MSA Scoring

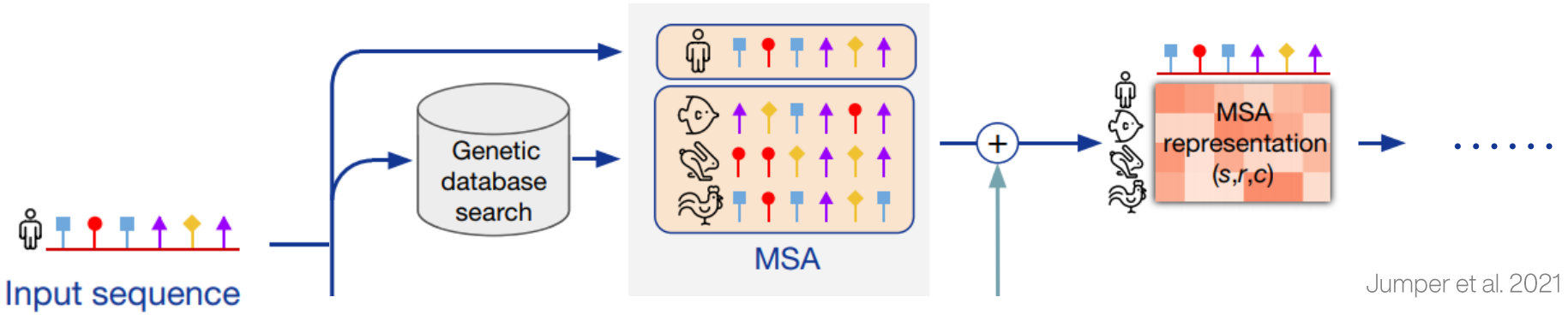
Based on MSA Transformer

Group lianyh, tengyue & yangxch

Background

AlphaFold & MSA Transformer

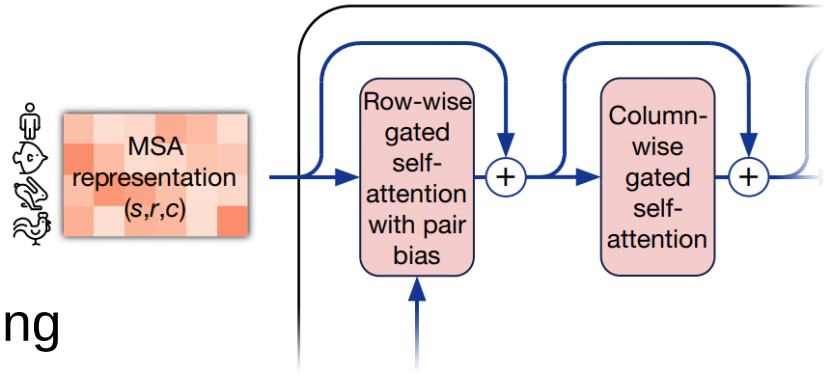
Background: AlphaFold



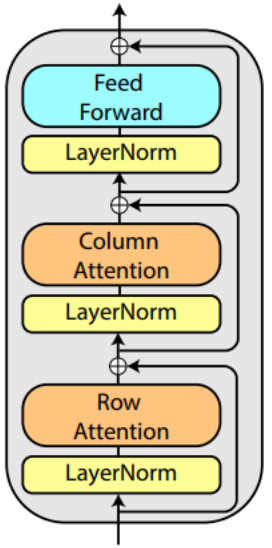
- Highly accurate in protein structure prediction
- Requires **Multiple-Sequence Alignment (MSA)** as a key input
- Various MSA acquisition approaches
  - Databases: `BFD, Uniclust30, Uniref90, MGnify, …`
  - Tools: `jackhammer, HHBlits, HHSearch, MMseqs264, …`
- **Motivation**: how to determine **whether a MSA input is of high quality** to help AlphaFold predict highly accurate structure?

Background: MSA Transformer

- Unsupervised protein language model
- **Masked language modeling objective**
  - Reconstruct masked tokens, similar to BERT pretraining
  - AlphaFold has similar **BERT-like loss** at training
- **Axial attention** over rows and columns
  - enables to extract information from dependencies in the input set and generalize patterns across MSAs
  - Also similar to part of AlphaFold **Evoformer** blocks, which exchange information within the MSA to enable direct reasoning about the spatial and evolutionary relationships



AlphaFold Evoformer block, Jumper et al. 2021 ↑
MSA Transformer block, Rao et al. 2021 ↓

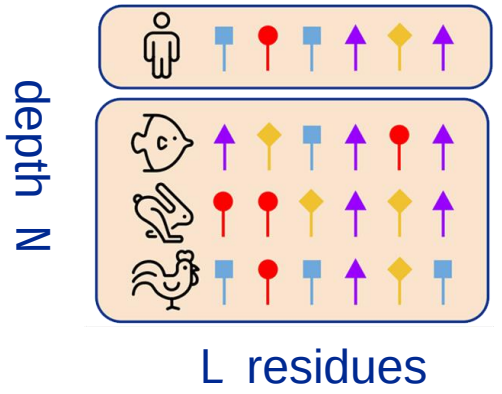


Methodology

Dataset & Networks & Loss Metrics

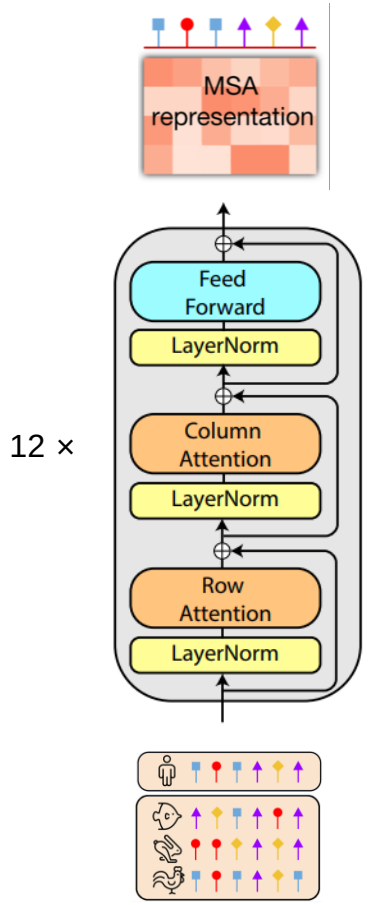
Methodology: Dataset

- 2850 MSAs from CASP14 Dataset
  - 2660 train MSAs: 95 query sequences and 28 MSAs for each query
  - 190 test MSAs: 1 pair of MSAs for each query sequence
- MSA subsampling strategy
  - HH-Filter: >256 to ≈256 sequences
    - `hhfilter –i input.a3m -o filtered.a3m -diff 256`
  - Diversity Maximizing: ≈256 to 256 sequences
    - greedily pick sequence with maximum average hamming distance
  - Result shape: N × L
    - N ≤ 256 and L ≤ 584



Methodology: MSA Embedding

- Use **MSA Transformer** as encoder
  - Import `esm.pretrained.esm_msa1b_t12_100M_UR50S()`
  - Freeze weights of all layers for transfer learning
  - Save extracted embedding of each MSA for efficient training
- Result MSA embedding shape: `N × L × D`
  - MSA Transformer embedding dimension → D=768
  - Add zero paddings → L=584
  - Extract query reference → **`L × D` feature map** each MSA



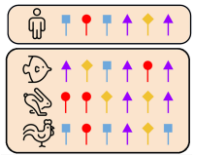
Methodology: Score Regression Network

- Use **Multi-Layer-Perceptron(MLP)** for vector regression
  - Averaging pooling/extract `<bos>` embedding → `768`-dim vector
  - Feed into `3` fully-connected layers with `leaky_relu` activations
  - Output one score and compute `MSELoss`

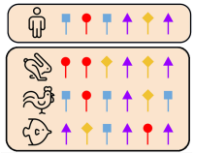
- Use **Convolutional Neural Network(CNN)** for feature map scoring
  - `L × D` feature map → convolution layers
  - Add residual blocks and pooling layers
  - Extract query reference → `L × D` feature map each MSA

Methodology: Siamese Framework

Pointwise MSA Dataset



$\text{MSA}_1 \quad s_1$

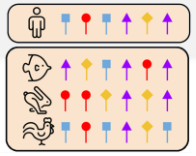
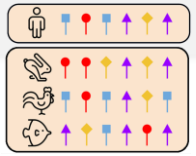
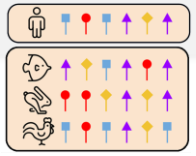
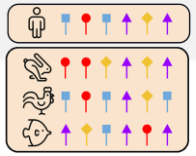


$\text{MSA}_2 \quad s_2$

↓

Group MSAs by query sequence

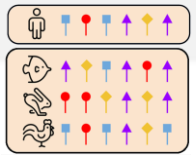
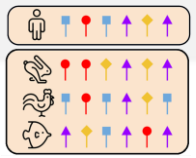
query i

↓

Paired MSA Dataset

query i

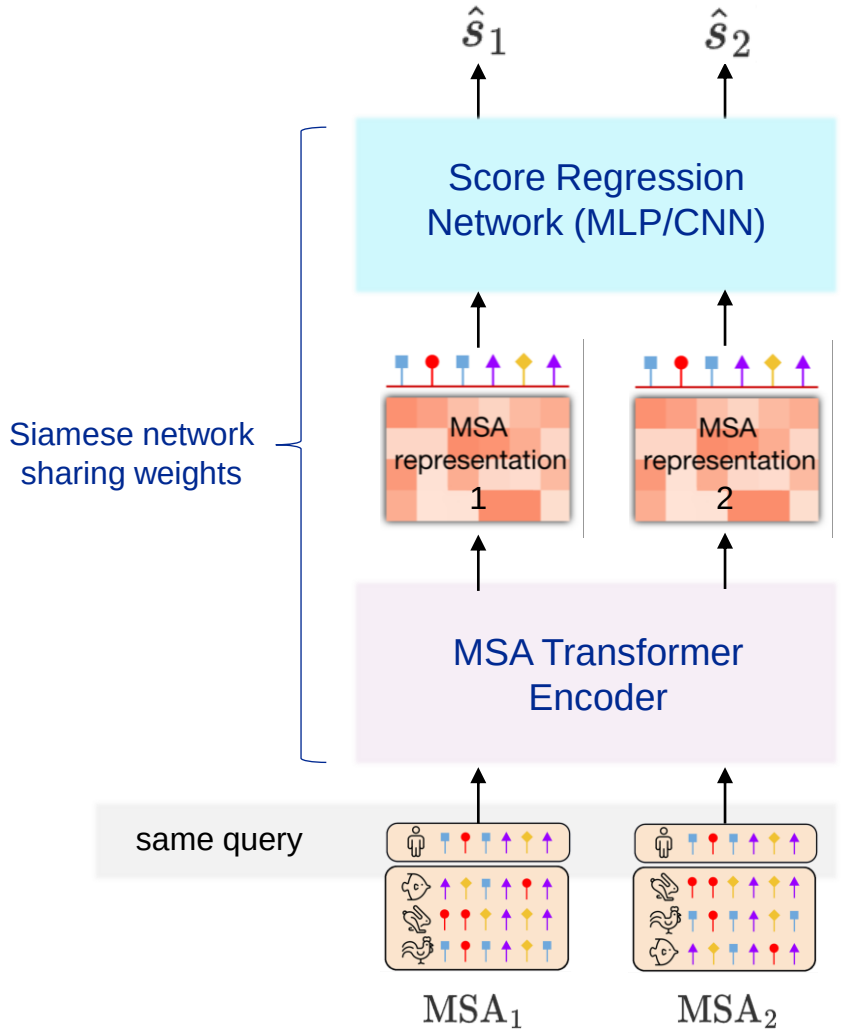



$\text{MSA}_1 \quad \text{MSA}_2$

$s_1 \quad s_2$

$$y_{12} = \begin{cases} 1 & s_1 > s_2 \\ 0 & s_1 < s_2 \end{cases}$$

Methodology: Siamese Framework



$\hat{P}_{12}$: possibility that MSA_1 is of higher quality than MSA_2

$$\hat{P}_{12} = \mathrm{sigmoid}(\hat{s}_1 - \hat{s}_2) = \frac{1}{1+e^{-(\hat{s}_1-\hat{s}_2)}}$$ [1]

Pairwise Binary Cross Entropy loss:

$$\ell_{\mathrm{BCE}} = -[y_{12} \cdot \log \hat{P}_{12} + (1 - y_{12}) \log(1 - \hat{P}_{12})]$$

$$y_{12} = \begin{cases} 1 & s_1 > s_2 \\ 0 & s_1 < s_2 \end{cases}$$

Mean Squared Error loss:

$$\ell_{\mathrm{MSE}} = \frac{1}{2}[(\hat{s}_1 - s_1)^2 + (\hat{s}_2 - s_2)^2]$$

Total loss for back propagation:

$$\ell = \ell_{\mathrm{BCE}} + \lambda \ell_{\mathrm{MSE}}$$

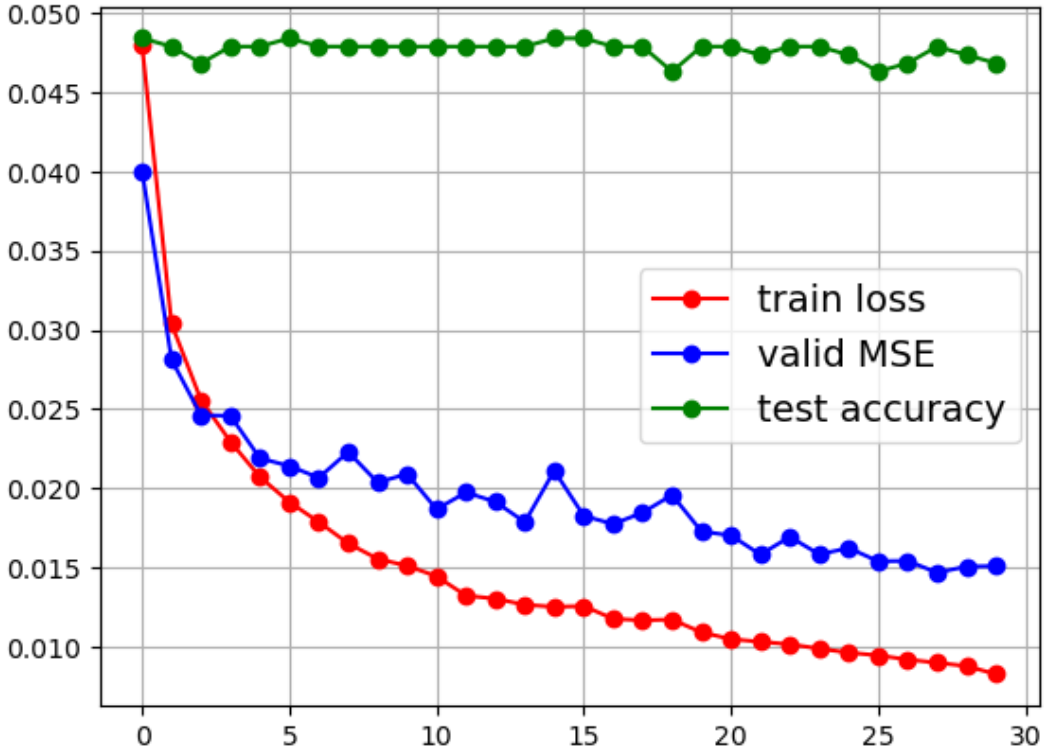
[1] Burges C . From RankNet to LambdaRank to LambdaMART: An Overview[J]. learning, 2010.

Results & Conclusion

MLP vs. CNN & Pointwise vs. Pairwise & Metric Reports

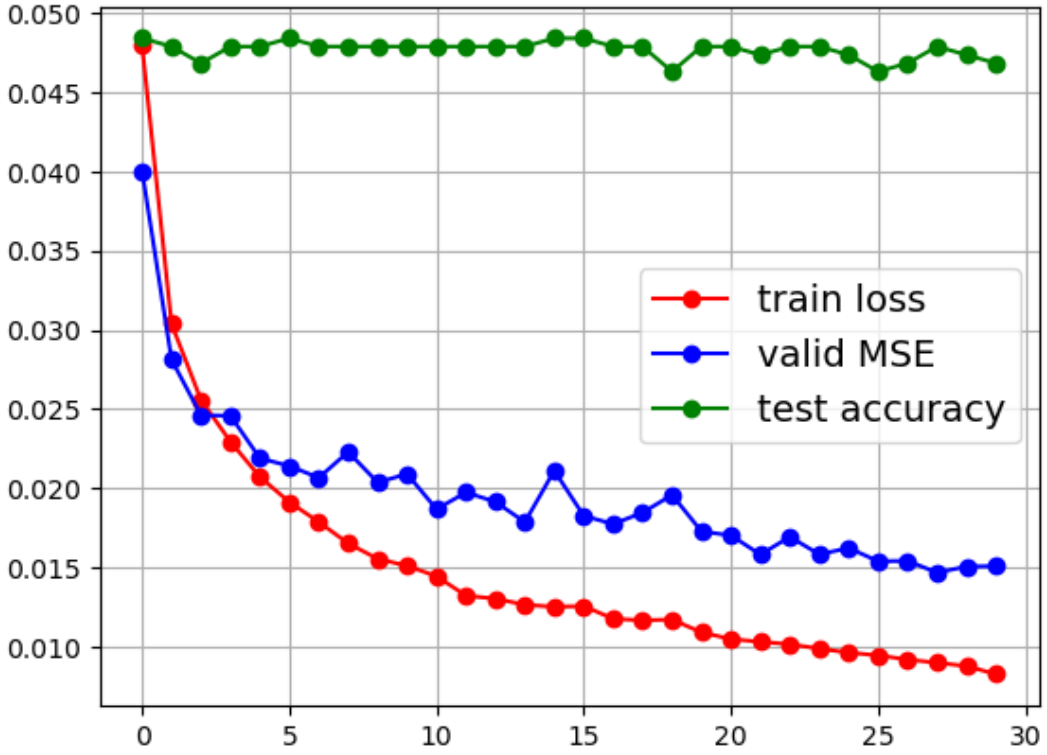
Results: Pointwise Scoring Network

- MLP: `0.00826` train MSE after `20` epochs
- CNN: `0.00826` train MSE after `20` epochs
- Naïve baseline for comparison:
  - output MSA with more sequences
  - baseline test acc: `0.7789`
  - a great boost!



Results: Siamese Network + Pairwise Loss

- MLP: `0.00826` train MSE after `20` epochs
- CNN: `0.00826` train MSE after `20` epochs
- Hyperparameter settings:
  - output MSA with more sequences
  - baseline test acc: `0.7789`
  - a great boost!



Thanks for your Attention!

MSA Scoring Based on MSA Transformer

Group lianyh, tengyue & yangxch