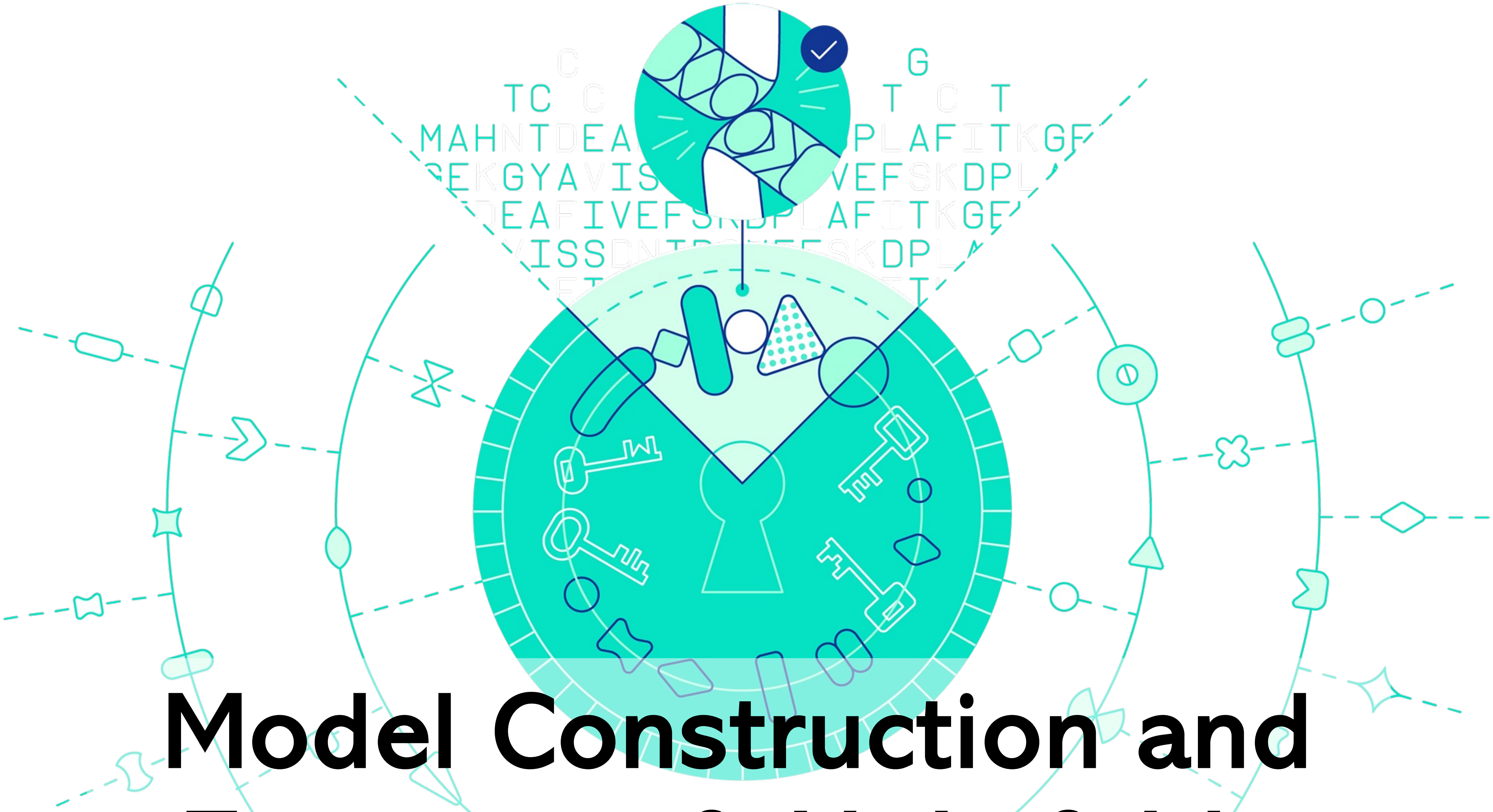
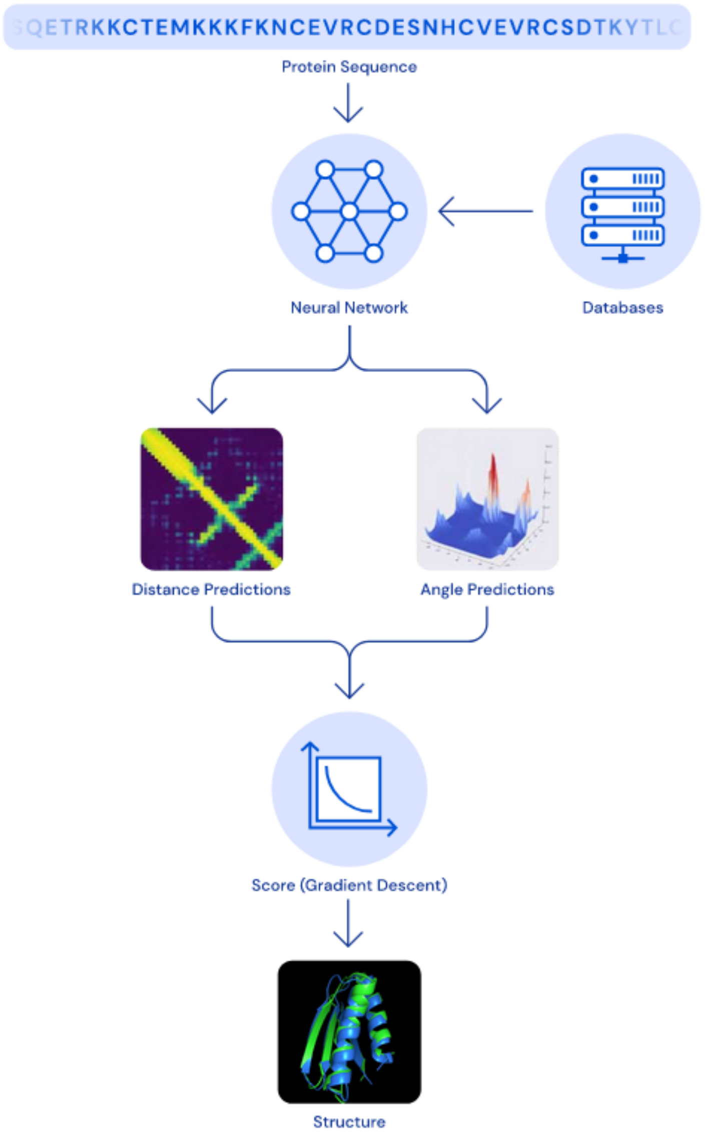


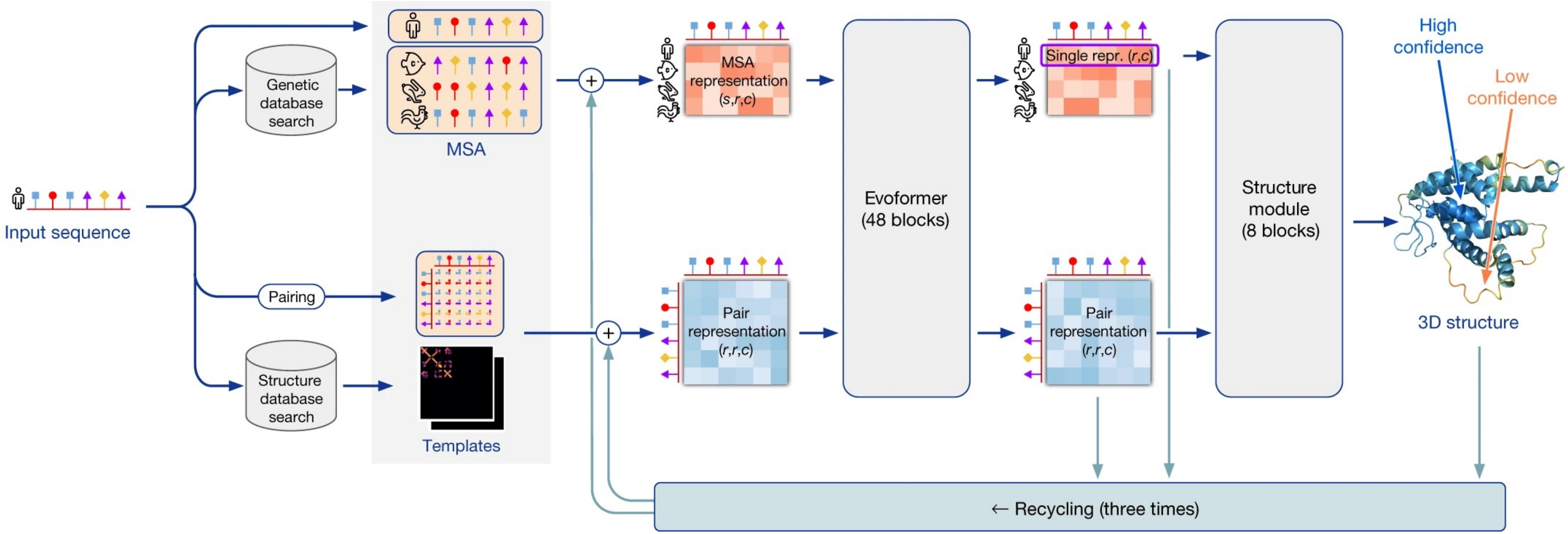
# Model Construction and Features of Alphafold2

# Lecture topics

- Input
- Evoformer module
- Structure module
- Output formatting and recycling
- Training regime and data
- Inference

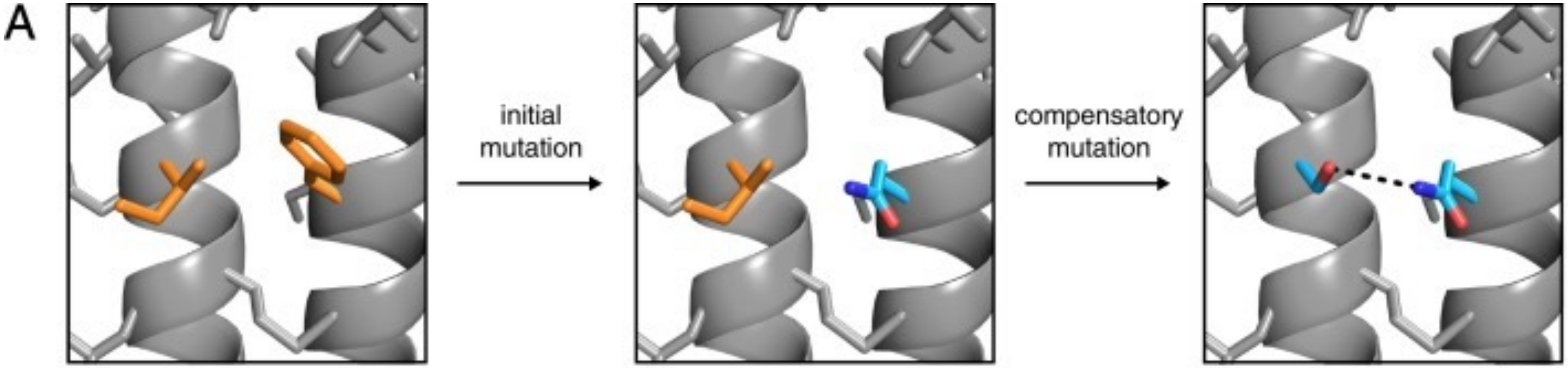


# Network overview


Input sequence
Genetic database search
Pairing
Structure database search
MSA
Templates
+
+
MSA representation (s,r,c)
Pair representation (r,r,c)
Evoformer (48 blocks)
Single repr. (r,c)
Pair representation (r,r,c)
Structure module (8 blocks)
High confidence
Low confidence
3D structure
← Recycling (three times)

# Input

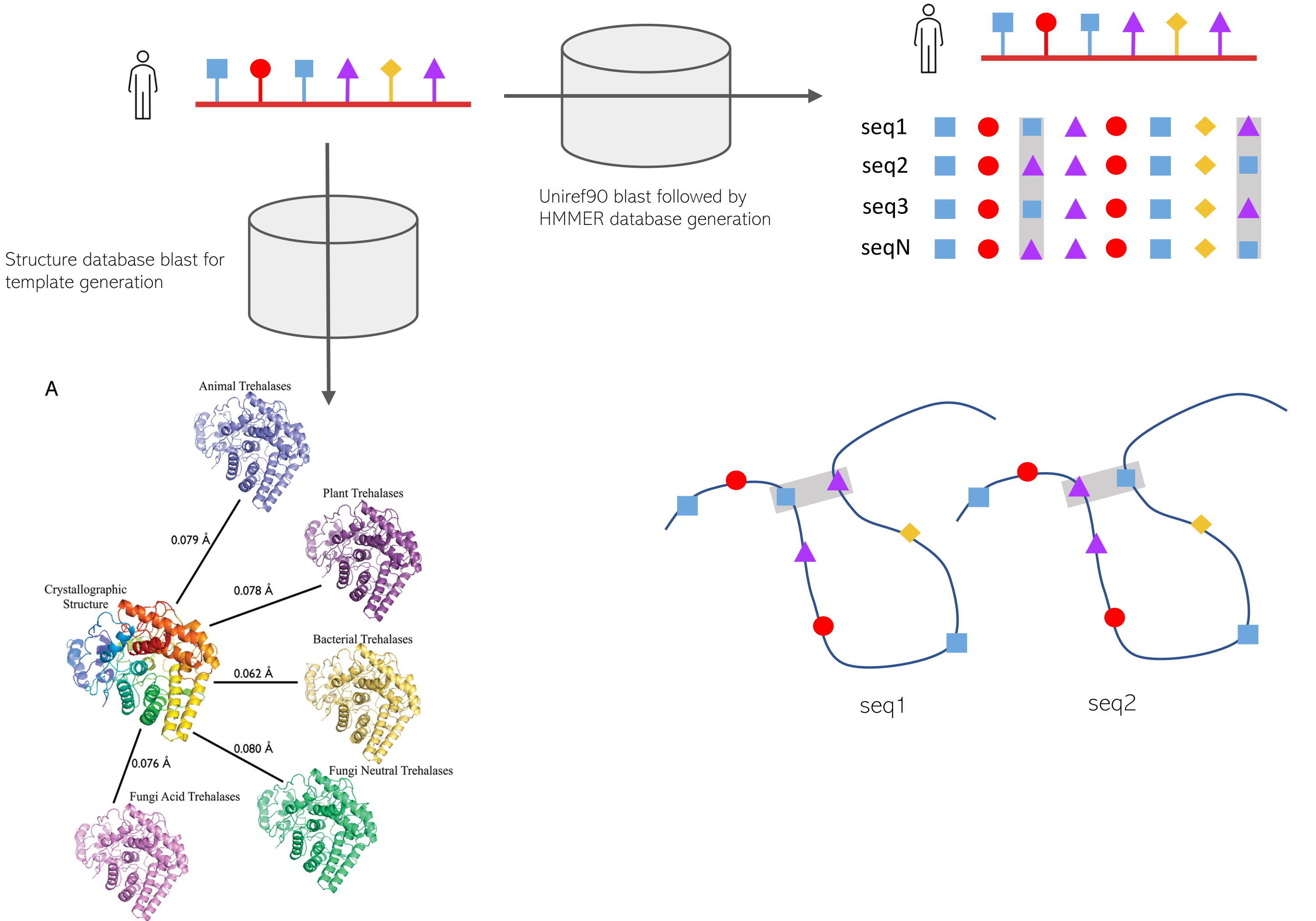
- How is 3D structure encoded within the sequence?



Nicoludis, J. M., & Gaudet, R. (2018). Applications of sequence coevolution in membrane protein biochemistry. Biochimica et Biophysica Acta (BBA) - Biomembranes, 1860(4), 895–908. doi:10.1016/j.bbamem.2017.10.004

# Input

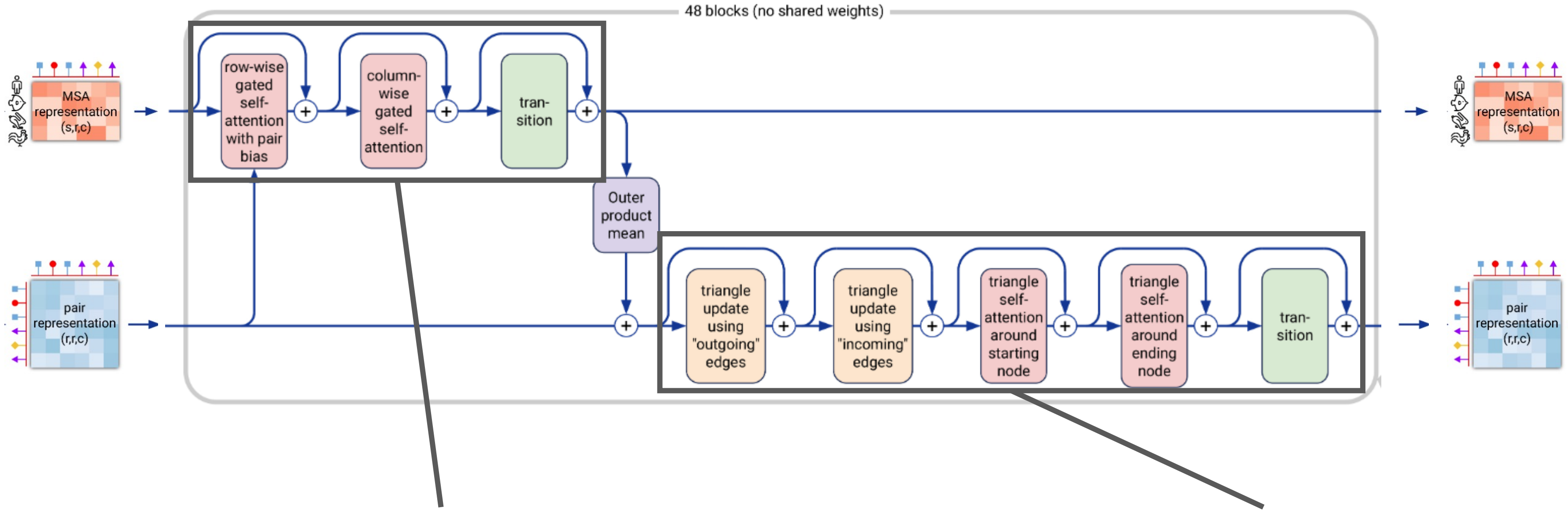
- How is 3D structure encoded within the sequence?



Structure database blast for template generation

Uniref90 blast followed by HMMER database generation

# Evoformer module


48 blocks (no shared weights)
MSA representation (s,r,c)
pair representation (r,r,c)
row-wise gated self-attention with pair bias
column-wise gated self-attention
tran-sition
Outer product mean
triangle update using "outgoing" edges
triangle update using "incoming" edges
triangle self-attention around starting node
triangle self-attention around ending node
tran-sition
MSA representation (s,r,c)
pair representation (r,r,c)

Conservation, trends among sequences

Self-attention gives context clues to importance across the sequence, as well as the relation between difference sequences.

Pairwise residue co-information

Pair representation to notice context-related clues, such as coevolution to gauge structural closeness

# Evoformer module

- Problem: How do you make a computer understand the contact network from a sequence?

# Evoformer module

- Problem: How do you make a computer understand the contact network from a sequence?

- Problem: How do you understand a meaning from a sequence of letters?

# Evoformer module

- Problem: How do you make a computer understand the contact network from a sequence?

- STR**E**VKLR
- LLV**E**ILVAAG

- Problem: How do you understand a meaning from a sequence of letters?

- Walk by the river bank vs.
- Get cash from the bank

# Evoformer module

- Problem: How do you make a computer understand the contact network from a sequence?

- STR**E**VKLR
- LLV**E**ILVAAG
  - Language models and suitability
  - Self-attention networks

- Problem: How do you understand a meaning from a sequence of letters?

- Walk by the river bank vs.
- Get cash from the bank

# Evoformer module

- Language models and suitability
- Self-attention networks
- Embedding space

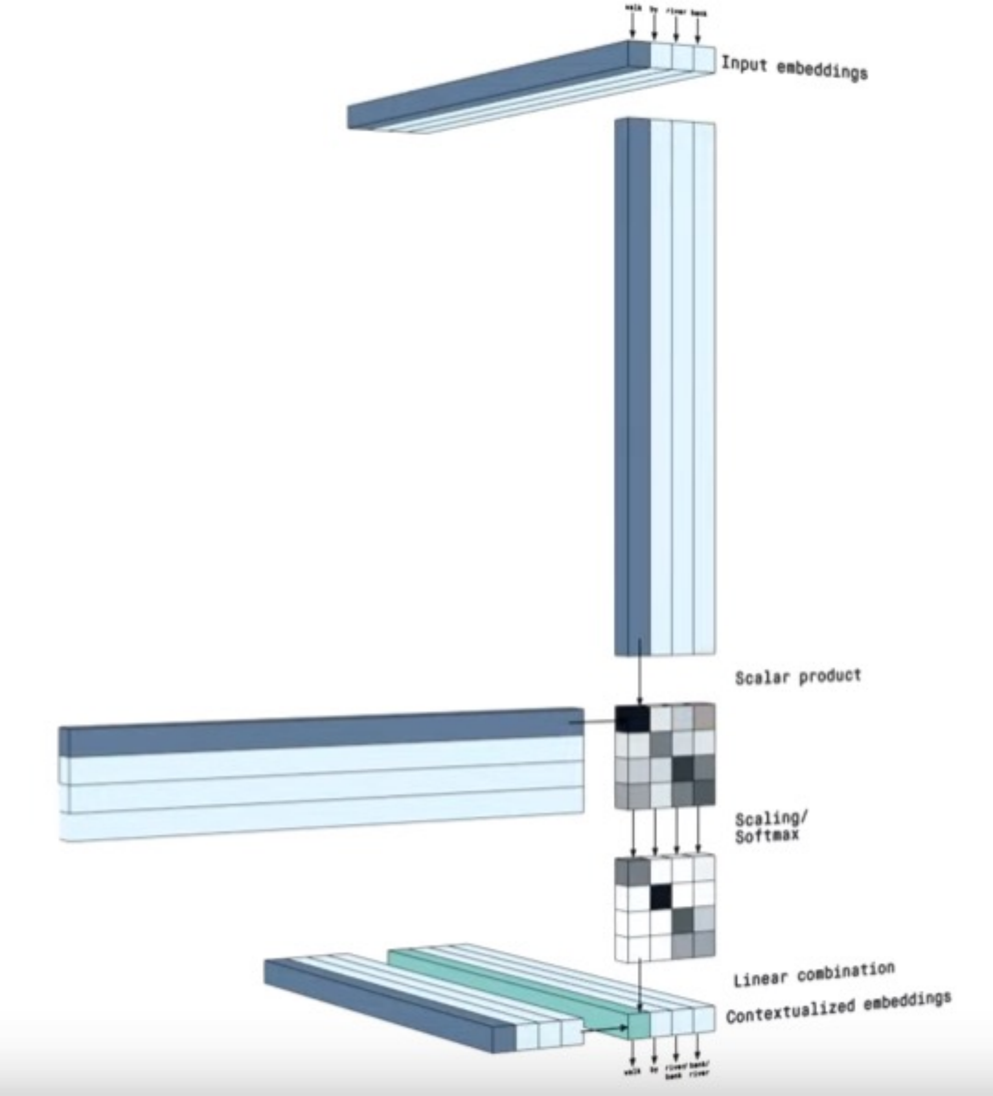
**Walk** by the river bank

[ 0 0 0 0 1 0 0 0 0 ] **One-hot encoding**

$\boldsymbol{W_V * o.h.e}$

[ 0.2 0.3 0.4 0.1 0.8 0.7 ] $\boldsymbol{V_0}$**: First embedded word ("walk")**

**Attention(V) =** $\boldsymbol{\frac{1}{\sqrt{\dim(V)}} softmax\left(V * V^T\right) * V}$



# Evoformer module

- Language models and suitability
- Self-attention networks
- Query, Key and Value (variable embedding)

**Walk** by the river bank

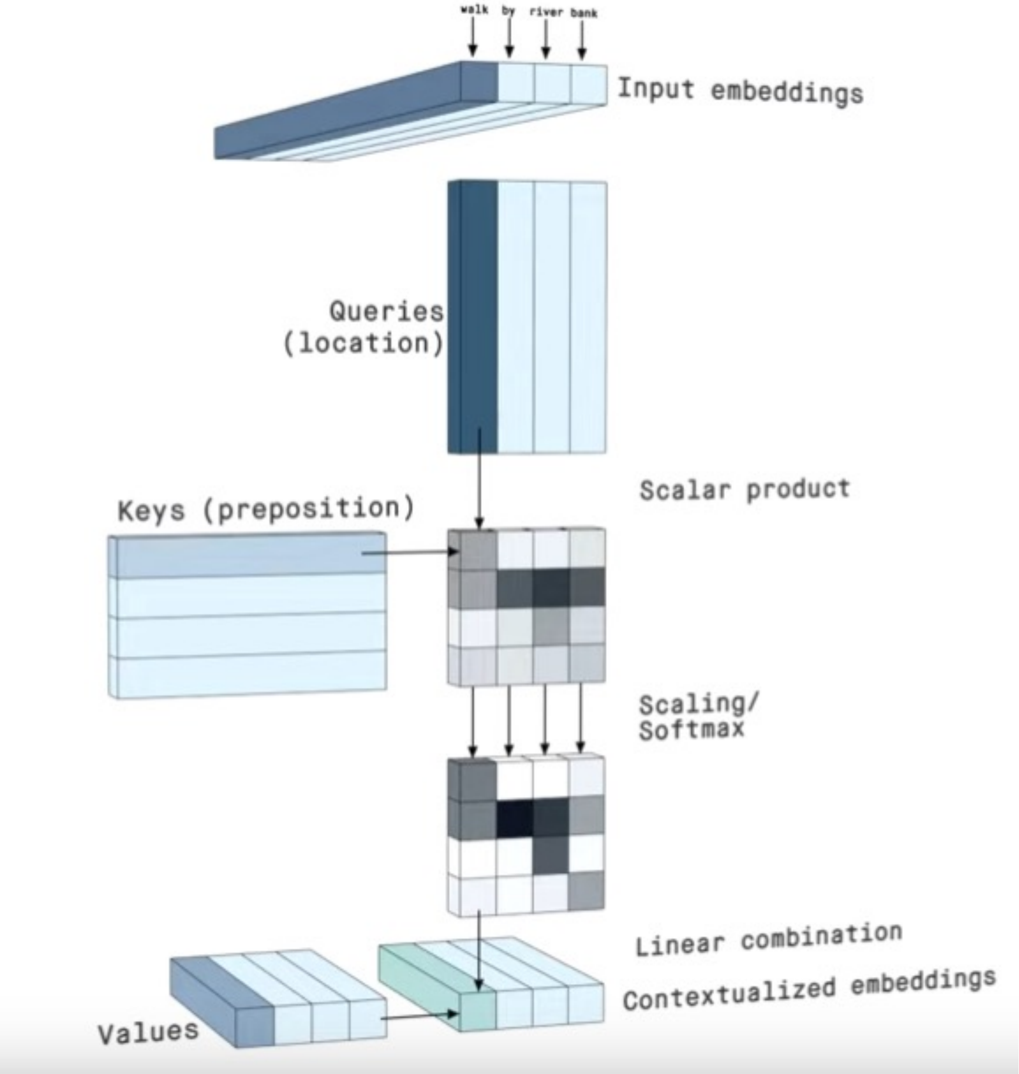
**One-hot encoding** $\begin{pmatrix}1 & \cdots & 0\\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1\end{pmatrix}$

$$\boldsymbol{W_q * o.h.e = Q}$$

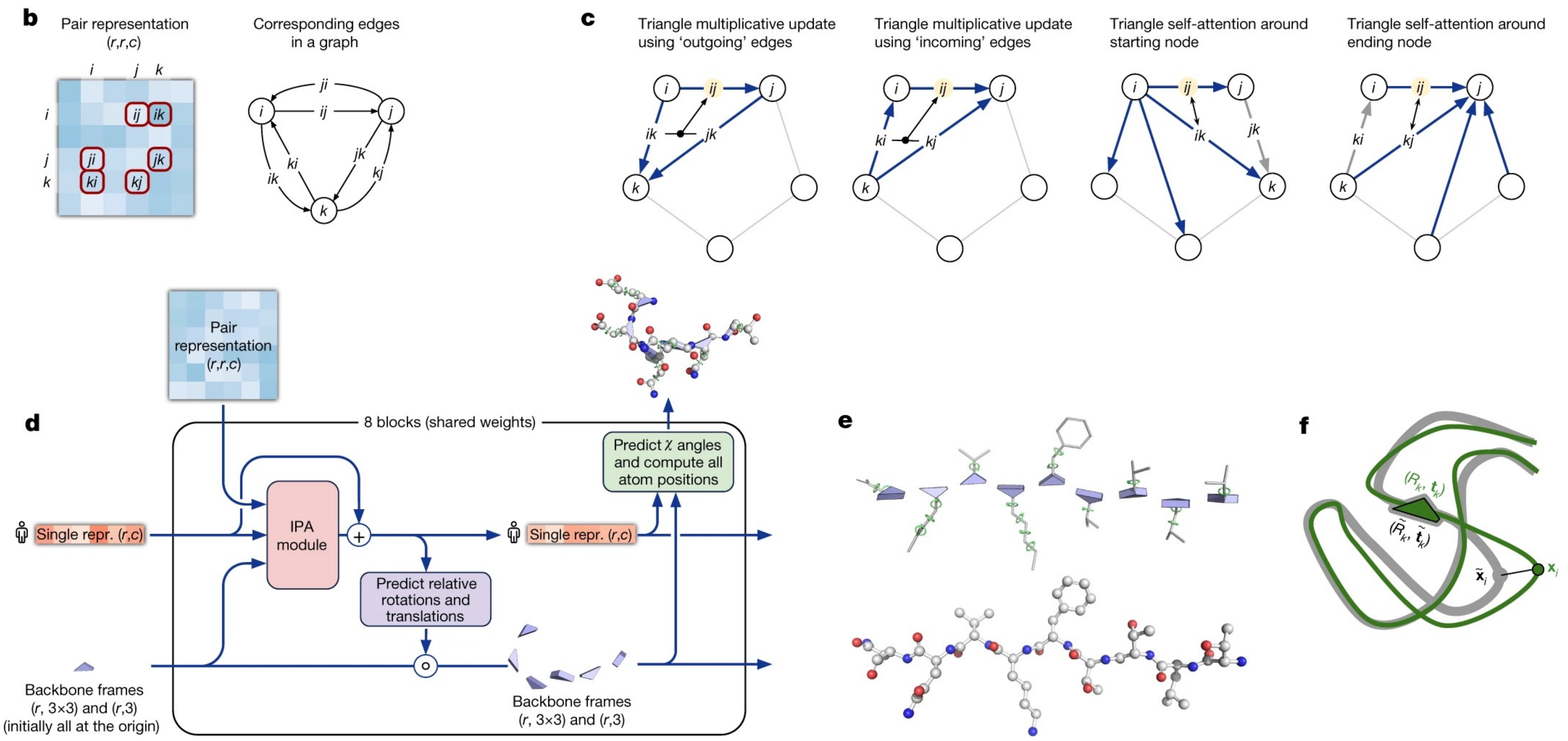
$$\boldsymbol{W_K * o.h.e = K}$$

$$\boldsymbol{W_V * o.h.e = V}$$

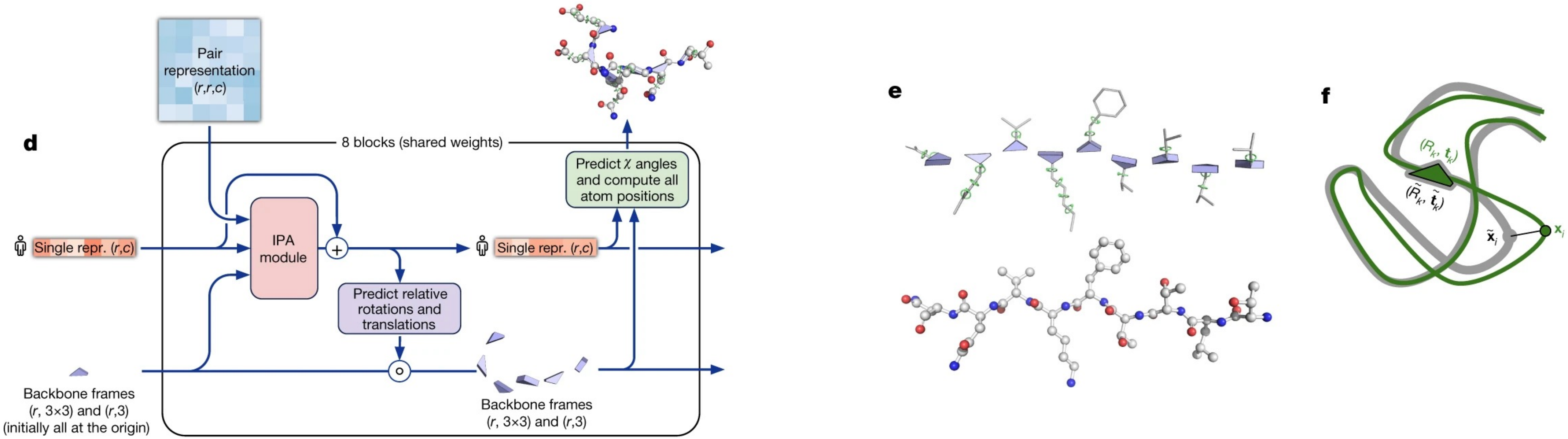
**Attention(Q,K,T) =** $\frac{1}{\sqrt{\dim(Q,K,T)}} softmax(Q * K^T) * V$



# Structure module



# Structure module


d
Pair representation (r,r,c)
8 blocks (shared weights)
Single repr. (r,c)
IPA module
+
Predict relative rotations and translations
Single repr. (r,c)
Predict χ angles and compute all atom positions
Backbone frames (r, 3×3) and (r,3) (initially all at the origin)
Backbone frames (r, 3×3) and (r,3)
e
f
(R_k, t_k)
(R̃_k, t̃_k)
x̃_i
x_i

- 3D representations iteratively built from the evoformer pair representations with the single representation as input.

- “Rapidly develop[s] and refine[s] a highly accurate protein structure with precise atomic details. “

- Breaking the chain structure (forbidden in previous methods) and putting substantial weights on pairs from evoformer (requires evoformer to provide all information)

- Iterative refinement using recycling

# Structure module

- Invariant Point Attention (IPA)
- The invariance comes from that the global transformation cancels out in the affinity computation, since L2-norm of a vector is invariant under rigid transformations.

**One-hot encoding** $\begin{pmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{pmatrix}$

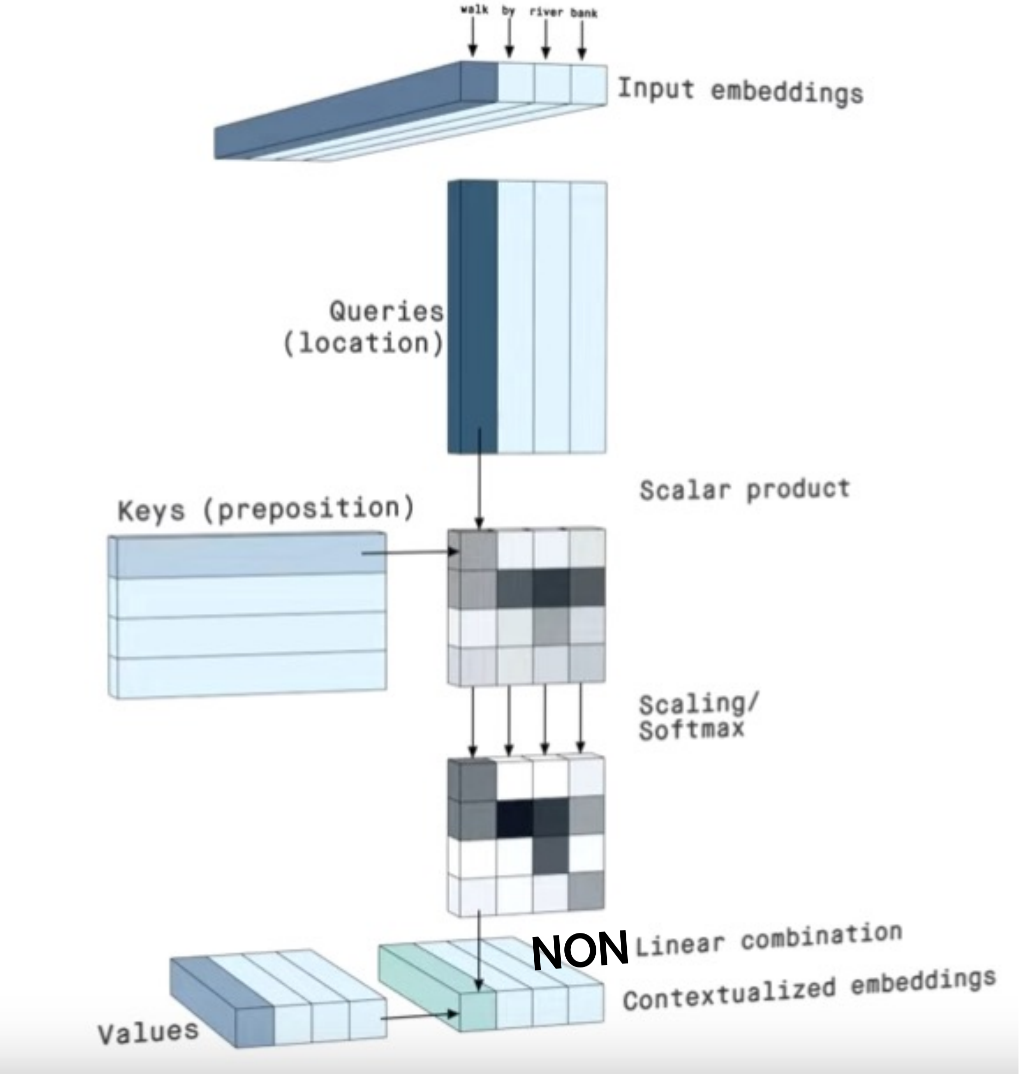
$$\boldsymbol{W_q * o.h.e = Q}$$

$$\boldsymbol{W_K * o.h.e = K}$$

$$\boldsymbol{W_V * o.h.e = V}$$

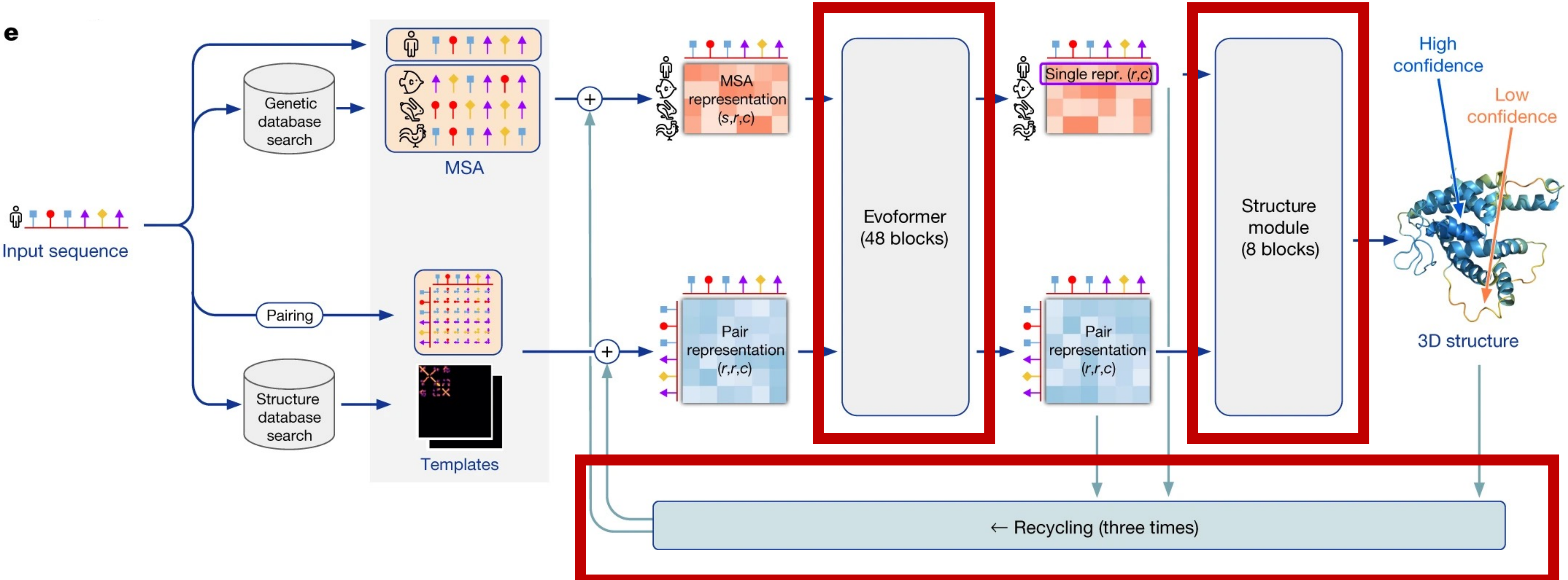
**Attention(Q,K,T) =** $\frac{1}{\sqrt{\dim(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{T})}} \boldsymbol{softmax}\left(\boldsymbol{Q * K^T + b - \frac{gw_c}{2} *} \right.$

$\left. \boldsymbol{Translation * (Q - K)}\right) \boldsymbol{* V}$



# Recycling

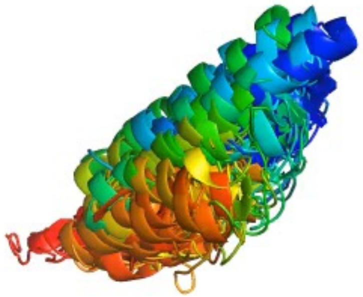
- Efficiency
- Convergence properties


e
Input sequence
Genetic database search
Pairing
Structure database search
MSA
Templates
MSA representation (s,r,c)
Pair representation (r,r,c)
Evoformer (48 blocks)
Single repr. (r,c)
Pair representation (r,r,c)
Structure module (8 blocks)
High confidence
Low confidence
3D structure
← Recycling (three times)

# Training regime and data preprocessing

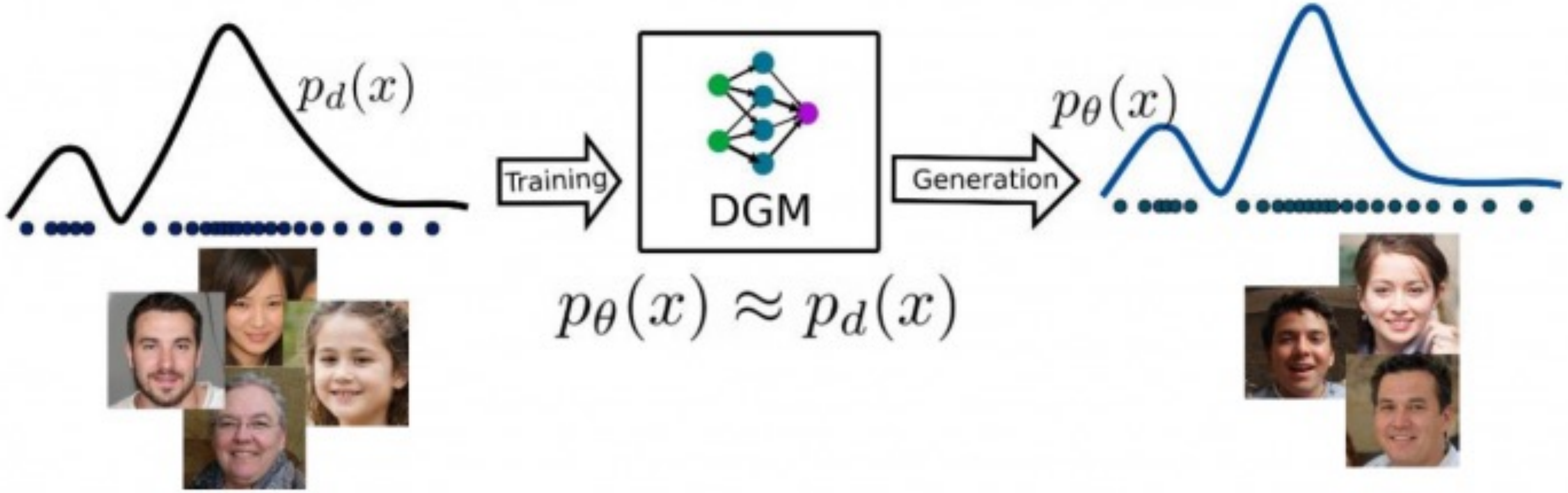
- For evaluation on recent PDB sequences (Figs. 2a–d, 4a, 5a), we used a copy of the PDB downloaded 15 February 2021. Structures were filtered to those with a release date after 30 April 2018 (the date limit for inclusion in the training set for AlphaFold). Chains were further filtered to remove sequences that consisted of a single amino acid as well as sequences with an ambiguous chemical component at any residue position. Exact duplicates were removed, with the chain with the most resolved Cα atoms used as the representative sequence. Subsequently, structures with less than 16 resolved residues, with unknown residues or solved by NMR methods were removed. As the PDB contains many near-duplicate sequences, the chain with the highest resolution was selected from each cluster in the PDB 40% sequence clustering of the data. Furthermore, we removed all sequences for which fewer than 80 amino acids had the alpha carbon resolved and removed chains with more than 1,400 residues. The final dataset contained 10,795 protein sequences.
- The procedure for filtering the recent PDB dataset based on prior template identity was as follows. Hmmsearch was run with default parameters against a copy of the PDB SEQRES fasta downloaded 15 February 2021. Template hits were accepted if the associated structure had a release date earlier than 30 April 2018. Each residue position in a query sequence was assigned the maximum identity of any template hit covering that position. Filtering then proceeded as described in the individual figure legends, based on a combination of maximum identity and sequence coverage.
- The MSA depth analysis was based on computing the normalized number of effective sequences ($N_{\text{eff}}$) for each position of a query sequence. Per-residue $N_{\text{eff}}$ values were obtained by counting the number of non-gap residues in the MSA for this position and weighting the sequences using the $N_{\text{eff}}$ scheme[76] with a threshold of 80% sequence identity measured on the region that is non-gap in either sequence.

# Inference reasoning



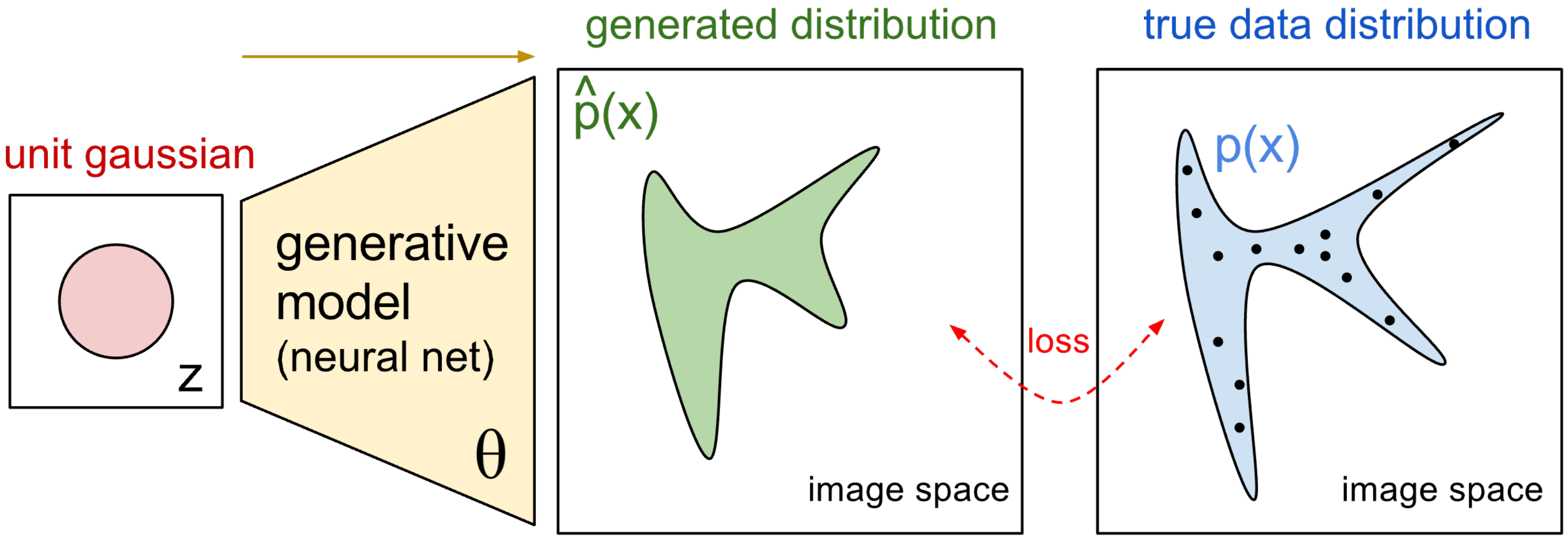
Recycling iteration 0, block 01
Secondary structure assigned from the final prediction

# Part 2: Bioinformatics and generative modelling



# General properties

- Black box appproach
- Probability and generative capabilities
- Feature importance

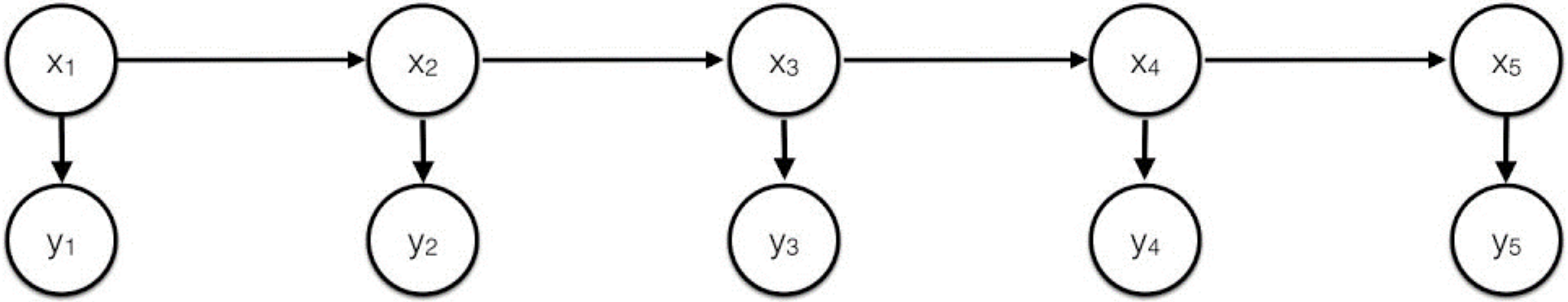


# HMMs

$p(y_t|x_t)$ observation probability

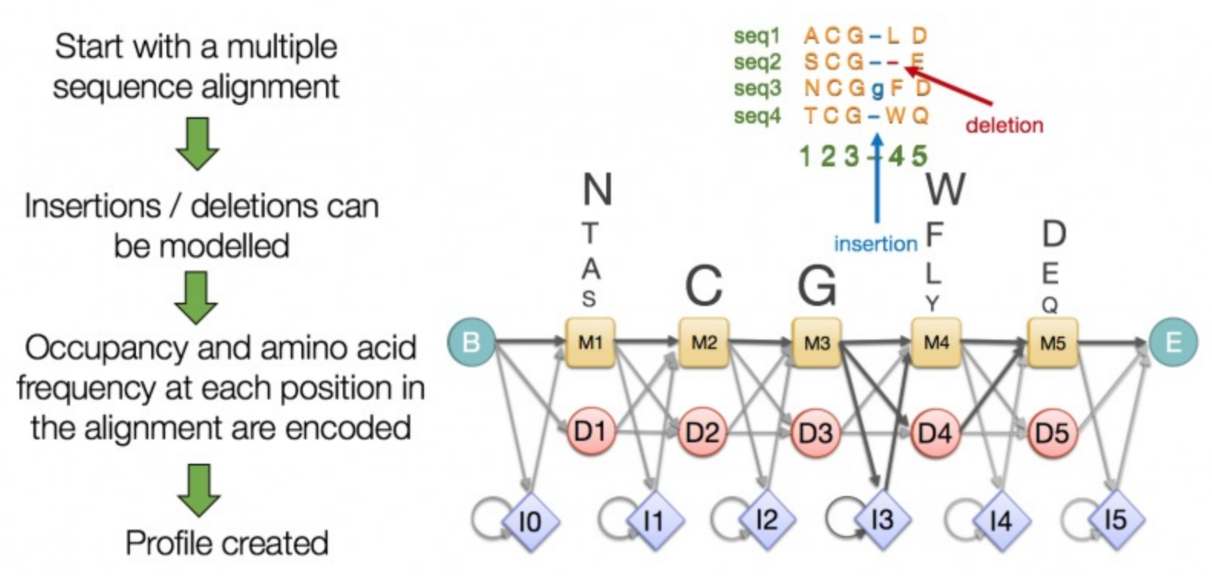
$p(x_t|x_{t-1})$ transition probability

$$p(X, Y) = p(x_1) \prod_{t=1}^{T-1} p(x_{t+1}|x_t) \prod_{t'=1}^{T} p(y_{t'}|x_{t'})$$



# HMMs - applications

- Sequence bioinformatics
  - MSA representation as a sequence of strings
  - MSA representation as a sequence of source outputs
  - Every sequence is a result of a random walk between sources



Start with a multiple sequence alignment

Insertions / deletions can be modelled

Occupancy and amino acid frequency at each position in the alignment are encoded

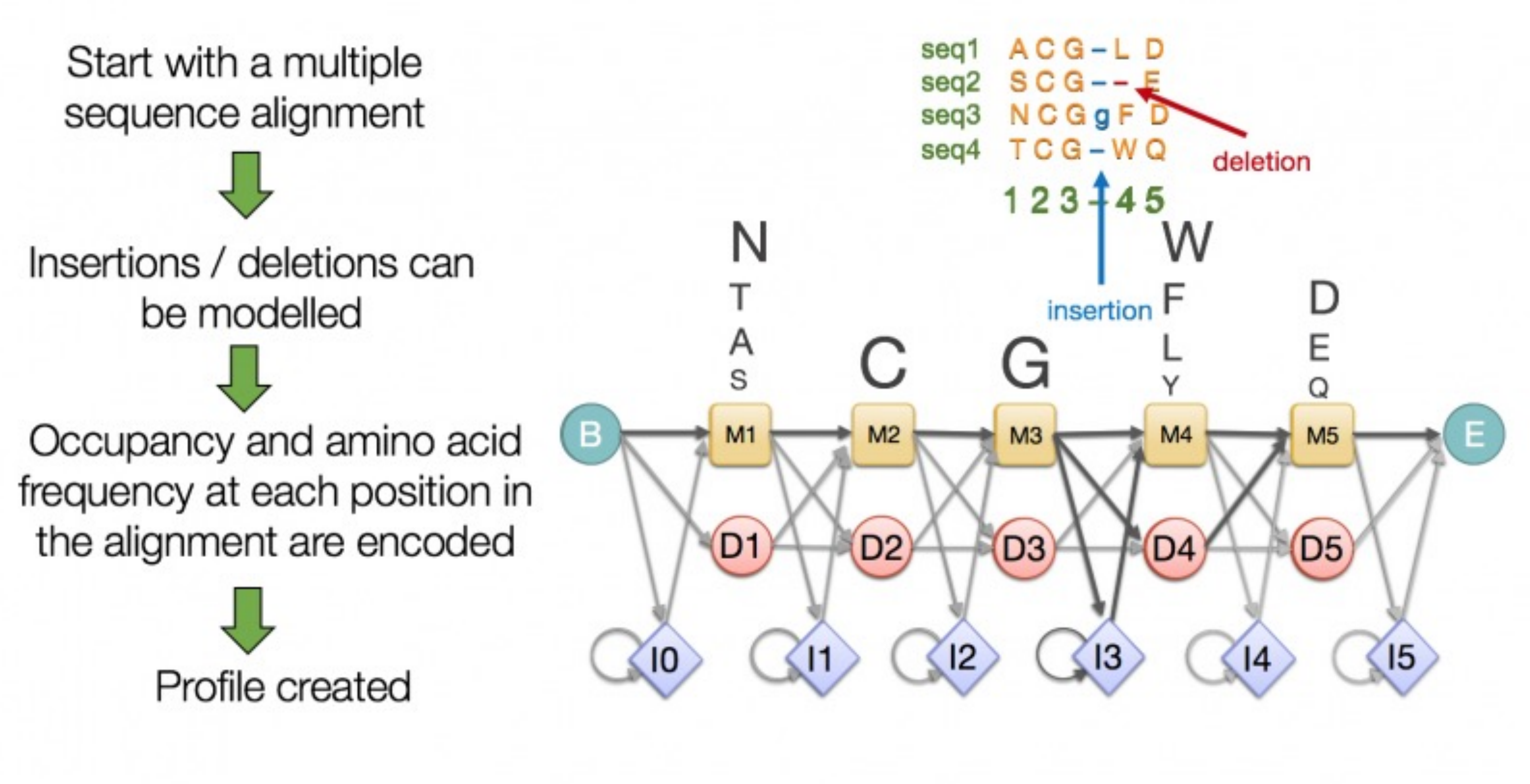
Profile created

# HMMs – modelling choices

- How many sources?
- Initialization?
  - Transition matrix helps define prior
- Optimization algorithm
  - EM algorithm

# HMMs for MSA generation

- HMMer
- Distributive learning
  - Batches
  - Bayesian learning



Start with a multiple sequence alignment

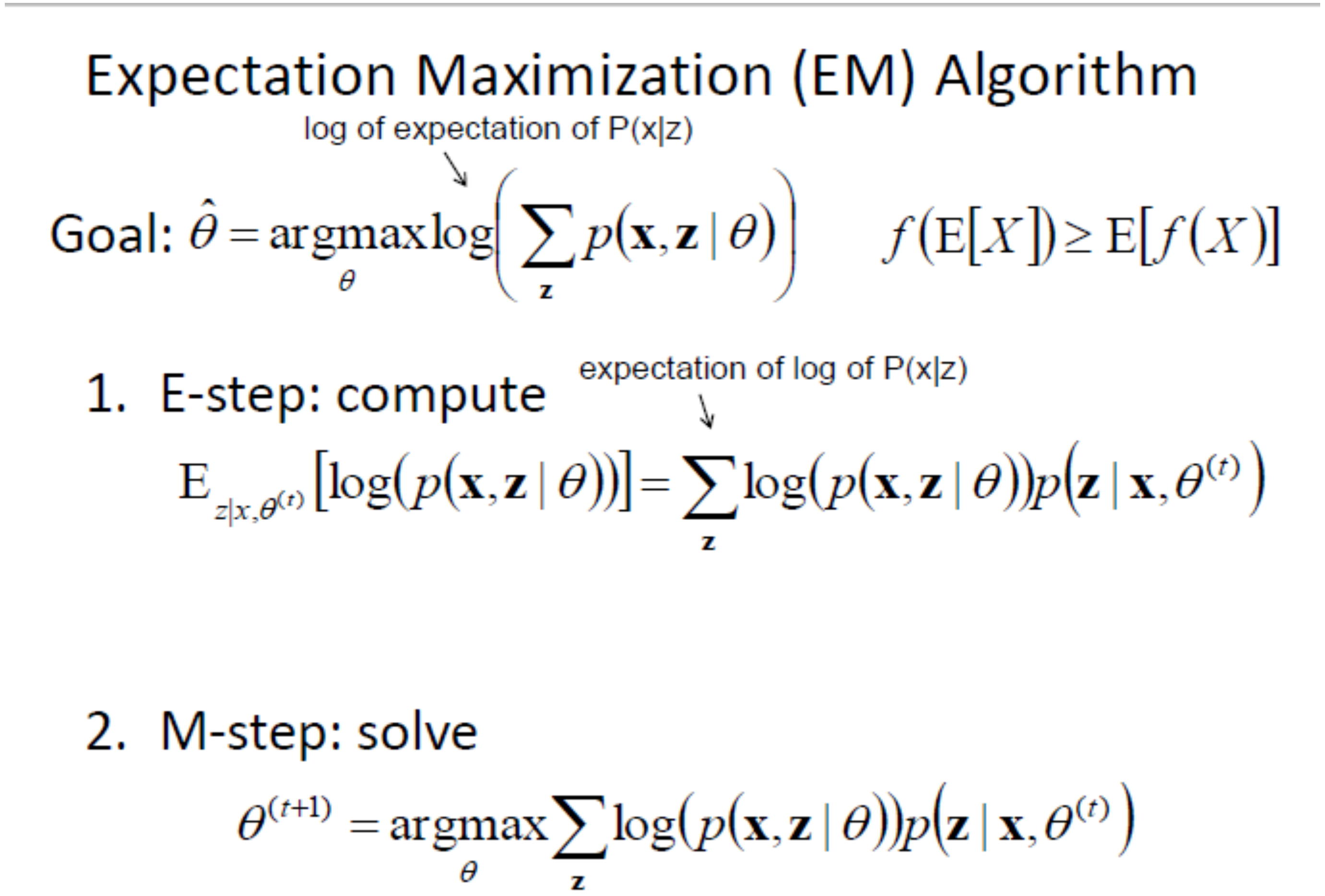
Insertions / deletions can be modelled

Occupancy and amino acid frequency at each position in the alignment are encoded

Profile created

# EM algorithm

- E-step:
  - Evaluate the expectation value of the observations
- M-step:
  - Given the source distribution, calculate the optimum parameter space



## Expectation Maximization (EM) Algorithm

log of expectation of P(x|z)

Goal: $\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \log\left( \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z} \mid \theta) \right)$ $\quad f(\mathrm{E}[X]) \geq \mathrm{E}[f(X)]$

1. E-step: compute

expectation of log of P(x|z)

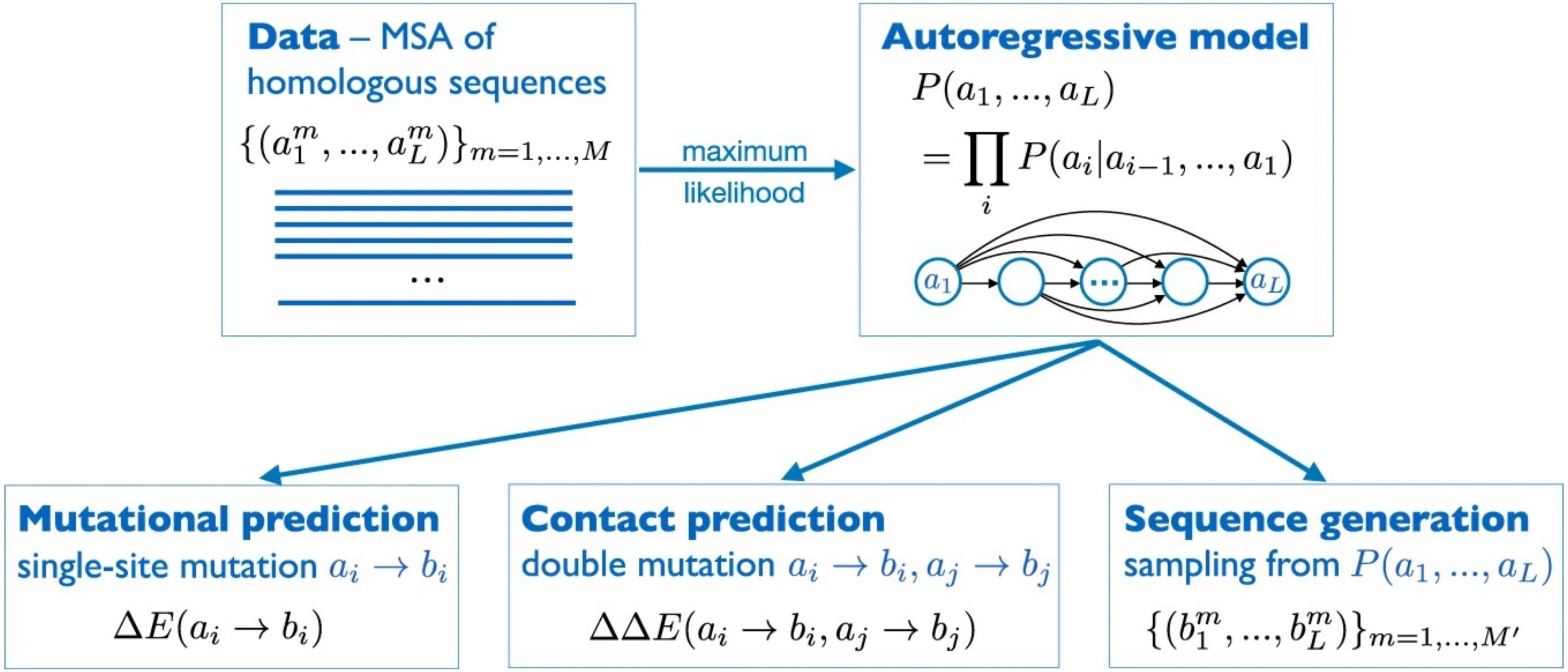
$$\mathrm{E}_{z|x,\theta^{(t)}}\left[\log(p(\mathbf{x}, \mathbf{z} \mid \theta))\right] = \sum_{\mathbf{z}} \log(p(\mathbf{x}, \mathbf{z} \mid \theta)) p\left(\mathbf{z} \mid \mathbf{x}, \theta^{(t)}\right)$$

2. M-step: solve

$$\theta^{(t+1)} = \underset{\theta}{\operatorname{argmax}} \sum_{\mathbf{z}} \log(p(\mathbf{x}, \mathbf{z} \mid \theta)) p\left(\mathbf{z} \mid \mathbf{x}, \theta^{(t)}\right)$$

# Autoregressive models

- Lab introduction



**Data** – MSA of homologous sequences

$\{(a_1^m, ..., a_L^m)\}_{m=1,...,M}$

**Autoregressive model**

$$P(a_1, ..., a_L) = \prod_i P(a_i | a_{i-1}, ..., a_1)$$

**Mutational prediction**

single-site mutation $a_i \to b_i$

$\Delta E(a_i \to b_i)$

**Contact prediction**

double mutation $a_i \to b_i, a_j \to b_j$

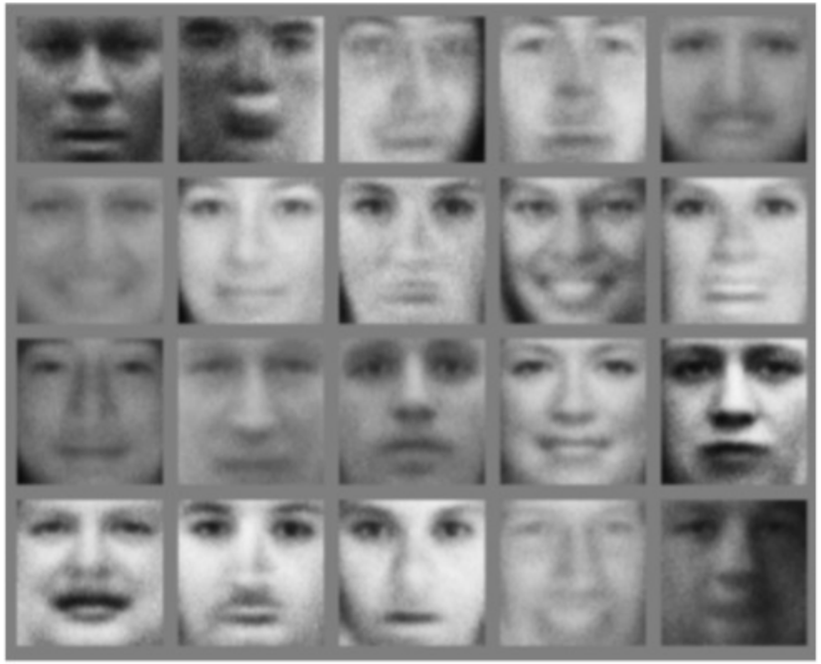
$\Delta\Delta E(a_i \to b_i, a_j \to b_j)$

**Sequence generation**

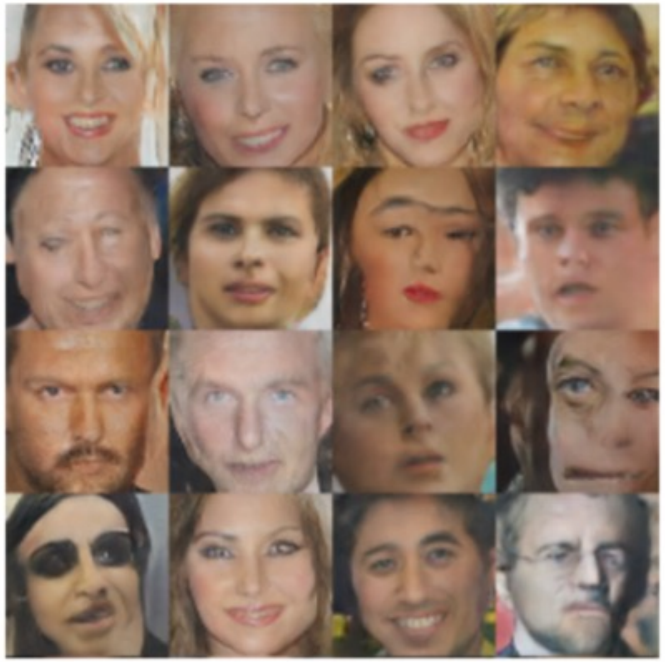
sampling from $P(a_1, ..., a_L)$

$\{(b_1^m, ..., b_L^m)\}_{m=1,...,M'}$

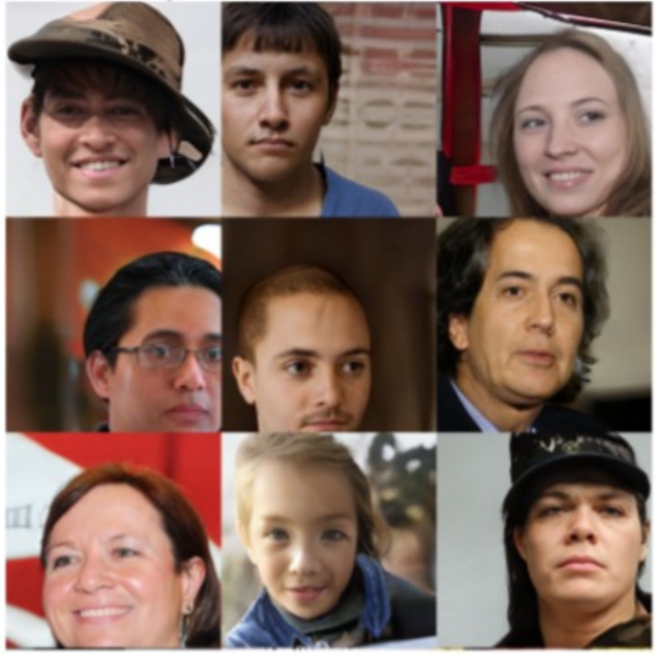
# Deep Generative ML models



GAN 2014



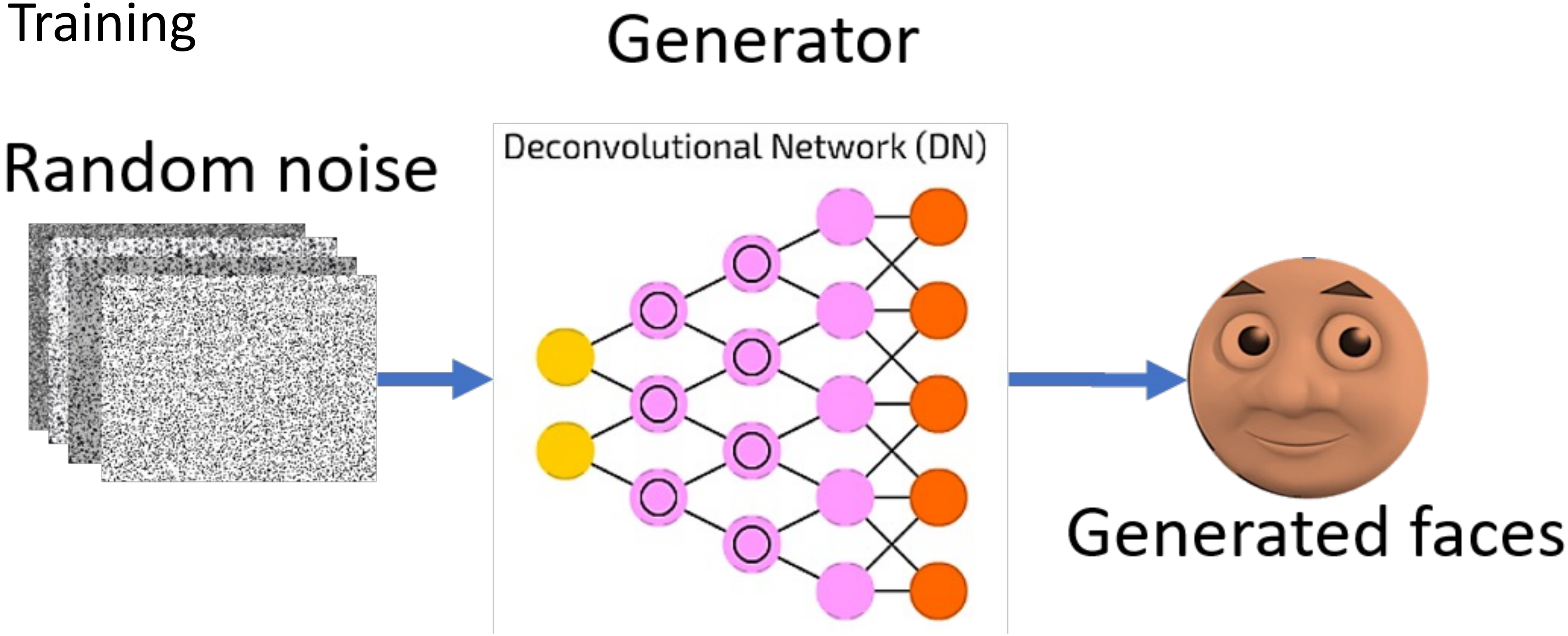
DCGAN 2016



StyleGAN3 2021

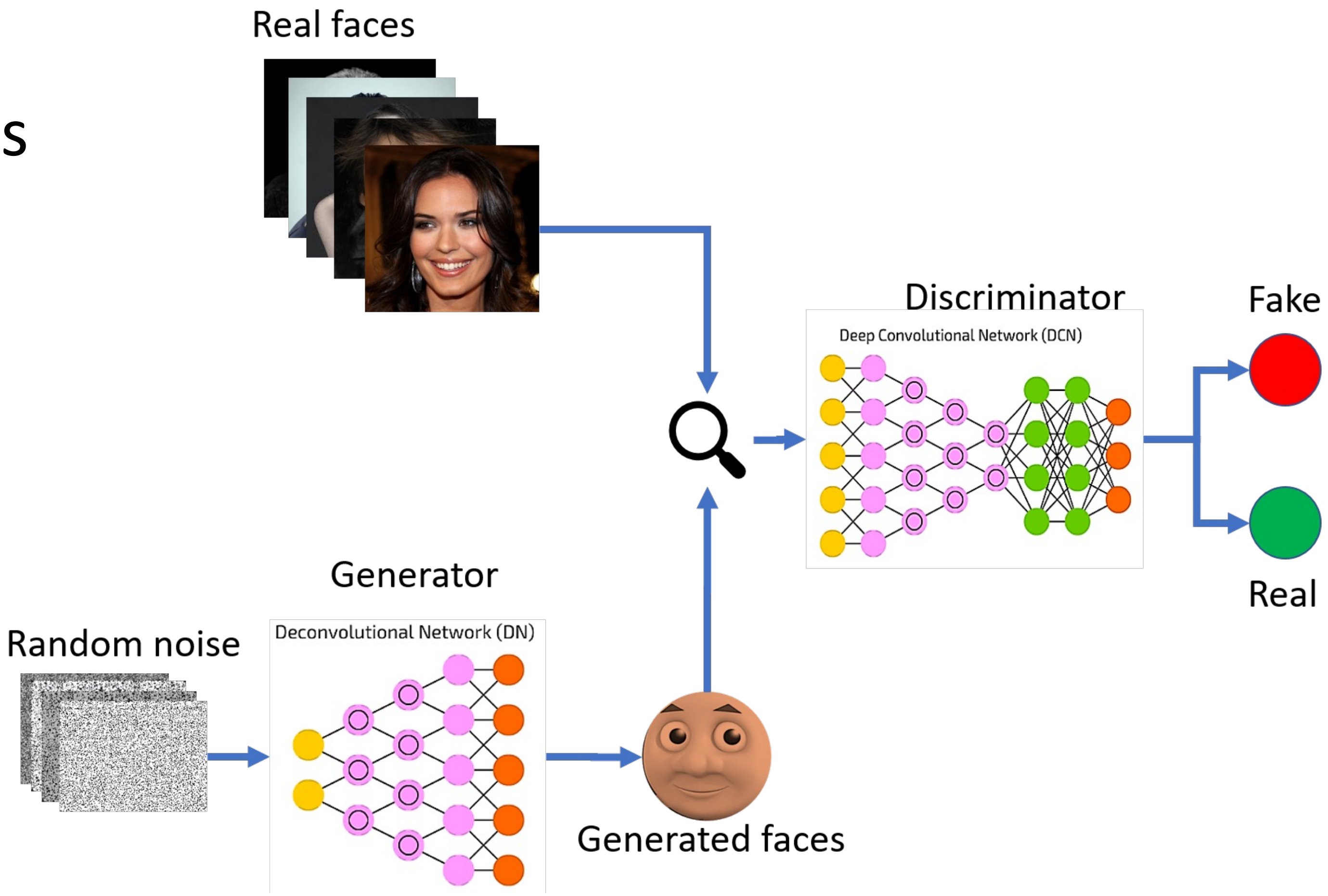
# Deep Generative ML models

- A sufficiently complex neural networks can model  any regression
- Training



# GANs

- Neural Networks
- Training



# Validation techniques

- However, as you imply, we can additionally asses the ability of the generative algorithms in modelling the underlying process that generates data. A commonly used group of metrics for this is "information theoretic scores" that derive from the idea of likelihood (log-likelihood). Below are some well-known information theoretic scores:
- 1- log-likelihood (LL) score
- 2- minimum description length (MDL) score
- 3- minimum message length (MML) score
- 4- Akaike Information Criterion (AIC) score
- 5- Bayesian Information Criterion (BIC) score
- Note that 2, 3, 4, and 5 use some complexity penalisation factor over the LL score. This is good practice to combat over-fitting.

# Applications in Life sciences

- Sequence analysis
- Face recognition
- Data augmentation
- Sample generation