Highly accurate protein structure prediction with AlphaFold

Ning Sun

High-level impact

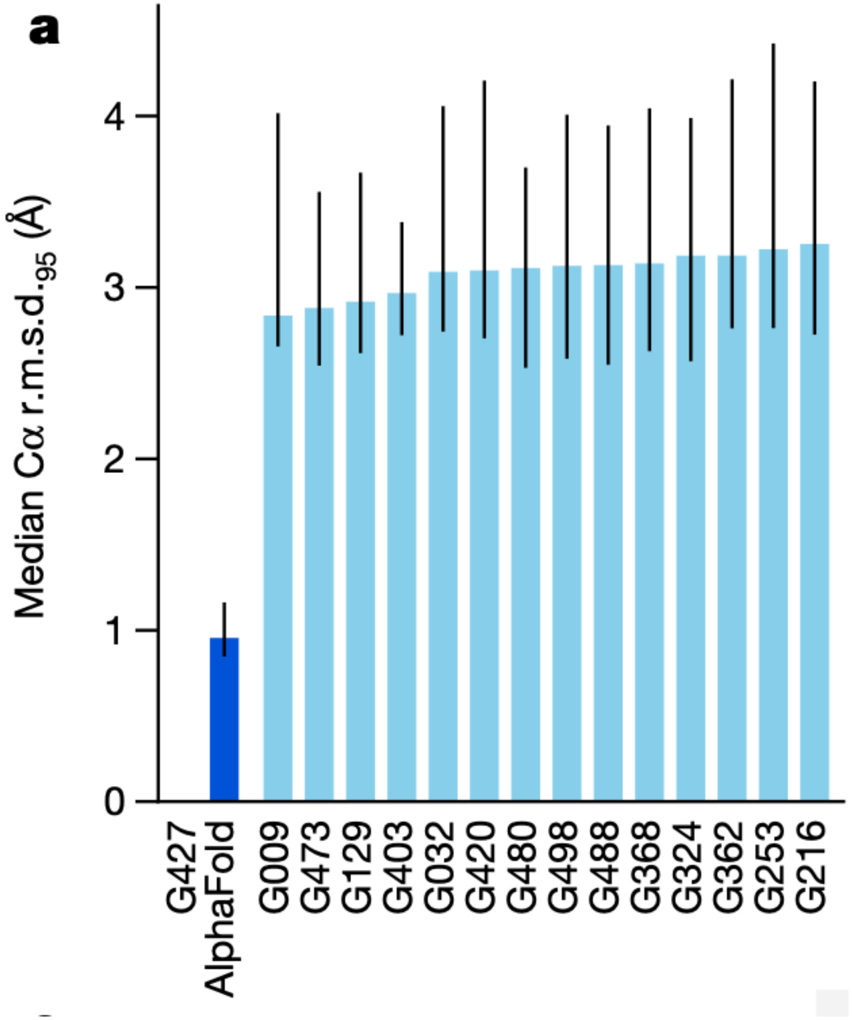
Timeline

- Dec 2018: Alphafold 1 wins CASP
    - CASP: Critical Assessment of protein Structure Prediction
- Nov 2020: Alphafold 2 solves CASP

Impact on drug discovery

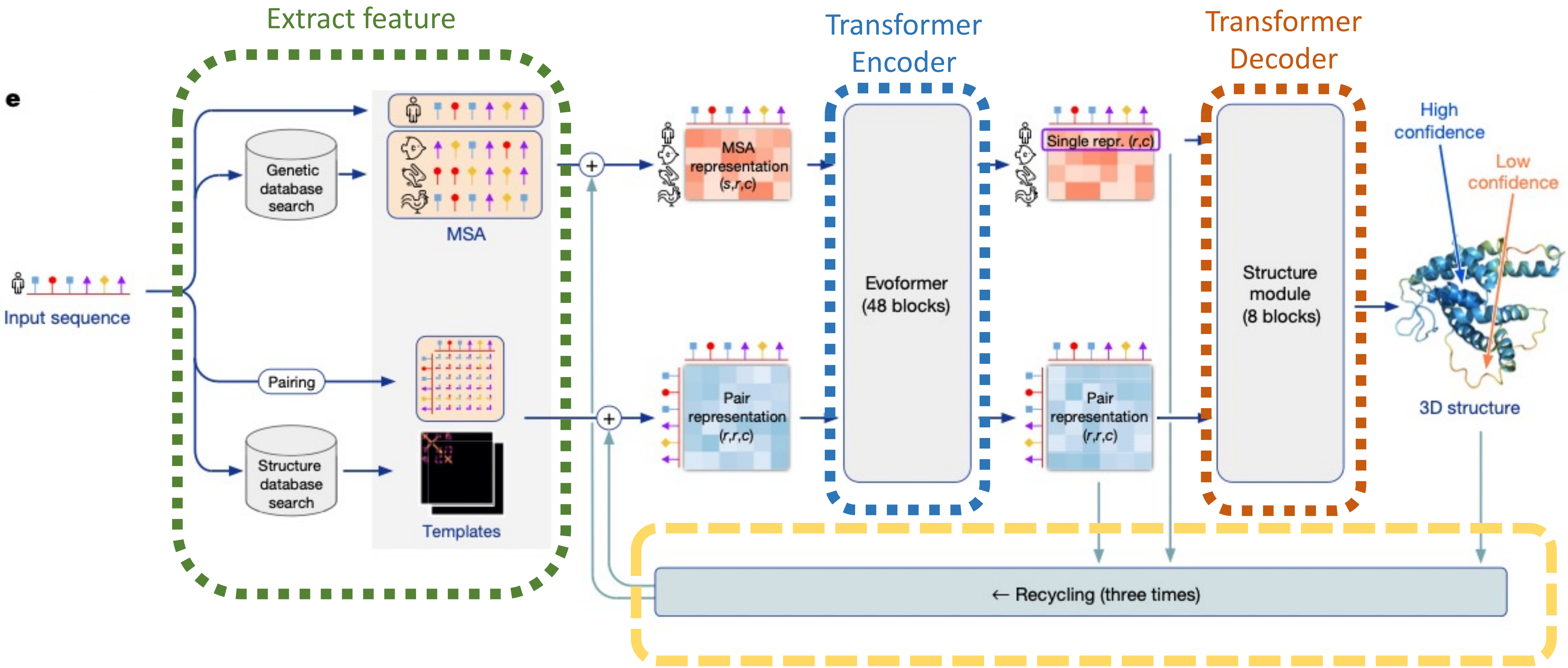
- Universal end-to-end molecular drug discovery pipeline now available

Performance at CASP



First computational method that predict protein structures with atomic accuracy.

Alphafold2



Extract feature
Transformer Encoder
Transformer Decoder
e
Genetic database search
MSA
Input sequence
Pairing
Structure database search
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)

Data Preprocessing

Dataset: 25% PDB dataset + 75% self-distillation Uniclust30 dataset(unlabeled)

MSA: multiple sequence alignments

Template: 3D atom coordinates of homologous structures

MSA Preprocess:

- Filter
- MSA Block Deletion
- MSA Clustering (cluster_profile, extra_msa, mask_msa)
- Residue Cropping(training)

Training & Inference details

MSA resample and ensemble

- MSA block deletion and clustering is stochastic method
- Resampling at training and inference time
- Ensemble at inference time

Recycling

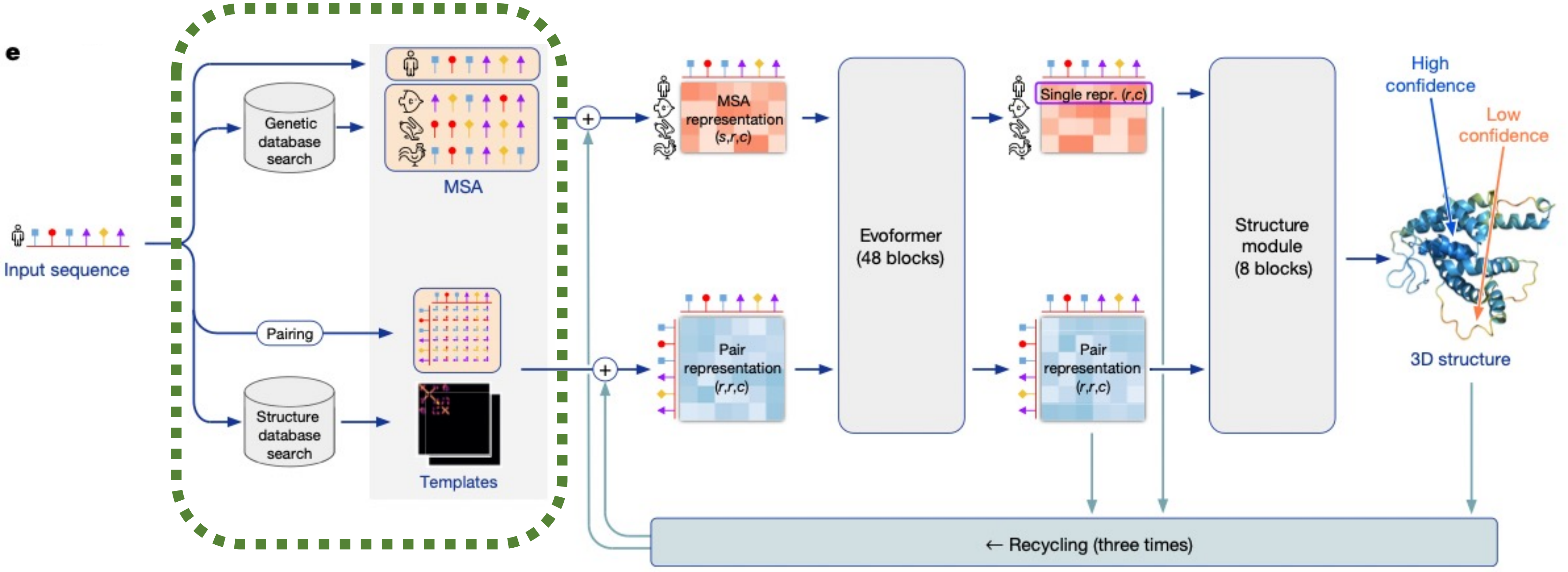
- Random $\mathrm{N} \in [0, N_{cycle}]$ at training time
- Fixed N_{cycle} at inference time

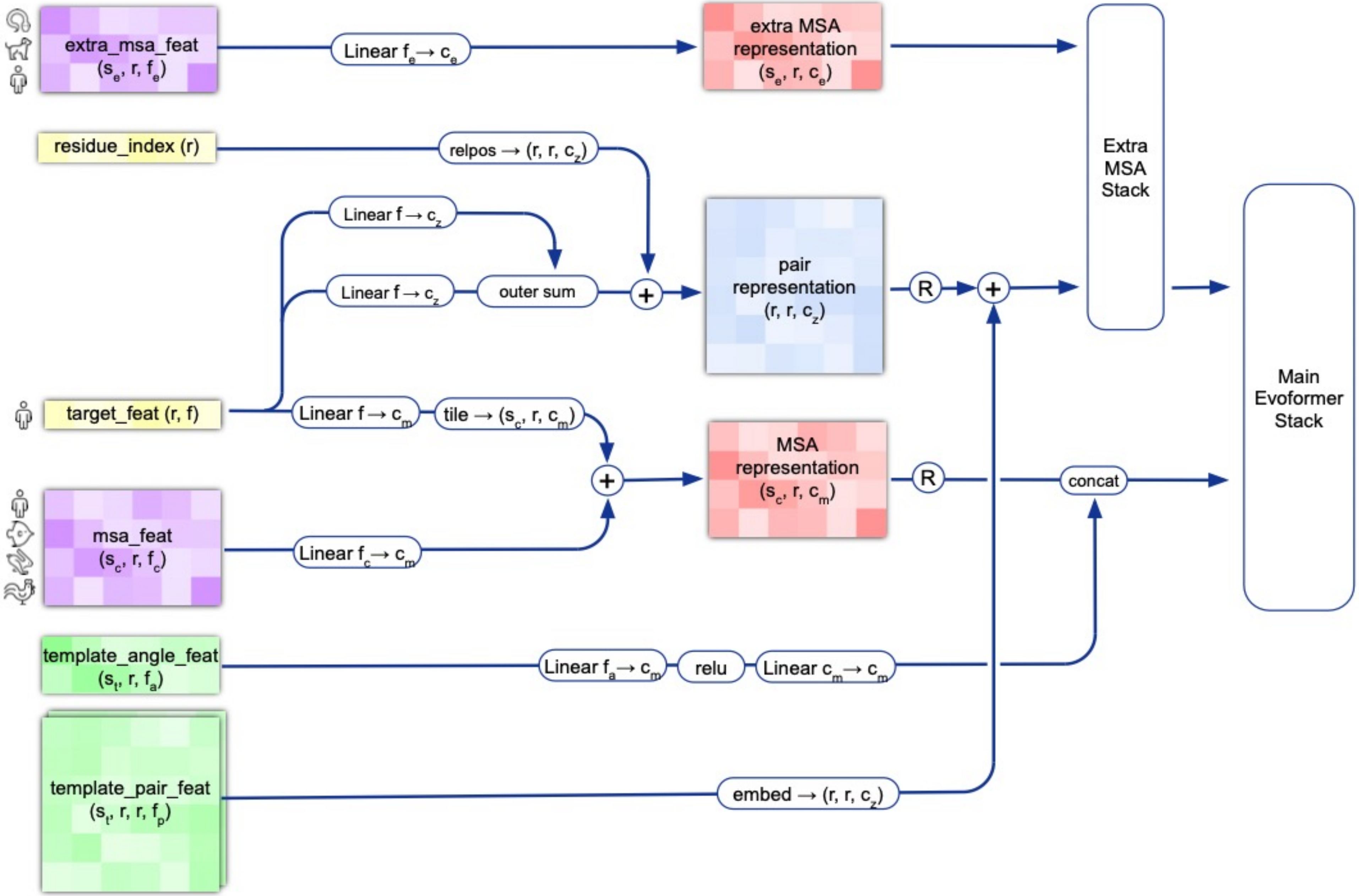
Reducing the memory consumption

- gradient checkpoint at training time
- ‘chunk’ at inference time

Alphafold2

Extract feature





extra_msa_feat
(s_e, r, f_e)
Linear $f_e \rightarrow c_e$
extra MSA representation
(s_e, r, c_e)
residue_index (r)
relpos $\rightarrow (r, r, c_z)$
Linear $f \rightarrow c_z$
Linear $f \rightarrow c_z$
outer sum
pair representation
(r, r, c_z)
R
Extra MSA Stack
target_feat (r, f)
Linear $f \rightarrow c_m$
tile $\rightarrow (s_c, r, c_m)$
MSA representation
(s_c, r, c_m)
R
concat
Main Evoformer Stack
msa_feat
(s_c, r, f_c)
Linear $f_c \rightarrow c_m$
template_angle_feat
(s_t, r, f_a)
Linear $f_a \rightarrow c_m$
relu
Linear $c_m \rightarrow c_m$
template_pair_feat
(s_t, r, r, f_p)
embed $\rightarrow (r, r, c_z)$

Algorithm 4 Relative position encoding

def $\text{relpos}(\{f_i^{\text{residue\_index}}\}, \mathbf{v}_{\text{bins}} = [-32, -31, \ldots, 32]) :$

$d_{ij} = f_i^{\text{residue\_index}} - f_j^{\text{residue\_index}}$ $\qquad d_{ij} \in \mathbb{Z}$

$\mathbf{p}_{ij} = \text{Linear}(\text{one\_hot}(d_{ij}, \mathbf{v}_{\text{bins}}))$ $\qquad \mathbf{p}_{ij} \in \mathbb{R}^{c_z}$

return $\{\mathbf{p}_{ij}\}$

Algorithm 17 Template pointwise attention

def $\text{TemplatePointwiseAttention}(\{\mathbf{t}_{s_t ij}\}, \{\mathbf{z}_{ij}\}, c = 64, N_{\text{head}} = 4) :$

$\mathbf{q}_{ij}^h = \text{LinearNoBias}(\mathbf{z}_{ij})$ $\qquad \mathbf{q}_{ij}^h \in \mathbb{R}^c,\ h \in \{1, \ldots, N_{\text{head}}\}$

$\mathbf{k}_{s_t ij}^h, \mathbf{v}_{s_t ij}^h = \text{LinearNoBias}(\mathbf{t}_{s_t ij})$ $\qquad \mathbf{k}_{s_t ij}^h, \mathbf{v}_{s_t ij}^h \in \mathbb{R}^c$

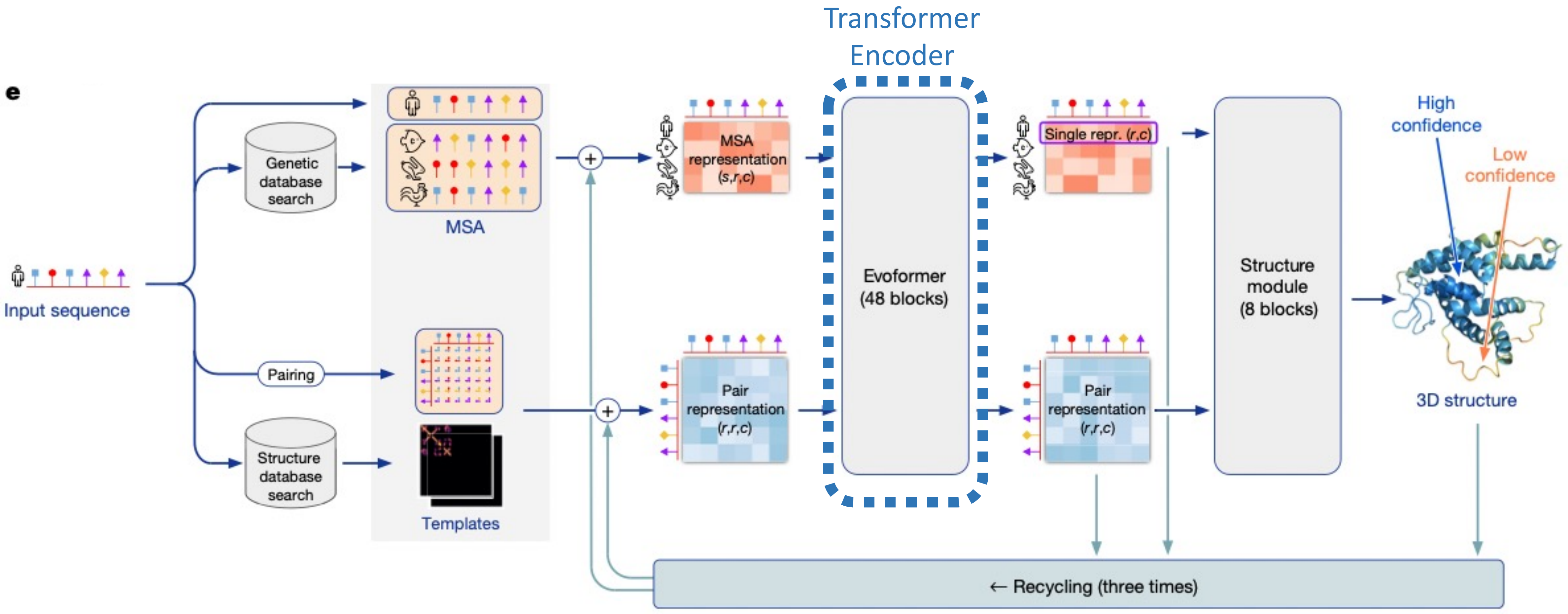
$a_{s_t ij}^h = \text{softmax}_{s_t}\left(\frac{1}{\sqrt{c}}\, {\mathbf{q}_{ij}^h}^\top \mathbf{k}_{s_t ij}^h\right)$

$\mathbf{o}_{ij}^h = \sum_{s_t} a_{s_t ij}^h \mathbf{v}_{s_t ij}^h$

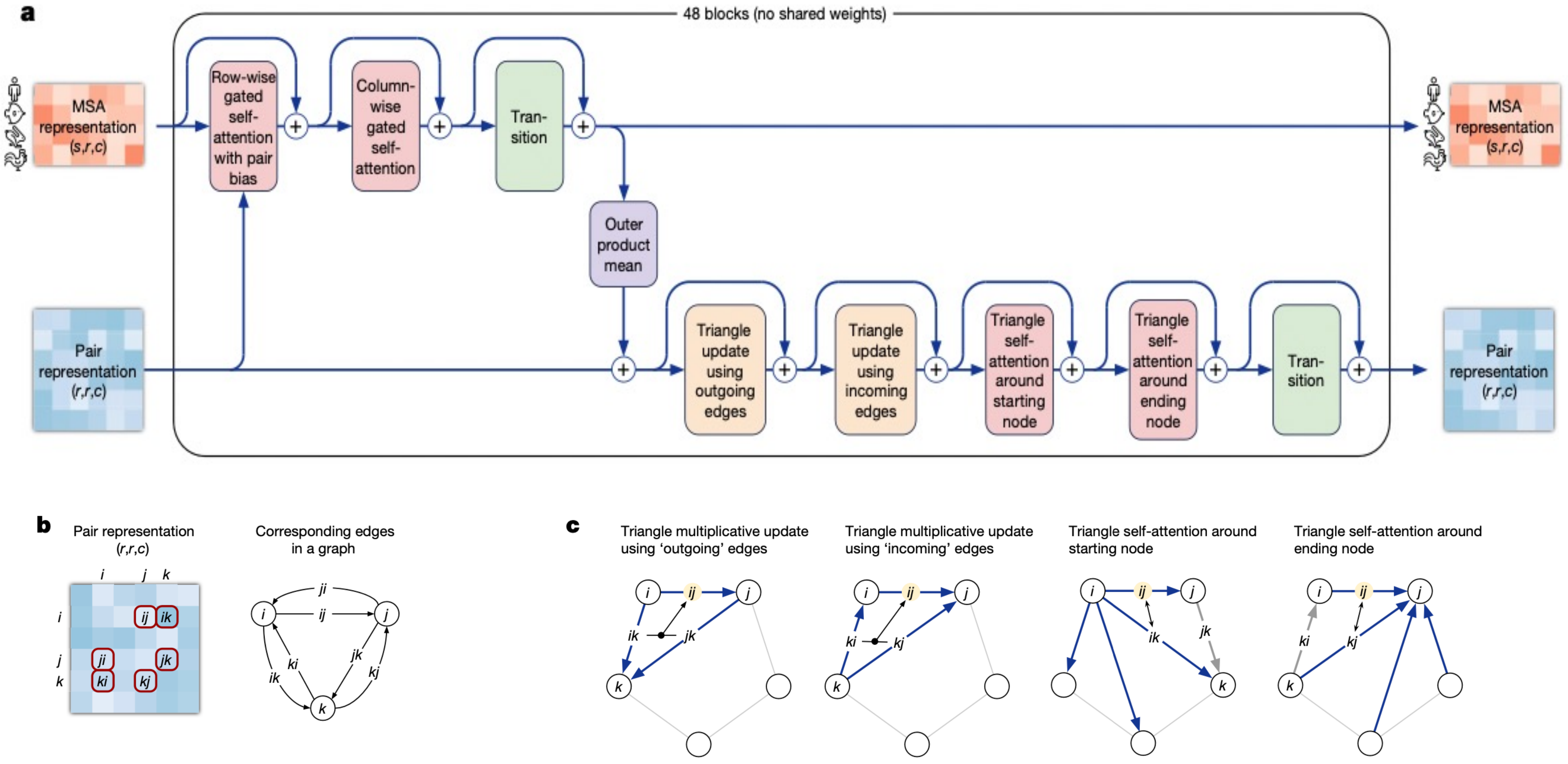
$\{\tilde{\mathbf{z}}_{ij}\} = \text{Linear}\left(\text{concat}_h(\{\mathbf{o}_{ij}^h\})\right)$ $\qquad \tilde{\mathbf{z}}_{ij} \in \mathbb{R}^{c_z}$

return $\{\tilde{\mathbf{z}}_{ij}\}$

Alphafold2

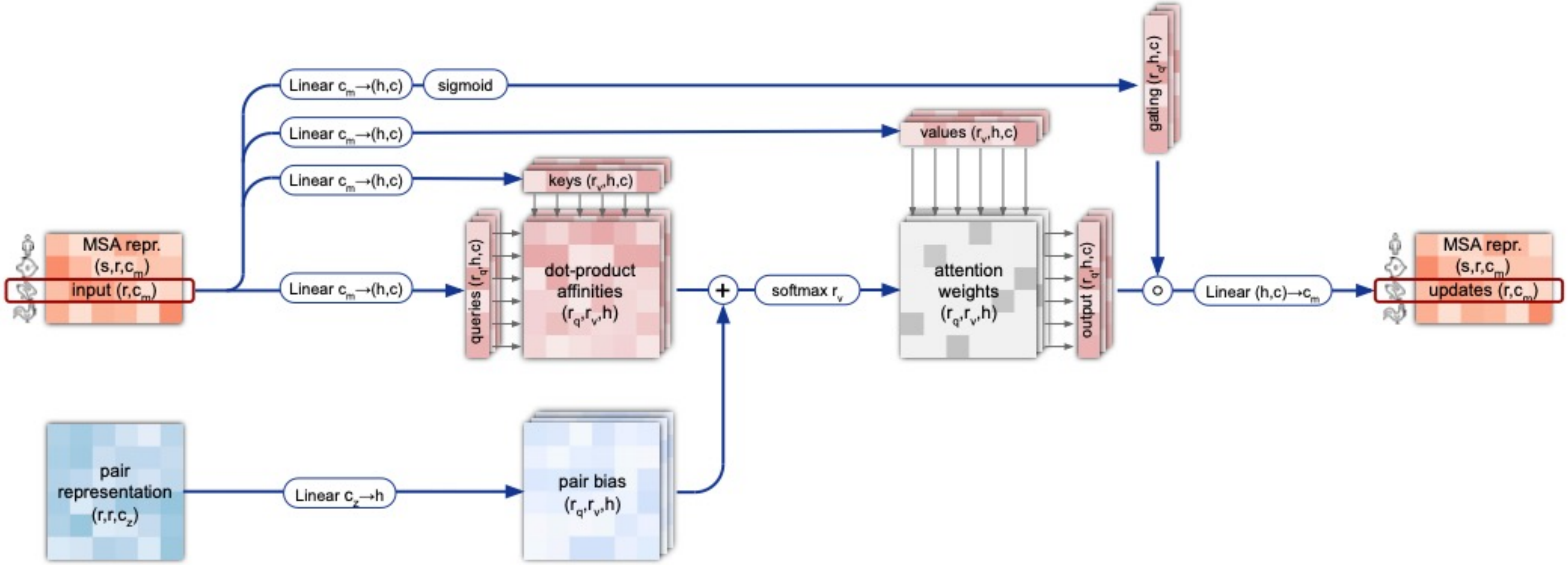

Transformer
Encoder
e
Input sequence
Genetic database search
MSA
Pairing
Structure database search
Templates
MSA representation (s,r,c)
Pair representation (r,r,c)
Evoformer (48 blocks)
Single repr. (r,c)
Pair representation (r,r,c)
Structure module (8 blocks)
High confidence
Low confidence
3D structure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

Evoformer



a
48 blocks (no shared weights)
MSA representation (s,r,c)
Row-wise gated self-attention with pair bias
Column-wise gated self-attention
Tran-sition
Outer product mean
Pair representation (r,r,c)
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)
b
Pair representation (r,r,c)
Corresponding edges in a graph
c
Triangle multiplicative update using 'outgoing' edges
Triangle multiplicative update using 'incoming' edges
Triangle self-attention around starting node
Triangle self-attention around ending node

Row-wise gated self-attention



Supplementary Figure 2 | MSA row-wise gated self-attention with pair bias. Dimensions: s: sequences, r: residues, c: channels, h: heads.

Algorithm 7 MSA row-wise gated self-attention with pair bias

def MSARowAttentionWithPairBias($\{\mathbf{m}_{si}\}, \{\mathbf{z}_{ij}\}, c = 32, N_{\text{head}} = 8$) :

 # Input projections

1: $\mathbf{m}_{si} \leftarrow \text{LayerNorm}(\mathbf{m}_{si})$

2: $\mathbf{q}_{si}^h, \mathbf{k}_{si}^h, \mathbf{v}_{si}^h = \text{LinearNoBias}(\mathbf{m}_{si})$ $\quad\quad$ $\mathbf{q}_{si}^h, \mathbf{k}_{si}^h, \mathbf{v}_{si}^h \in \mathbb{R}^c,\ h \in \{1, \ldots, N_{\text{head}}\}$

3: $b_{ij}^h = \text{LinearNoBias}(\text{LayerNorm}(\mathbf{z}_{ij}))$

4: $\mathbf{g}_{si}^h = \text{sigmoid}\left(\text{Linear}(\mathbf{m}_{si})\right)$ $\quad\quad$ $\mathbf{g}_{si}^h \in \mathbb{R}^c$

 # Attention

5: $a_{sij}^h = \text{softmax}_j \left(\frac{1}{\sqrt{c}}\, \mathbf{q}_{si}^{h\top} \mathbf{k}_{sj}^h + b_{ij}^h \right)$

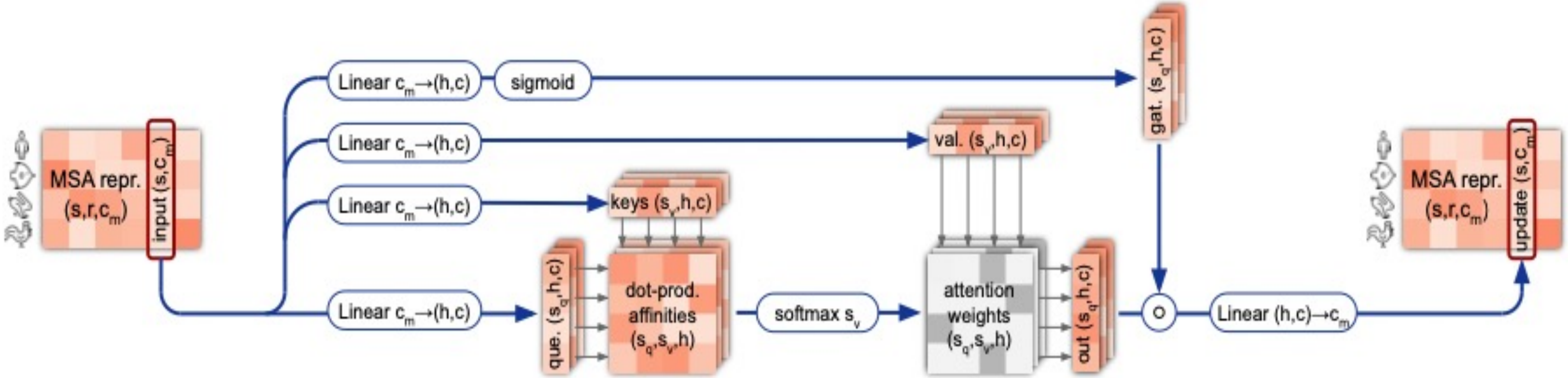
6: $\mathbf{o}_{si}^h = \mathbf{g}_{si}^h \odot \sum_j a_{sij}^h \mathbf{v}_{sj}^h$

 # Output projection

7: $\tilde{\mathbf{m}}_{si} = \text{Linear}\left(\text{concat}_h(\mathbf{o}_{si}^h)\right)$ $\quad\quad$ $\tilde{\mathbf{m}}_{si} \in \mathbb{R}^{c_m}$

8: **return** $\{\tilde{\mathbf{m}}_{si}\}$

Column-wise gated self-attention



Supplementary Figure 3 | MSA column-wise gated self-attention. Dimensions: s: sequences, r: residues, c: channels, h: heads.

Algorithm 8 MSA column-wise gated self-attention

def $\text{MSAColumnAttention}(\{\mathbf{m}_{si}\}, c = 32, N_{\text{head}} = 8)$:

 # Input projections

1: $\mathbf{m}_{si} \leftarrow \text{LayerNorm}(\mathbf{m}_{si})$

2: $\mathbf{q}_{si}^h, \mathbf{k}_{si}^h, \mathbf{v}_{si}^h = \text{LinearNoBias}(\mathbf{m}_{si})$ $\qquad \mathbf{q}_{si}^h, \mathbf{k}_{si}^h, \mathbf{v}_{si}^h \in \mathbb{R}^c,\ h \in \{1, \dots, N_{\text{head}}\}$

3: $\mathbf{g}_{si}^h = \text{sigmoid}\left(\text{Linear}(\mathbf{m}_{si})\right)$ $\qquad \mathbf{g}_{si}^h \in \mathbb{R}^c$

 # Attention

4: $a_{sti}^h = \text{softmax}_t\left(\frac{1}{\sqrt{c}}\, {\mathbf{q}_{si}^h}^\top \mathbf{k}_{ti}^h\right)$

5: $\mathbf{o}_{si}^h = \mathbf{g}_{si}^h \odot \sum_t a_{sti}^h \mathbf{v}_{st}^h$

 # Output projection

6: $\tilde{\mathbf{m}}_{si} = \text{Linear}\left(\text{concat}_h(\mathbf{o}_{si}^h)\right)$ $\qquad \tilde{\mathbf{m}}_{si} \in \mathbb{R}^{c_m}$

7: **return** $\{\tilde{\mathbf{m}}_{si}\}$

Algorithm 13 Triangular gated self-attention around starting node

def $\mathrm{TriangleAttentionStartingNode}(\{\mathbf{z}_{ij}\}, c = 32, N_{\text{head}} = 4):$

 # Input projections

$\mathbf{z}_{ij} \leftarrow \mathrm{LayerNorm}(\mathbf{z}_{ij})$

$\mathbf{q}_{ij}^h, \mathbf{k}_{ij}^h, \mathbf{v}_{ij}^h = \mathrm{LinearNoBias}(\mathbf{z}_{ij})$ $\qquad \mathbf{q}_{ij}^h, \mathbf{k}_{ij}^h, \mathbf{v}_{ij}^h \in \mathbb{R}^c,\ h \in \{1, \ldots, N_{\text{head}}\}$

$b_{ij}^h = \mathrm{LinearNoBias}(\mathbf{z}_{ij})$

$\mathbf{g}_{ij}^h = \mathrm{sigmoid}\left(\mathrm{Linear}(\mathbf{z}_{ij})\right)$ $\qquad \mathbf{g}_{ij}^h \in \mathbb{R}^c$

 # Attention

$a_{ijk}^h = \mathrm{softmax}_k\left(\frac{1}{\sqrt{c}}\, {\mathbf{q}_{ij}^h}^\top \mathbf{k}_{ik}^h + b_{jk}^h\right)$

$\mathbf{o}_{ij}^h = \mathbf{g}_{ij}^h \odot \sum_k a_{ijk}^h \mathbf{v}_{ik}^h$

 # Output projection

$\tilde{\mathbf{z}}_{ij} = \mathrm{Linear}\left(\mathrm{concat}_h(\mathbf{o}_{ij}^h)\right)$ $\qquad \tilde{\mathbf{z}}_{ij} \in \mathbb{R}^{c_z}$

return $\{\tilde{\mathbf{z}}_{ij}\}$

Algorithm 14 Triangular gated self-attention around ending node

def TriangleAttentionEndingNode($\{\mathbf{z}_{ij}\}, c = 32, N_{\text{head}} = 4$) :

 # Input projections

1: $\mathbf{z}_{ij} \leftarrow \text{LayerNorm}(\mathbf{z}_{ij})$

2: $\mathbf{q}_{ij}^h, \mathbf{k}_{ij}^h, \mathbf{v}_{ij}^h = \text{LinearNoBias}(\mathbf{z}_{ij})$ $\qquad \mathbf{q}_{ij}^h, \mathbf{k}_{ij}^h, \mathbf{v}_{ij}^h \in \mathbb{R}^c, \ h \in \{1, \dots, N_{\text{head}}\}$

3: $b_{ij}^h = \text{LinearNoBias}(\mathbf{z}_{ij})$

4: $\mathbf{g}_{ij}^h = \text{sigmoid}\left(\text{Linear}(\mathbf{z}_{ij})\right)$ $\qquad \mathbf{g}_{ij}^h \in \mathbb{R}^c$

 # Attention

5: $a_{ijk}^h = \text{softmax}_k \left(\frac{1}{\sqrt{c}} \, {\mathbf{q}_{ij}^h}^\top \mathbf{k}_{kj}^h + b_{ki}^h \right)$

6: $\mathbf{o}_{ij}^h = \mathbf{g}_{ij}^h \odot \sum_k a_{ijk}^h \mathbf{v}_{kj}^h$

 # Output projection

7: $\tilde{\mathbf{z}}_{ij} = \text{Linear}\left(\text{concat}_h \left(\mathbf{o}_{ij}^h\right)\right)$ $\qquad \tilde{\mathbf{z}}_{ij} \in \mathbb{R}^{c_z}$

8: **return** $\{\tilde{\mathbf{z}}_{ij}\}$

Algorithm 11 Triangular multiplicative update using "outgoing" edges

def TriangleMultiplicationOutgoing($\{\mathbf{z}_{ij}\}, c = 128$) :

1: $\mathbf{z}_{ij} \leftarrow \text{LayerNorm}(\mathbf{z}_{ij})$

2: $\mathbf{a}_{ij}, \mathbf{b}_{ij} = \text{sigmoid}\left(\text{Linear}(\mathbf{z}_{ij})\right) \odot \text{Linear}(\mathbf{z}_{ij})$ $\qquad \mathbf{a}_{ij}, \mathbf{b}_{ij} \in \mathbb{R}^{c}$

3: $\mathbf{g}_{ij} = \text{sigmoid}\left(\text{Linear}(\mathbf{z}_{ij})\right)$ $\qquad \mathbf{g}_{ij} \in \mathbb{R}^{c_z}$

4: $\tilde{\mathbf{z}}_{ij} = \mathbf{g}_{ij} \odot \text{Linear}(\text{LayerNorm}(\sum_k \mathbf{a}_{ik} \odot \mathbf{b}_{jk}))$ $\qquad \tilde{\mathbf{z}}_{ij} \in \mathbb{R}^{c_z}$

5: **return** $\{\tilde{\mathbf{z}}_{ij}\}$

Algorithm 12 Triangular multiplicative update using "incoming" edges

def TriangleMultiplicationIncoming($\{\mathbf{z}_{ij}\}, c = 128$) :

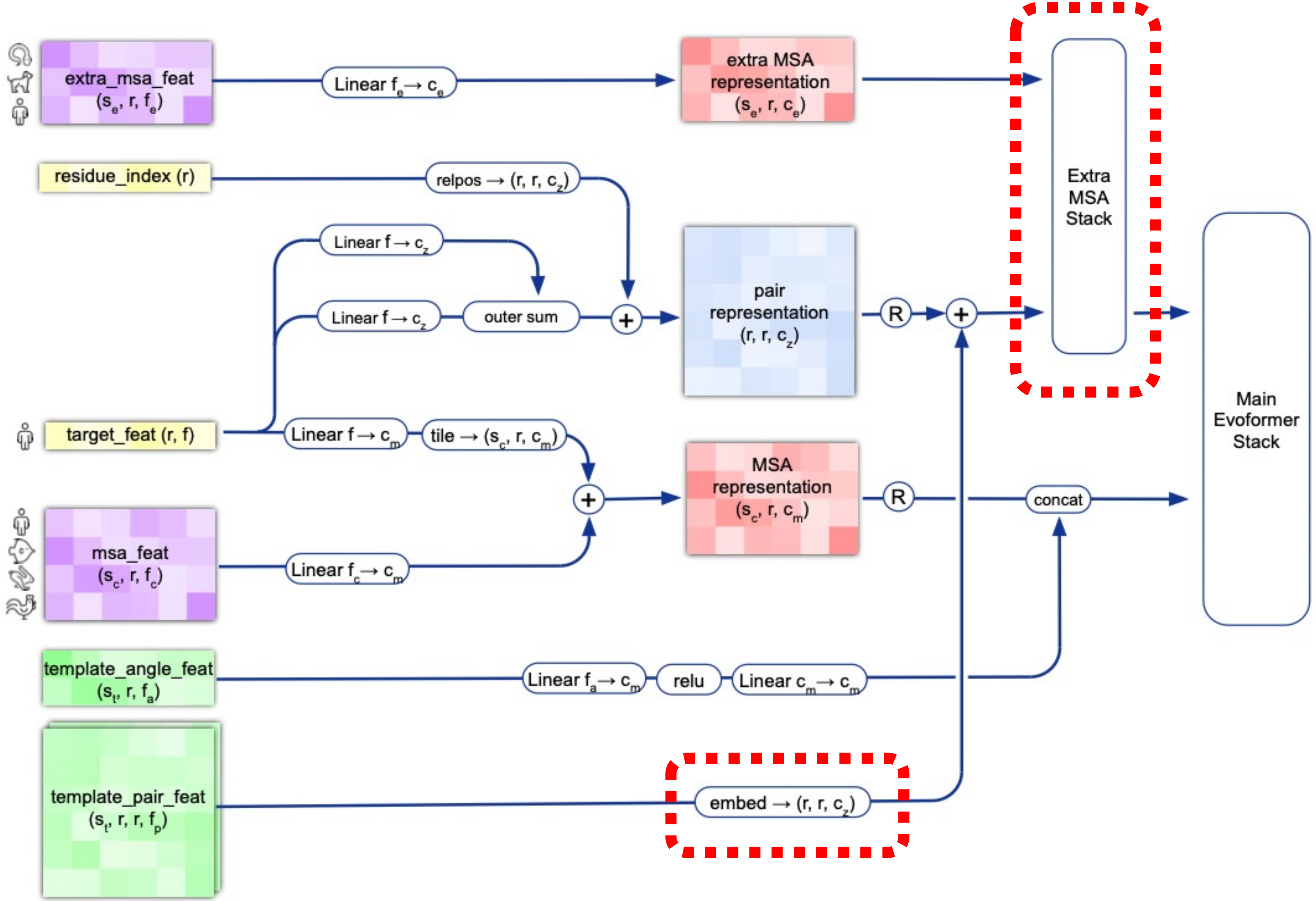
1: $\mathbf{z}_{ij} \leftarrow \text{LayerNorm}(\mathbf{z}_{ij})$

2: $\mathbf{a}_{ij}, \mathbf{b}_{ij} = \text{sigmoid}\left(\text{Linear}(\mathbf{z}_{ij})\right) \odot \text{Linear}(\mathbf{z}_{ij})$ $\qquad \mathbf{a}_{ij}, \mathbf{b}_{ij} \in \mathbb{R}^{c}$

3: $\mathbf{g}_{ij} = \text{sigmoid}\left(\text{Linear}(\mathbf{z}_{ij})\right)$ $\qquad \mathbf{g}_{ij} \in \mathbb{R}^{c_z}$

4: $\tilde{\mathbf{z}}_{ij} = \mathbf{g}_{ij} \odot \text{Linear}(\text{LayerNorm}(\sum_k \mathbf{a}_{ki} \odot \mathbf{b}_{kj}))$ $\qquad \tilde{\mathbf{z}}_{ij} \in \mathbb{R}^{c_z}$

5: **return** $\{\tilde{\mathbf{z}}_{ij}\}$


extra_msa_feat
(s_e, r, f_e)
Linear $f_e \rightarrow c_e$
extra MSA representation
(s_e, r, c_e)
Extra MSA Stack
residue_index (r)
relpos → (r, r, c_z)
Linear $f \rightarrow c_z$
Linear $f \rightarrow c_z$
outer sum
pair representation
(r, r, c_z)
R
Main Evoformer Stack
target_feat (r, f)
Linear $f \rightarrow c_m$
tile → (s_c, r, c_m)
MSA representation
(s_c, r, c_m)
R
concat
msa_feat
(s_c, r, f_c)
Linear $f_c \rightarrow c_m$
template_angle_feat
(s_t, r, f_a)
Linear $f_a \rightarrow c_m$
relu
Linear $c_m \rightarrow c_m$
template_pair_feat
(s_t, r, r, f_p)
embed → (r, r, c_z)

Algorithm 16 Template pair stack

def TemplatePairStack$(\{\mathbf{t}_{ij}\}, N_{\text{block}} = 2)$:

```
for all l ∈ [1, ..., N_block] do
    {t_ij} += DropoutRowwise_0.25(TriangleAttentionStartingNode({t_ij}, c = 64, N_head = 4))
    {t_ij} += DropoutColumnwise_0.25(TriangleAttentionEndingNode({t_ij}, c = 64, N_head = 4))
    {t_ij} += DropoutRowwise_0.25(TriangleMultiplicationOutgoing({t_ij}, c = 64))
    {t_ij} += DropoutRowwise_0.25(TriangleMultiplicationIncoming({t_ij}, c = 64))
    {t_ij} += PairTransition({t_ij}, n = 2)
end for
return LayerNorm({t_ij})
```

Algorithm 18 Extra MSA stack

def $\text{ExtraMsaStack}(\{\mathbf{e}_{s_e i}\}, \{\mathbf{z}_{ij}\}, N_{\text{block}} = 4):$

for all $l \in [1, \ldots, N_{\text{block}}]$ **do**

  # MSA stack

  $\{\mathbf{e}_{s_e i}\} \mathrel{+}= \text{DropoutRowwise}_{0.15}(\text{MSARowAttentionWithPairBias}(\{\mathbf{e}_{s_e i}\}, \{\mathbf{z}_{ij}\}, c = 8))$

  $\{\mathbf{e}_{s_e i}\} \mathrel{+}= \text{MSAColumnGlobalAttention}(\{\mathbf{e}_{s_e i}\})$

  $\{\mathbf{e}_{s_e i}\} \mathrel{+}= \text{MSATransition}(\{\mathbf{e}_{s_e i}\})$

  # Communication

  $\{\mathbf{z}_{ij}\} \mathrel{+}= \text{OuterProductMean}(\{\mathbf{e}_{s_e i}\})$

  # Pair stack

  $\{\mathbf{z}_{ij}\} \mathrel{+}= \text{DropoutRowwise}_{0.25}(\text{TriangleMultiplicationOutgoing}(\{\mathbf{z}_{ij}\}))$

  $\{\mathbf{z}_{ij}\} \mathrel{+}= \text{DropoutRowwise}_{0.25}(\text{TriangleMultiplicationIncoming}(\{\mathbf{z}_{ij}\}))$

  $\{\mathbf{z}_{ij}\} \mathrel{+}= \text{DropoutRowwise}_{0.25}(\text{TriangleAttentionStartingNode}(\{\mathbf{z}_{ij}\}))$

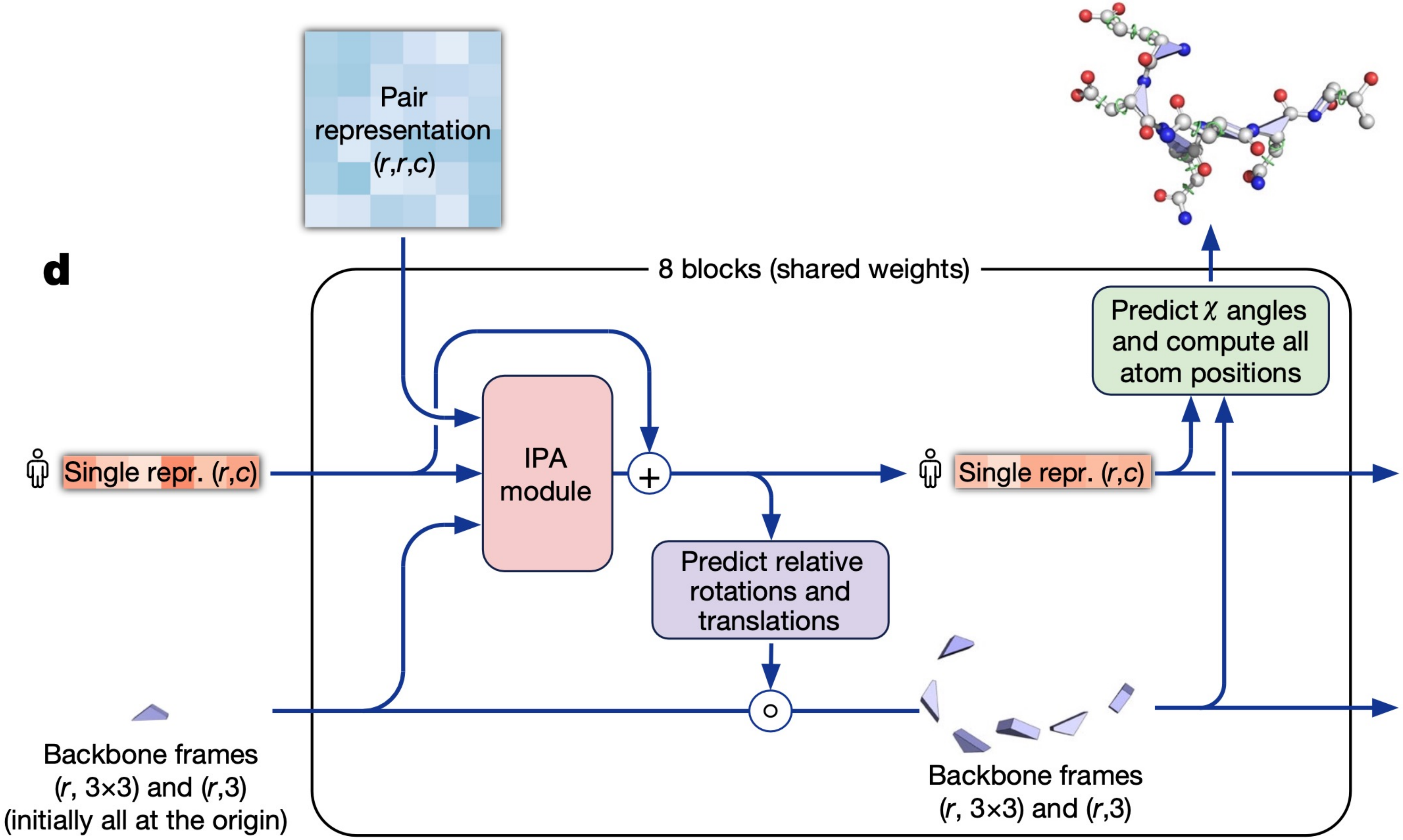
  $\{\mathbf{z}_{ij}\} \mathrel{+}= \text{DropoutColumnwise}_{0.25}(\text{TriangleAttentionEndingNode}(\{\mathbf{z}_{ij}\}))$

  $\{\mathbf{z}_{ij}\} \mathrel{+}= \text{PairTransition}(\{\mathbf{z}_{ij}\})$

end for

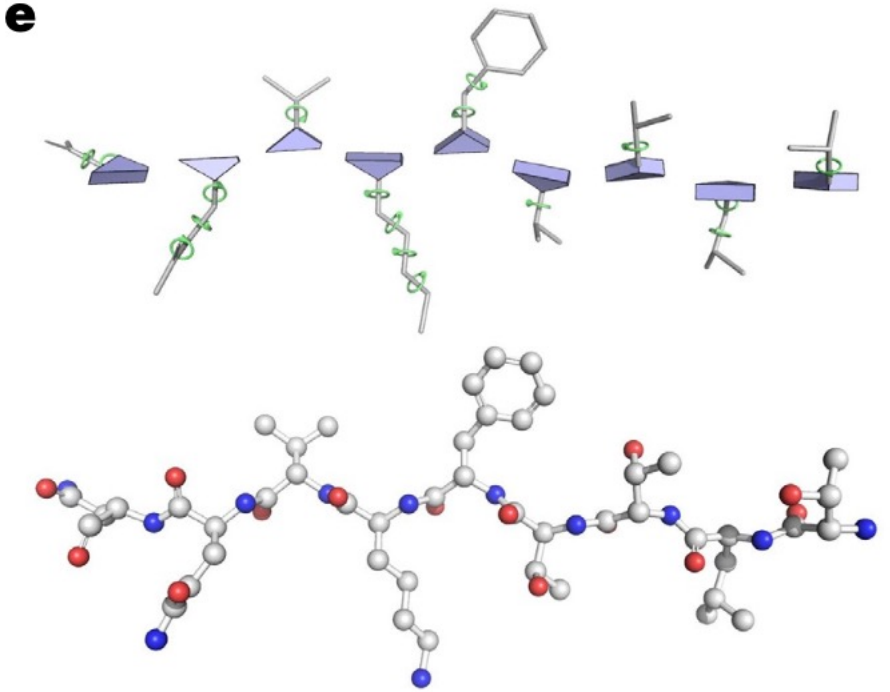
return $\{\mathbf{z}_{ij}\}$

Structure Module



Frame and torsion angle

e



- Backbone: $\{T_i = (R_i, t_i))\}$ map from local frame to global frame

- Side chain: the torsion angles are the only degrees of freedom, while all bond angles and bond lengths are fully rigid.

3D Equivariance

Algorithm 22 Invariant point attention (IPA)

def $\text{InvariantPointAttention}(\{\mathbf{s}_i\}, \{\mathbf{z}_{ij}\}, \{T_i\}, N_{\text{head}} = 12, c = 16, N_{\text{query points}} = 4, N_{\text{point values}} = 8):$

$\mathbf{q}_i^h, \mathbf{k}_i^h, \mathbf{v}_i^h = \text{LinearNoBias}(\mathbf{s}_i)$ $\qquad \mathbf{q}_i^h, \mathbf{k}_i^h, \mathbf{v}_i^h \in \mathbb{R}^c,\ h \in \{1, \ldots, N_{\text{head}}\}$

$\vec{\mathbf{q}}_i^{hp}, \vec{\mathbf{k}}_i^{hp} = \text{LinearNoBias}(\mathbf{s}_i)$ $\qquad \vec{\mathbf{q}}_i^{hp}, \vec{\mathbf{k}}_i^{hp}, \in \mathbb{R}^3,\ p \in \{1, \ldots, N_{\text{query points}}\},$ **units: nanometres**

$\vec{\mathbf{v}}_i^{hp} = \text{LinearNoBias}(\mathbf{s}_i)$ $\qquad \vec{\mathbf{v}}_i^{hp} \in \mathbb{R}^3,\ p \in \{1, \ldots, N_{\text{point values}}\},$ **units: nanometres**

$b_{ij}^h = \text{LinearNoBias}(\mathbf{z}_{ij})$

$w_C = \sqrt{\frac{2}{9 N_{\text{query points}}}},$

$w_L = \sqrt{\frac{1}{3}}$

$a_{ij}^h = \text{softmax}_j \left(w_L \left(\frac{1}{\sqrt{c}} \mathbf{q}_i^{h\top} \mathbf{k}_j^h + b_{ij}^h - \frac{\gamma^h w_C}{2} \sum_p \left\| T_i \circ \vec{\mathbf{q}}_i^{hp} - T_j \circ \vec{\mathbf{k}}_j^{hp} \right\|^2 \right) \right)$

$\tilde{\mathbf{o}}_i^h = \sum_j a_{ij}^h \mathbf{z}_{ij}$

$\mathbf{o}_i^h = \sum_j a_{ij}^h \mathbf{v}_j^h$

$\vec{\mathbf{o}}_i^{hp} = T_i^{-1} \circ \sum_j a_{ij}^h \left(T_j \circ \vec{\mathbf{v}}_j^{hp} \right)$

$\tilde{\mathbf{s}}_i = \text{Linear}\left(\text{concat}_{h,p}(\tilde{\mathbf{o}}_i^h, \mathbf{o}_i^h, \vec{\mathbf{o}}_i^{hp}, \left\| \vec{\mathbf{o}}_i^{hp} \right\|) \right)$

return $\{\tilde{\mathbf{s}}_i\}$

Algorithm 23 Backbone update

def $\text{BackboneUpdate}(\mathbf{s}_i)$:

$b_i, c_i, d_i, \vec{\mathbf{t}}_i = \text{Linear}(\mathbf{s}_i)$ $\qquad b_i, c_i, d_i \in \mathbb{R},\ \vec{\mathbf{t}}_i \in \mathbb{R}^3$

 # Convert (non-unit) quaternion to rotation matrix.

$(a_i, b_i, c_i, d_i) \leftarrow (1, b_i, c_i, d_i)/\sqrt{1 + b_i^2 + c_i^2, +d_i^2}$

$$R_i = \begin{pmatrix} a_i^2 + b_i^2 - c_i^2 - d_i^2 & 2b_ic_i - 2a_id_i & 2b_id_i + 2a_ic_i \\ 2b_ic_i + 2a_id_i & a_i^2 - b_i^2 + c_i^2 - d_i^2 & 2c_id_i - 2a_ib_i \\ 2b_id_i - 2a_ic_i & 2c_id_i + 2a_ib_i & a_i^2 - b_i^2 - c_i^2 + d_i^2 \end{pmatrix}$$

$T_i = (R_i, \vec{\mathbf{t}}_i)$

return T_i

R: orthogonal matrix, norm=1

Algorithm 24 Compute all atom coordinates

def computeAllAtomCoordinates$(T_i, \vec{\alpha}_i^f, F_i^{\text{aatype}})$:

$\hat{\vec{\alpha}}_i^f = \vec{\alpha}_i^f / \|\vec{\alpha}_i^f\|$

$(\vec{\omega}_i, \vec{\phi}_i, \vec{\psi}_i, \vec{\chi}_{1_i}, \vec{\chi}_{2_i}, \vec{\chi}_{3_i}, \vec{\chi}_{4_i}) = \hat{\vec{\alpha}}_i^f$

 # Make extra backbone frames.

$r_i = F_i^{\text{aatype}}$

$T_{i1} = T_i \circ T^{\text{lit}}_{r_i,(\omega \to \text{bb})} \circ \text{makeRotX}(\vec{\omega}_i)$

$T_{i2} = T_i \circ T^{\text{lit}}_{r_i,(\phi \to \text{bb})} \circ \text{makeRotX}(\vec{\phi}_i)$

$T_{i3} = T_i \circ T^{\text{lit}}_{r_i,(\psi \to \text{bb})} \circ \text{makeRotX}(\vec{\psi}_i)$

 # Make side chain frames (chain them up along the side chain).

$T_{i4} = T_i \circ T^{\text{lit}}_{r_i,(\chi_1 \to \text{bb})} \circ \text{makeRotX}(\vec{\chi}_{1_i})$

$T_{i5} = T_{i4} \circ T^{\text{lit}}_{r_i,(\chi_2 \to \chi_1)} \circ \text{makeRotX}(\vec{\chi}_{2_i})$

$T_{i6} = T_{i5} \circ T^{\text{lit}}_{r_i,(\chi_3 \to \chi_2)} \circ \text{makeRotX}(\vec{\chi}_{3_i})$

$T_{i7} = T_{i6} \circ T^{\text{lit}}_{r_i,(\chi_4 \to \chi_3)} \circ \text{makeRotX}(\vec{\chi}_{4_i})$

 # Map atom literature positions to the global frame.

$\vec{\mathbf{x}}_i^a = \text{concat}_{f,a'} \left(\{ T_i^f \circ \vec{\mathbf{x}}^{\text{lit}}_{r_i,f,a'} \} \right)$

return $T_i^f, \vec{\mathbf{x}}_i^a$

Loss

- FAPE: scores a set of predicted atom coordinates under a set of predicted local frames against the corresponding ground truth atom coordinates and ground truth local frames
- Auxiliary loss: FAPE + torsion Angle loss
- pLDDT(confidence loss): the per-residue lDDT-Cα scores.
- TM score: assessing global structure of predicted protein

Recycle

Algorithm 32 Embedding of Evoformer and Structure module outputs for recycling

def RecyclingEmbedder($\{\mathbf{m}_{1i}\}, \{\mathbf{z}_{ij}\}, \{\vec{\mathbf{x}}_i^{\mathrm{C}^\beta}\}$) :

 # Embed pair distances of backbone atoms:

1: $d_{ij} = \left\| \vec{\mathbf{x}}_i^{\mathrm{C}^\beta} - \vec{\mathbf{x}}_j^{\mathrm{C}^\beta} \right\|$ $\quad C^\alpha$ used for glycin

2: $\mathbf{d}_{ij} = \mathrm{Linear}(\mathrm{one\_hot}(d_{ij}, \mathbf{v}_{\mathrm{bins}} = [3\tfrac{3}{8}\,\text{Å},\ 5\tfrac{1}{8}\,\text{Å},\ \ldots, 21\tfrac{3}{8}\,\text{Å}]))$ $\quad \mathbf{d}_{ij} \in \mathbb{R}^{c_z}$

 # Embed output Evoformer representations:

3: $\tilde{\mathbf{z}}_{ij} = \mathbf{d}_{ij} + \mathrm{LayerNorm}(\mathbf{z}_{ij})$

4: $\tilde{\mathbf{m}}_{1i} = \mathrm{LayerNorm}(\mathbf{m}_{1i})$

5: **return** $\{\tilde{\mathbf{m}}_{1i}\}, \{\tilde{\mathbf{z}}_{ij}\}$

Inference time

-V100 GPU

-seq_len= 256   4.8min

-seq_len =384   9.2min

-seq_len =2500 18h

Ablation Study