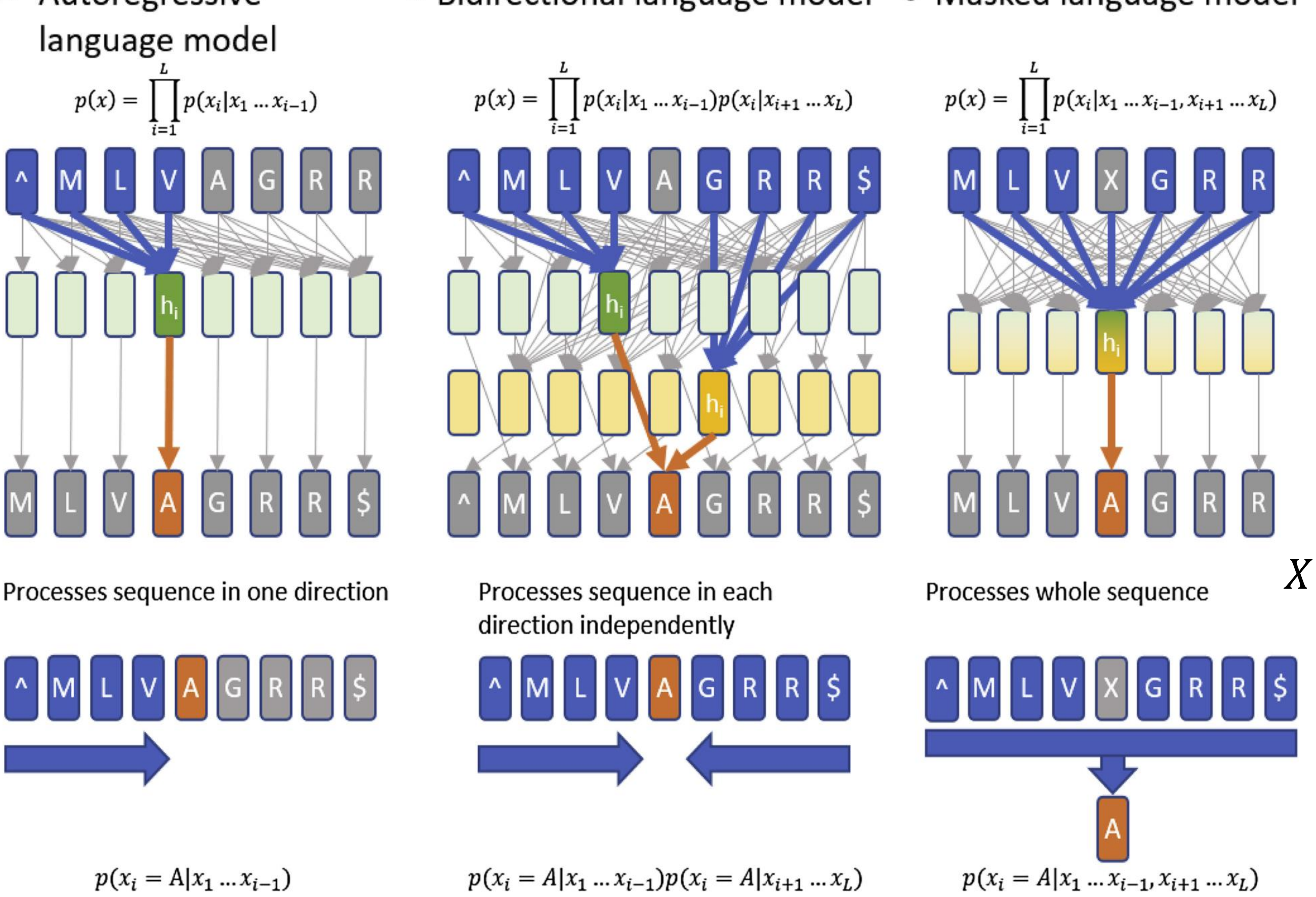
Sequence: $\boldsymbol{x} = (\boldsymbol{c}; \boldsymbol{a})$

Distribution: $\boldsymbol{P}(\boldsymbol{x}) = \prod_{i=1}^{n} \boldsymbol{P}(\boldsymbol{x}_i | \boldsymbol{x}_{<i})$

Dataset: $\boldsymbol{D} = \left[\boldsymbol{x}^1, \dots, \boldsymbol{x}^{|D|}\right]$

- **Loss Function**

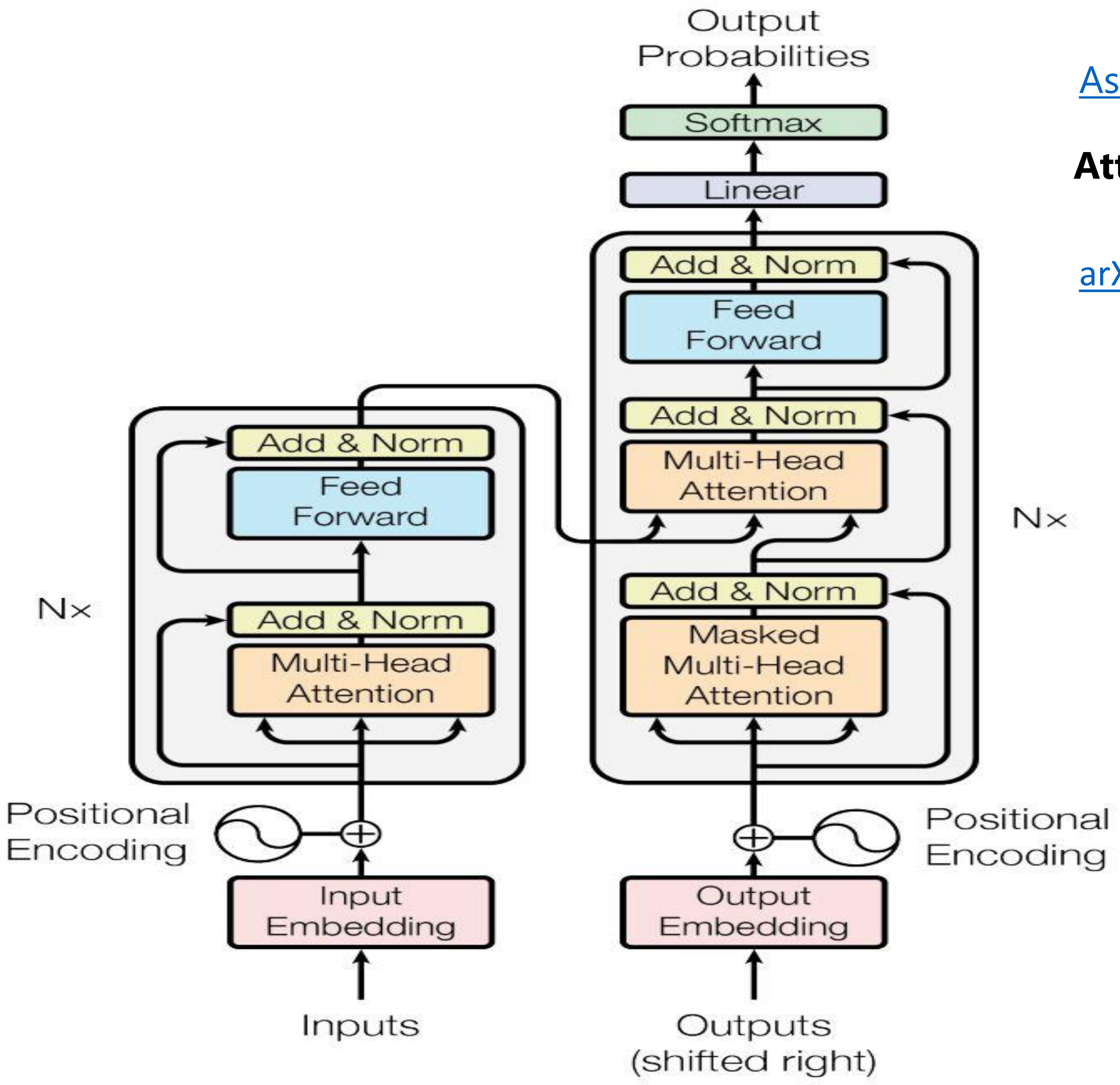
$$L(D) = -\sum_{k=1}^{|D|} \frac{1}{n_i} \sum_{i=1}^{n_i} \log P_\theta(x_i^k \,| x_{<i}^k)$$


Autoregressive language model
Bidirectional language model
Masked language model
$p(x) = \prod_{i=1}^{L} p(x_i|x_1 \dots x_{i-1})$
$p(x) = \prod_{i=1}^{L} p(x_i|x_1 \dots x_{i-1})p(x_i|x_{i+1} \dots x_L)$
$p(x) = \prod_{i=1}^{L} p(x_i|x_1 \dots x_{i-1}, x_{i+1} \dots x_L)$
^ M L V A G R R
h_i
M L V A G R R $
^ M L V A G R R $
h_i
h_i
^ M L V A G R R $
M L V X G R R
h_i
M L V A G R R
Processes sequence in one direction
Processes sequence in each direction independently
Processes whole sequence
^ M L V A G R R $
^ M L V A G R R $
^ M L V X G R R $
A
$p(x_i = \text{A}|x_1 \dots x_{i-1})$
$p(x_i = A|x_1 \dots x_{i-1})p(x_i = A|x_{i+1} \dots x_L)$
$p(x_i = A|x_1 \dots x_{i-1}, x_{i+1} \dots x_L)$

$$x_i = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}$$

$$X = \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & \cdots & 1 \end{bmatrix}$$

$$= (.)_{n \times d}$$



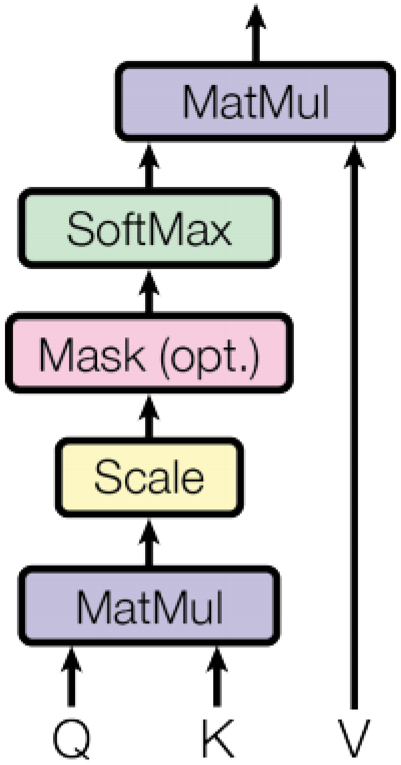
Ashish Vaswani et al. 2017

Attention Is All You Need

arXiv:1706.03762

Attention Mechanism

Scaled Dot-Product Attention



Matrices:

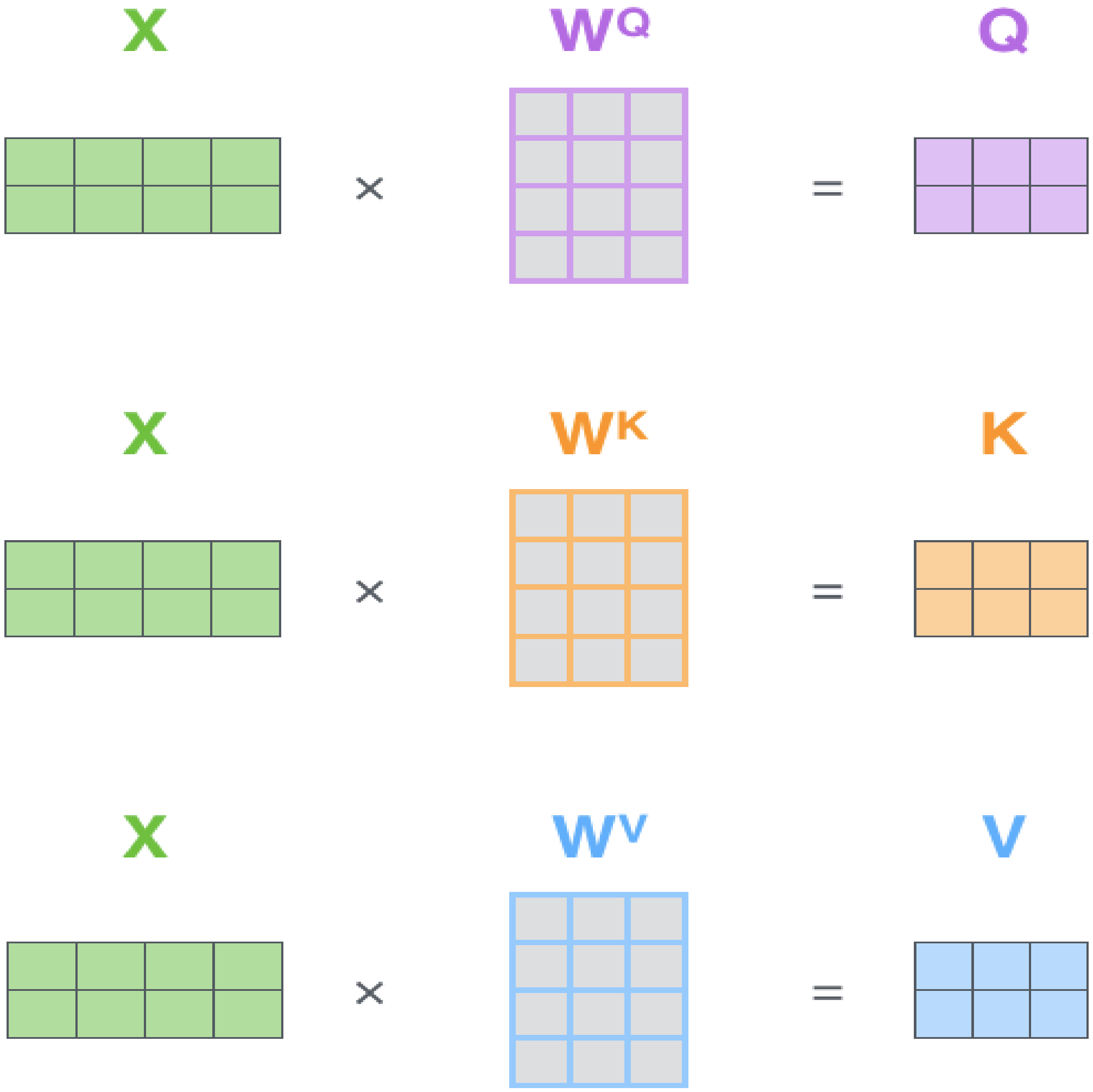
queries $Q \in R^{N\times D_k}, K \in R^{M\times D_k}, V \in R^{M\times D_v}$

$$\text{Attention } (Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{D_k}}\right)\mathrm{V} = \mathrm{AV}$$

$$A = \left(\alpha_{ij}\right)_{N\times M}$$

$$h_i = \sum\nolimits_j \alpha_{ij}V_j$$

$$\alpha_{ij} = \frac{\exp\{\sum_{l=1}^{D_k} q_{il}k_{jl}\}}{\sum_{j'=1}^{M}\exp\{\sum_{l=1}^{D_k} q_{il}k_{j'l}\}}$$



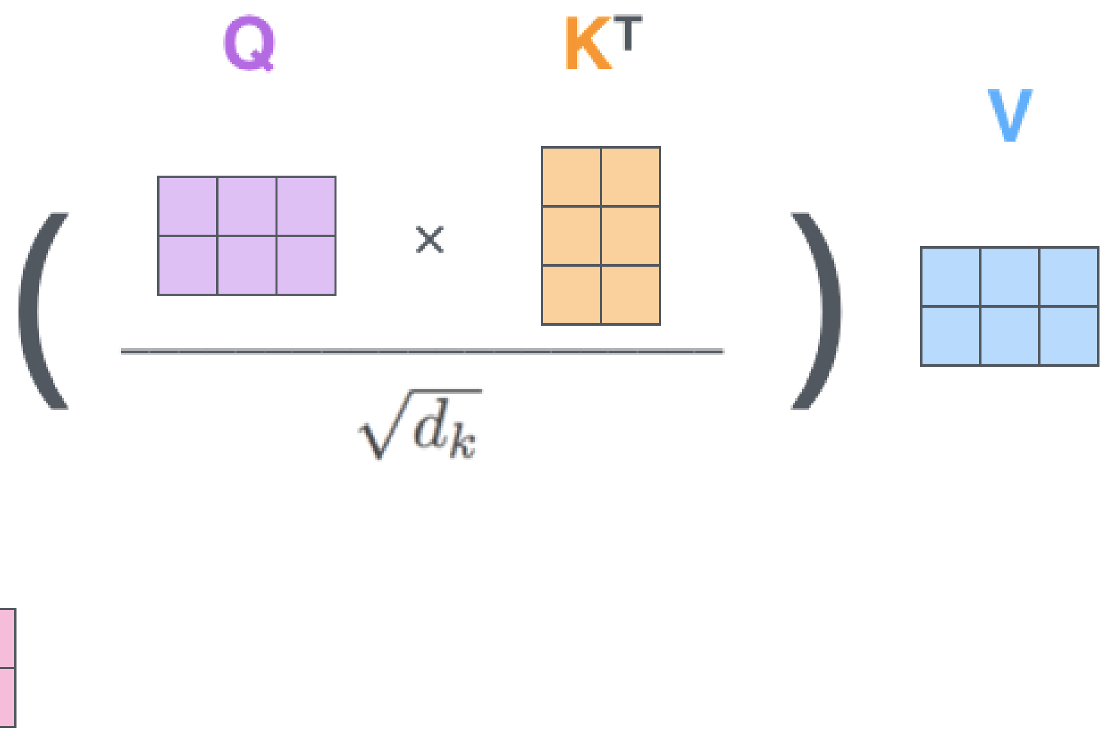
$$W^Q = W_j^1, Q = XW_j^1$$

$$W^k = W_j^2, K = XW_j^2$$

$$W^V = W_j^3, V = XW_j^3$$

https://zhuanlan.zhihu.com/p/47282410

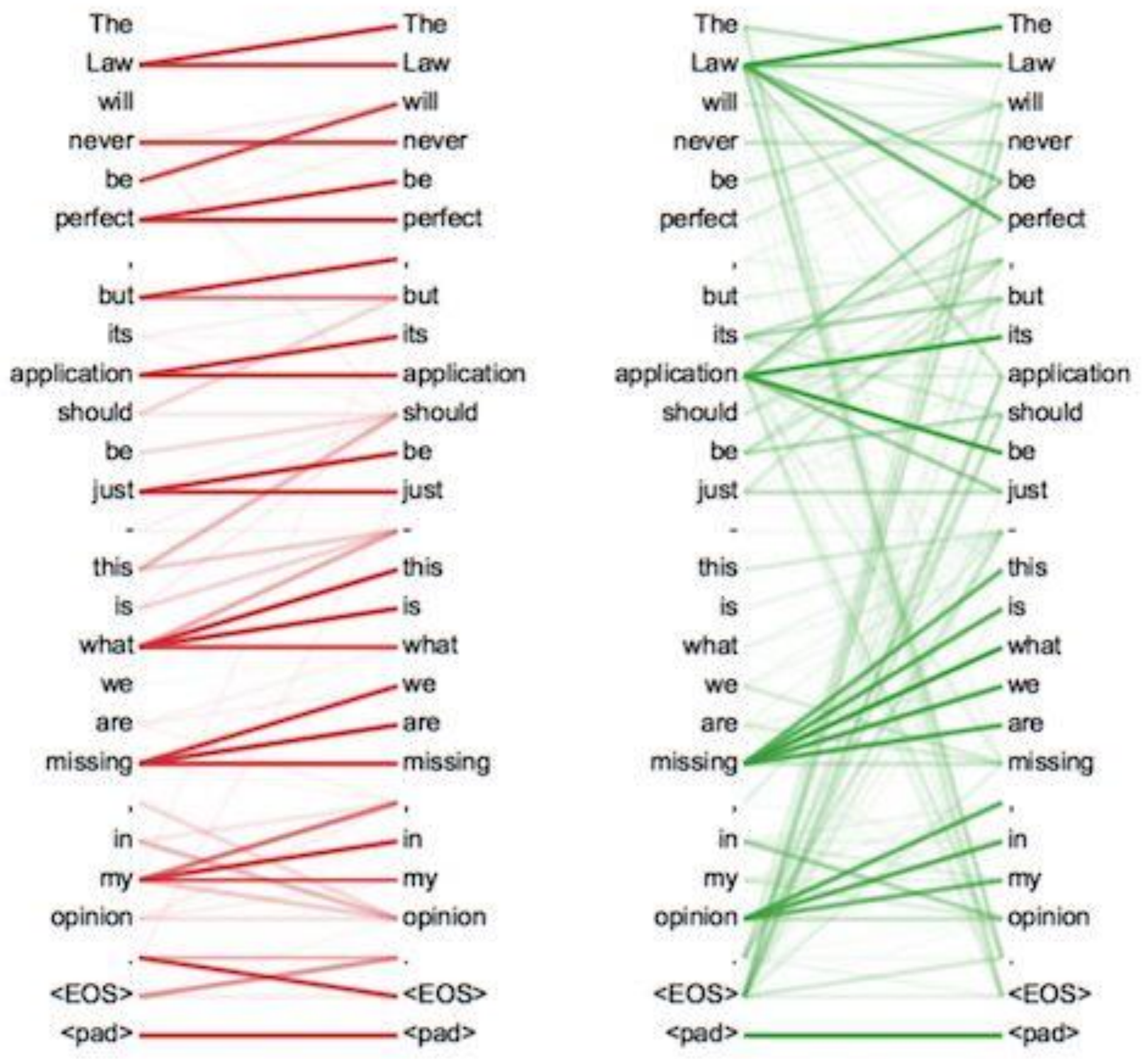
$$\text{Attention}\,(Q,K,V) = \texttt{softmax}\left(\frac{Q \times K^{T}}{\sqrt{d_k}}\right) V = Z$$

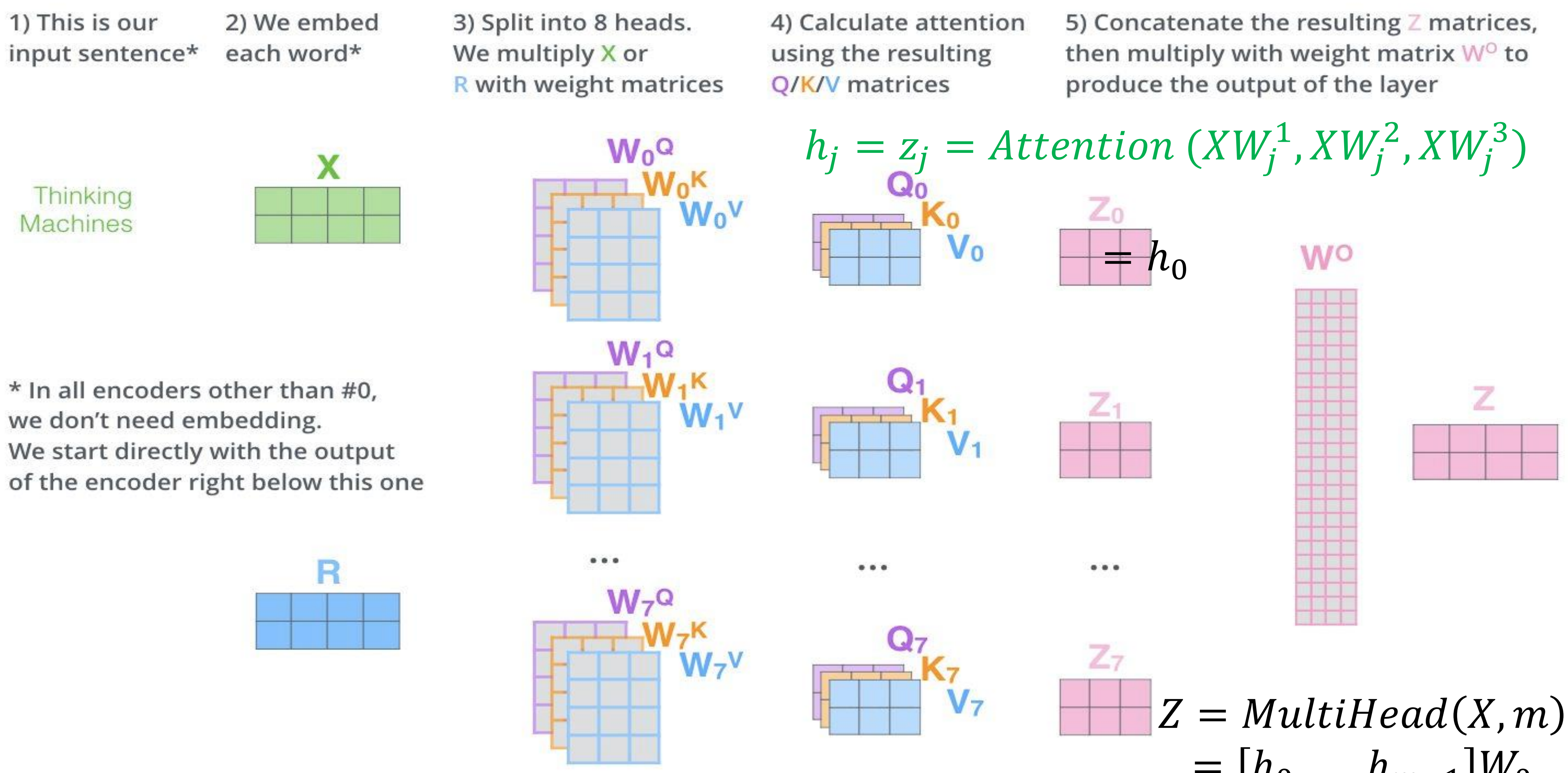


$$h_j = Z = Attention\,(XW_j^1, XW_j^2, XW_j^3)$$

https://zhuanlan.zhihu.com/p/47282410

Multihead


The
Law
will
never
be
perfect
,
but
its
application
should
be
just
-
this
is
what
we
are
missing
,
in
my
opinion
.
<EOS>
<pad>



1) This is our input sentence*

2) We embed each word*

3) Split into 8 heads. We multiply X or R with weight matrices

4) Calculate attention using the resulting Q/K/V matrices

5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer

* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one

$$h_j = z_j = Attention\,(XW_j^1, XW_j^2, XW_j^3)$$

$$= h_0$$

$$Z = MultiHead(X, m)$$
$$= [h_0, \dots, h_{m-1}]W_0$$

Residual Connection and Layer Normalization

$$H = Multihead\,(X, m) + X$$

$$\overline{H} = LayerNorm\,(H)$$

Position-wise FFN.

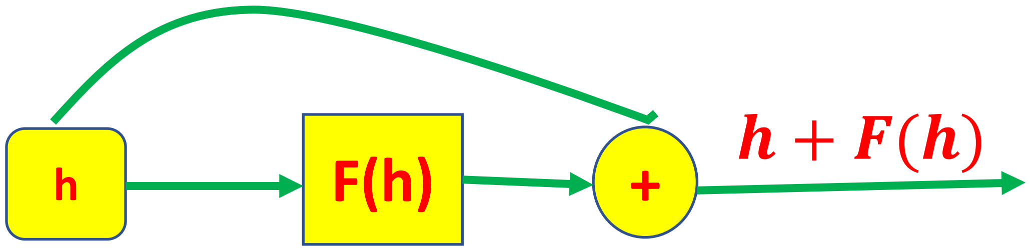
The position-wise FFN is a fully connected feed-forward module that operates separately and identically on each position

$$\boldsymbol{FFN(h) = ReLU\left(\overline{H}W^1 + b^1\right)W^2 + b^2}$$

the outputs of previous layer:

$$\boldsymbol{\overline{H}_i = \left[\overline{h}_1, \ldots, \overline{h}_m\right], W^1 \in R^{m \times D_f}, W^2 \in R^{D_f \times m}, b^1 \in R^{D_f}, b^2 \in R^m}$$

Residual Connection and Normalization

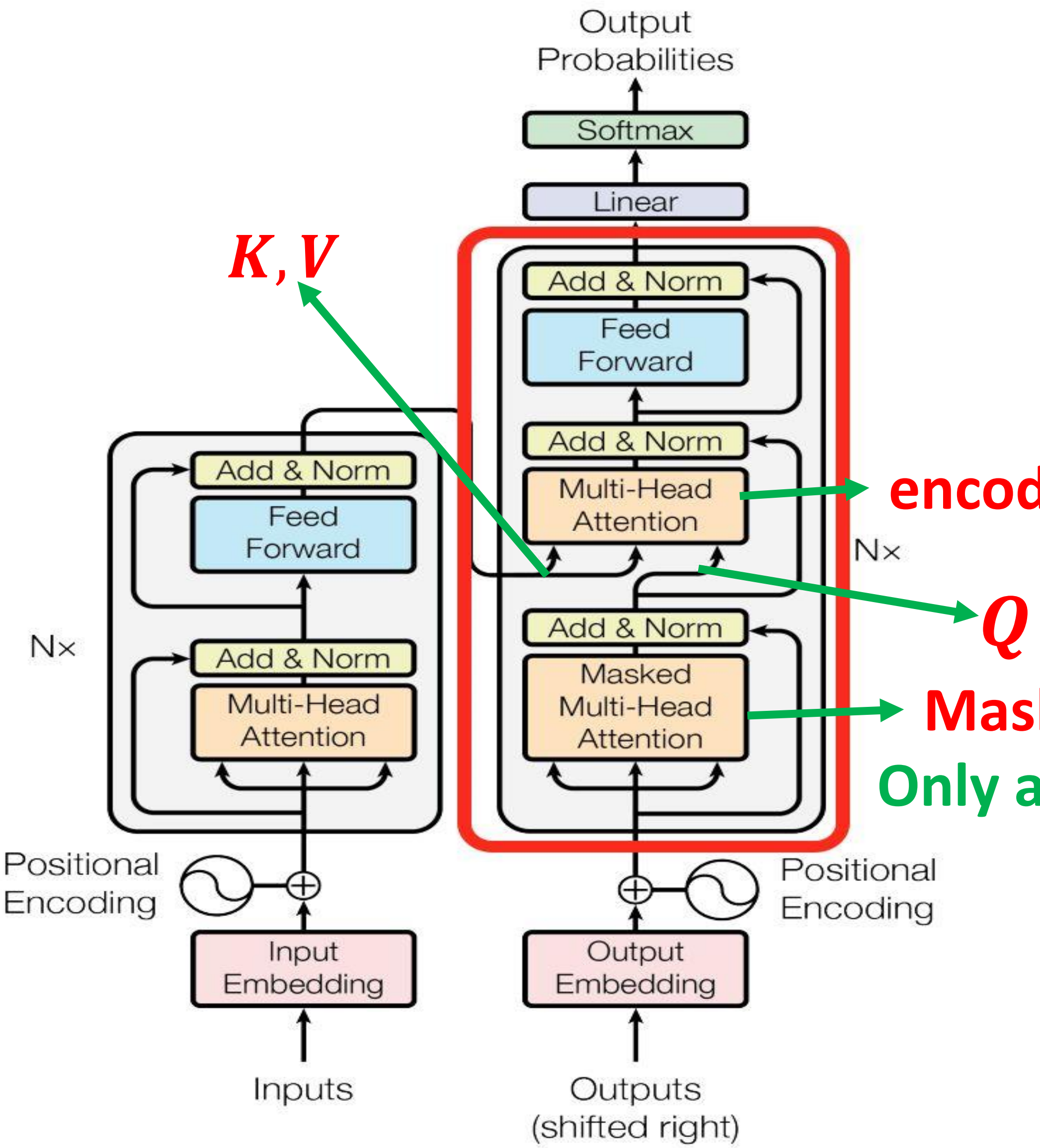


$$h' = LayerNorm(SelfAttention(h^{(l)}) + h^{(l)})$$

$$h^{(l+1)} = LayerNorm(FFN(h') + h')$$

$$X_{i+1} = LayerNorm(FFN\left(\bar{H}_i\right) + \bar{H}_i)$$

Decoder



encoder-decoder attention layer

Masked Multi-head

Only attend to the words that have been translated.

Scores

$$\boldsymbol{Scores\ (X_0) = LayerNorm(X_L)W_{Vocab}}$$

During training, these scores are the inputs of a cross entropy loss function. During generation, the scores corresponding to the final token are normalized with a softmax, yielding a distribution for sampling a new token.

TRANSFORMER MODELS: AN INTRODUCTION AND CATALOG

Xavier Amatriain
Los Gatos, CA 95032
xavier@amatriain.net

arXiv:2302.07730

Cover (One Topic at One Lecture)

- ProGen: Language Modeling for Protein Generation
- ProGen Generate functional protein sequences
- ESM2: Language Model generalize beyond natural proteins
- Language Models and Diffusion Process
- Language Model for Antibody Design
- MULTI-LEVEL PROTEIN STRUCTURE PRE-TRAINING WITH PROMPT LEARNING
- DNA Language Model
- Protein Language Models for Protein Docking
- Embeddings from language models predict conservation and variant effects
- Table Data, Language and Omics Data Embedding
- A new path to uncovering Disease Mechanism using language models, Causal Inference and Omics
- A New Paradigm for Drug Discovery and Drug Repurposing