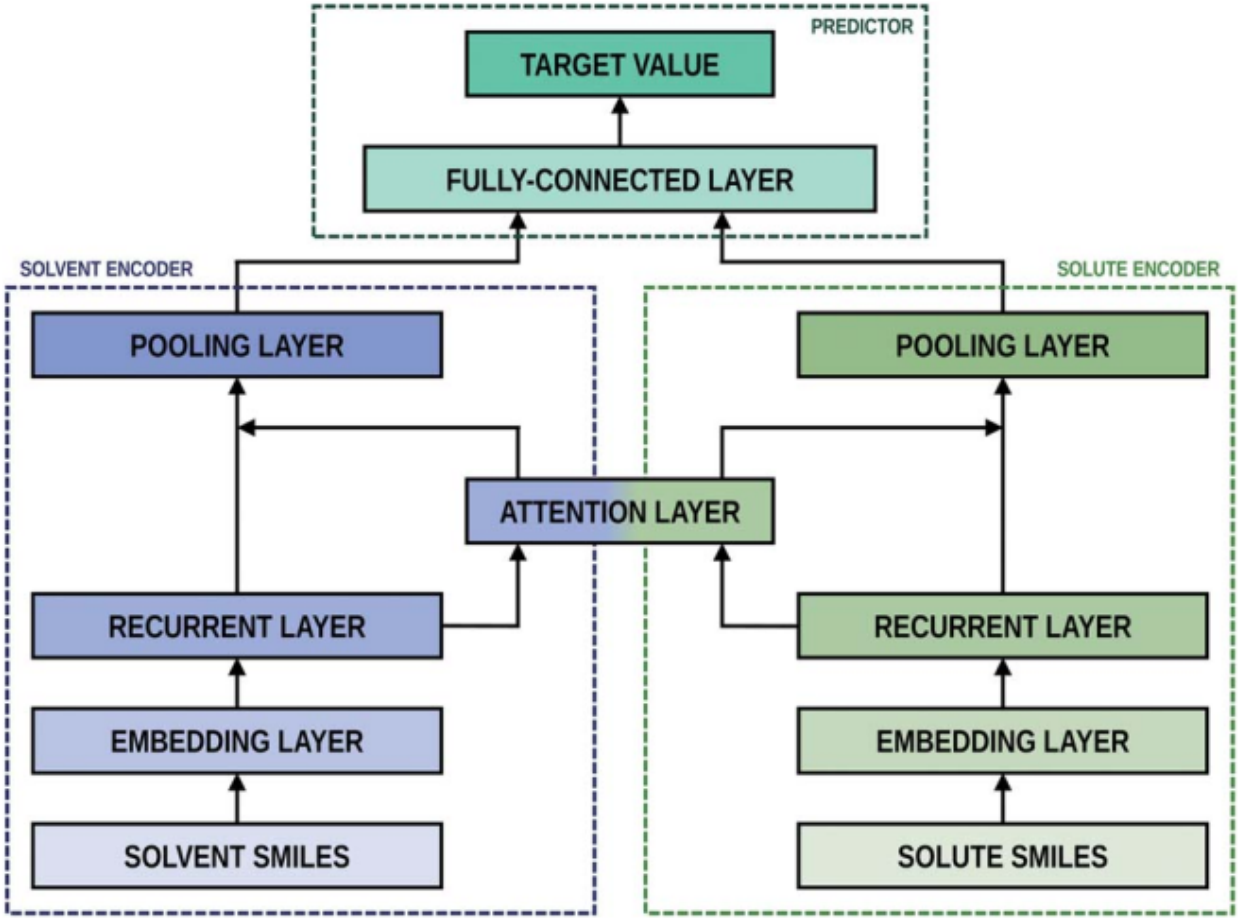
# Deep learning model for prediction of solvation free energies in generic organic solvents

# Architecture



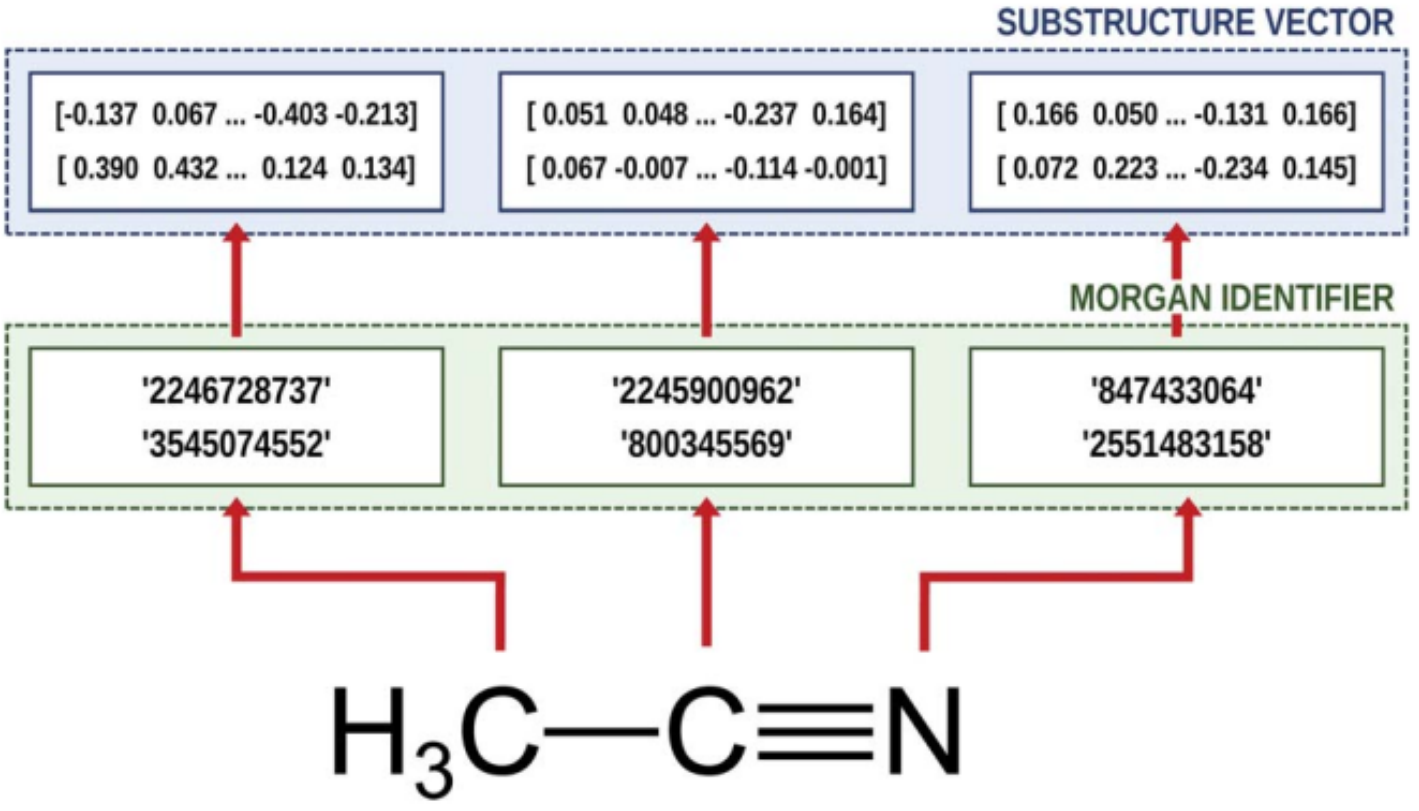
# Solute Smiles

- Minnesota solvation database (MNSOL)

# Embedding layer

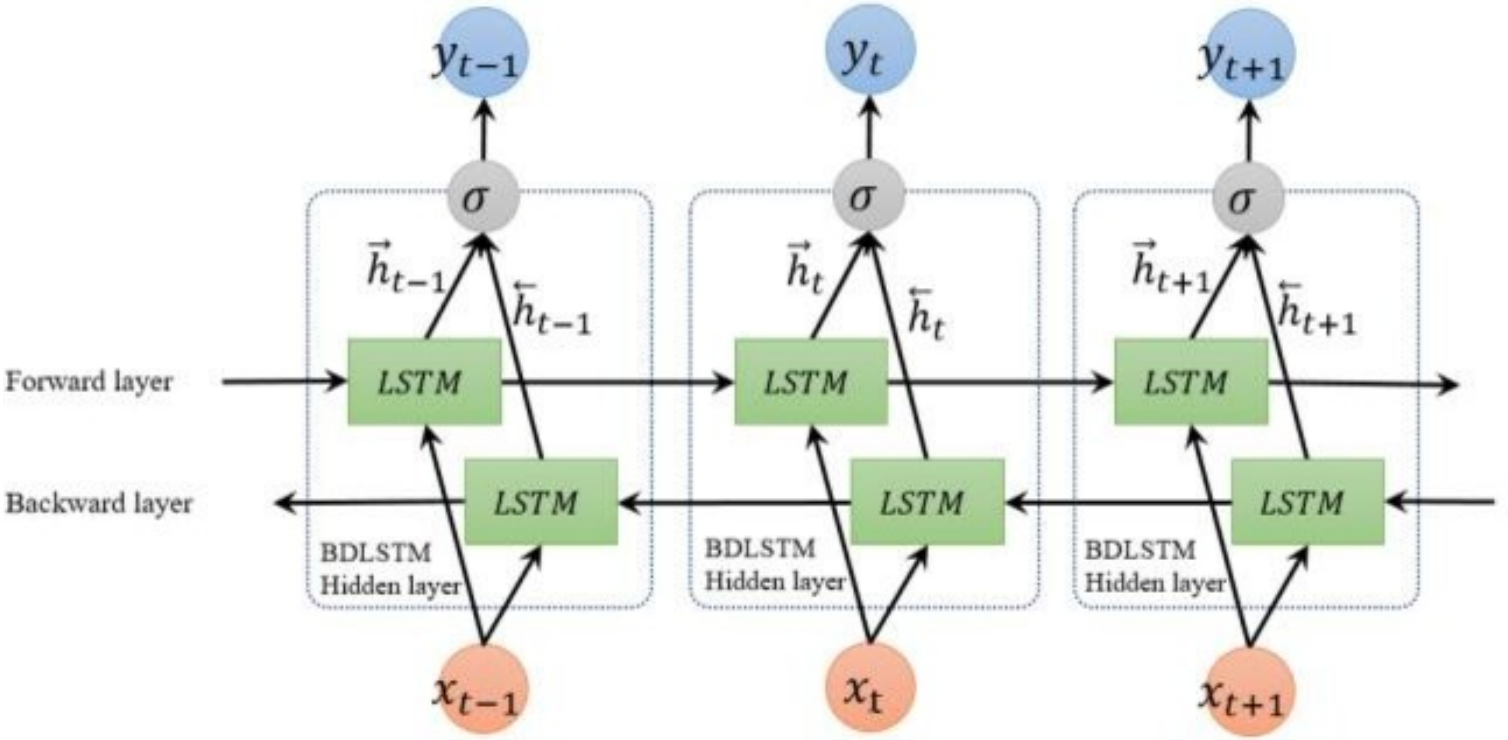
**Mol2vec**

**Morgan Algorithm**



# Recurrent layer

- Bidirectional LSTM



# Attention layer

Normalization with the softmax function

$$\alpha_{ij} = \frac{\exp\left(\text{score}\left(\mathbf{h}_i, \mathbf{g}_j\right)\right)}{\sum_k \exp(\text{score}(\mathbf{h}_i, \mathbf{g}_k))}$$

The solvent context, P denotes an emphasized hidden state H with the attention alignment. Solute context Q is obtained using the same procedure.

$$\mathbf{p}_i = \sum_j^M \alpha_{ij} \mathbf{g}_j$$

Luong's dot-product attention as a score function since it is computationally efficient.

$$\text{score}(\mathbf{h}_i, \mathbf{g}_j) = \mathbf{h}_i \cdot \mathbf{g}_j$$

# Pooling layer

The context weighted from the attention layer is an Lx2D matrix. Two max-pooling layers reduce contexts H, G, P, and Q to feature vectors u and v.

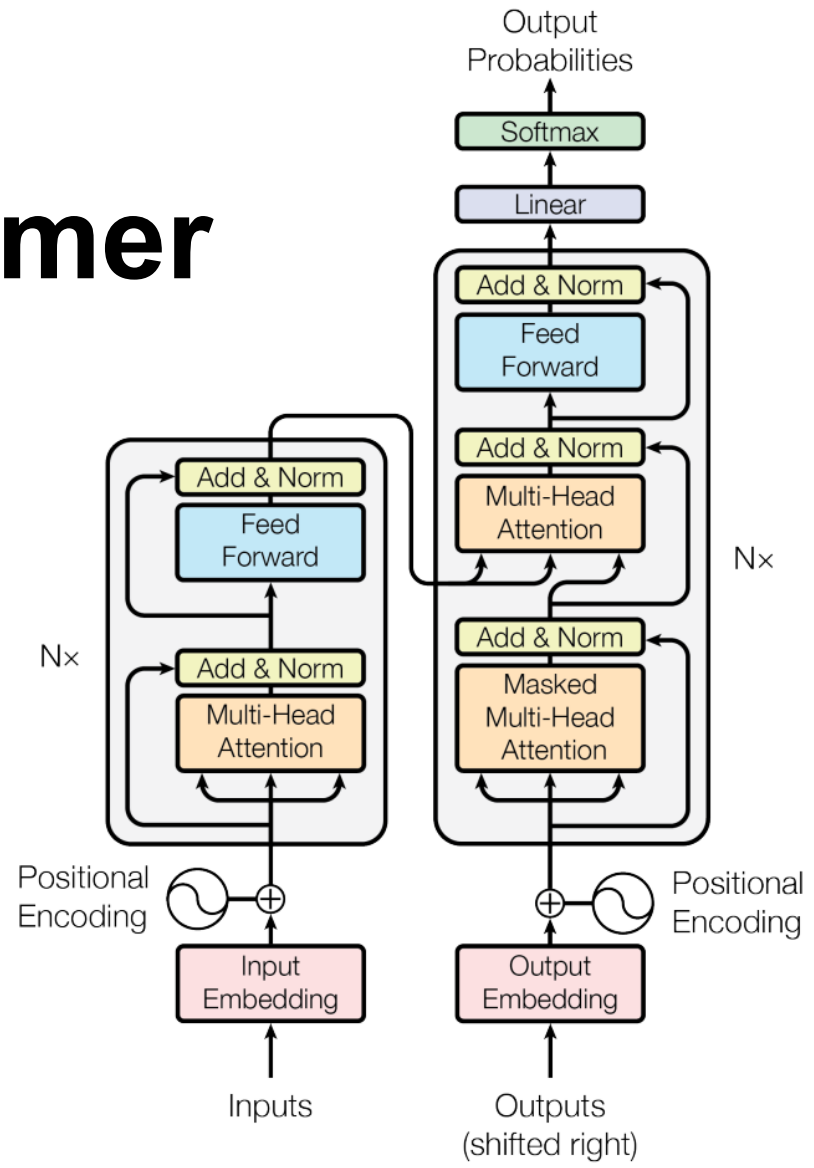
$$\mathbf{u} = \text{MaxPooling}([\mathbf{h}_1;\mathbf{p}_1,\ldots,\mathbf{h}_N;\mathbf{p}_N])$$

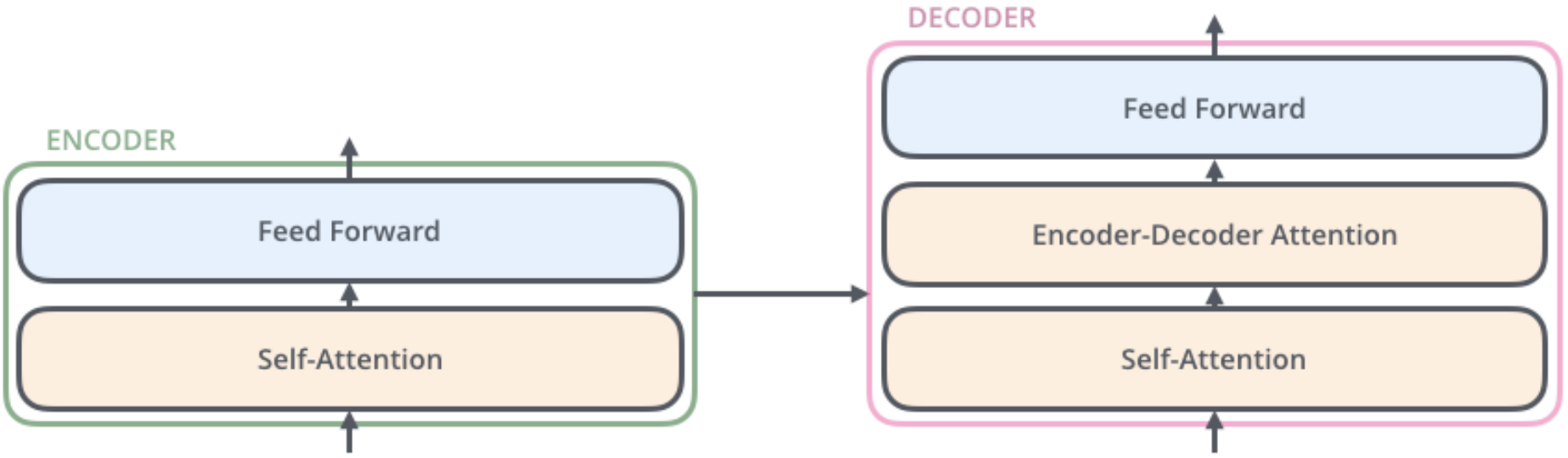
$$\mathbf{v} = \text{MaxPooling}([\mathbf{g}_1;\mathbf{q}_1,\ldots,\mathbf{g}_M;\mathbf{q}_M])$$

# **Predictor:** MLP

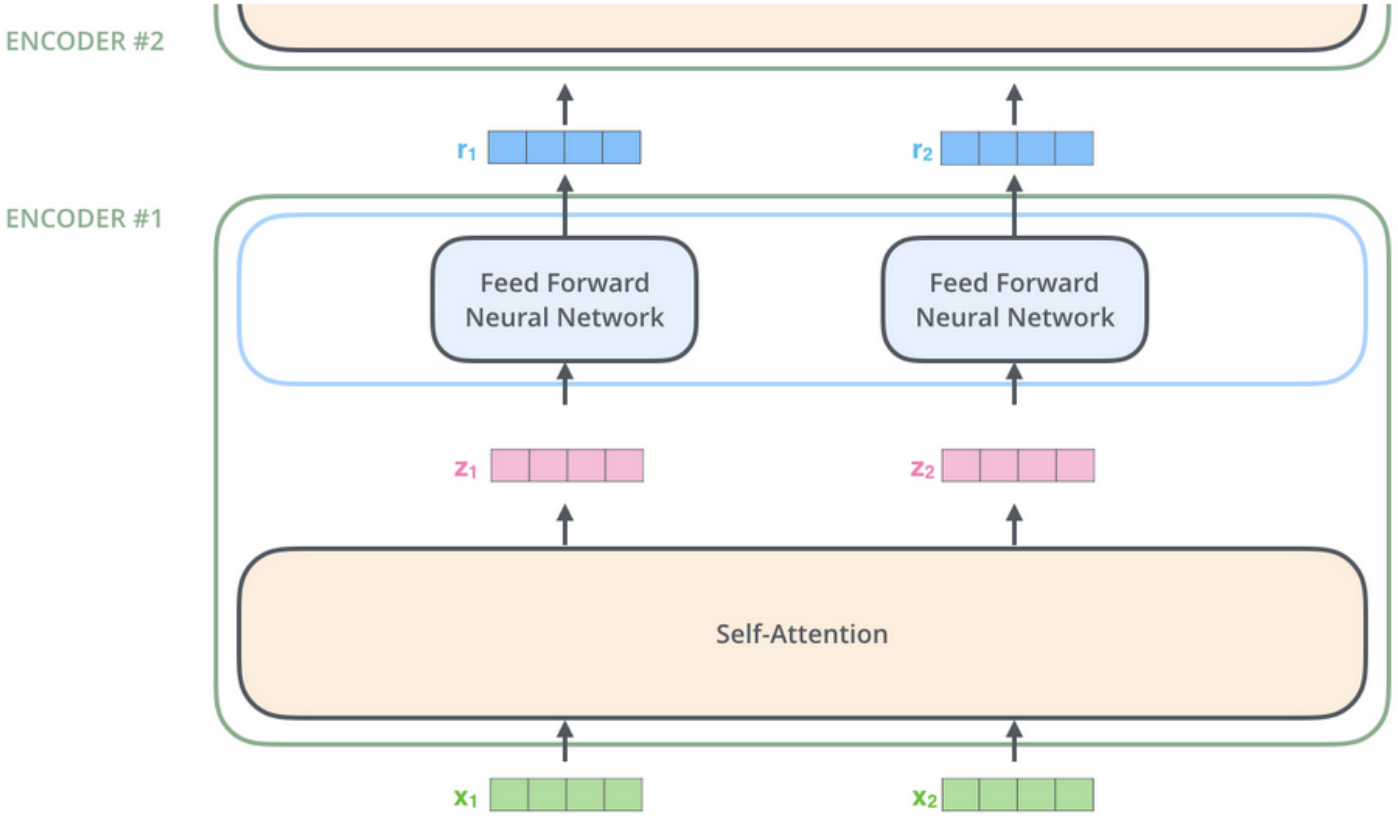
- 10-fold cross-validation (CV)
- Stochastic gradient descent (SGD) algorithm with Nesterov momentum:
  - learning rate is 0.0002
  - momentum is 0.9
- Loss function: root mean squared error (RMSE)
- **Result: RMSE 0.8**

# Transformer

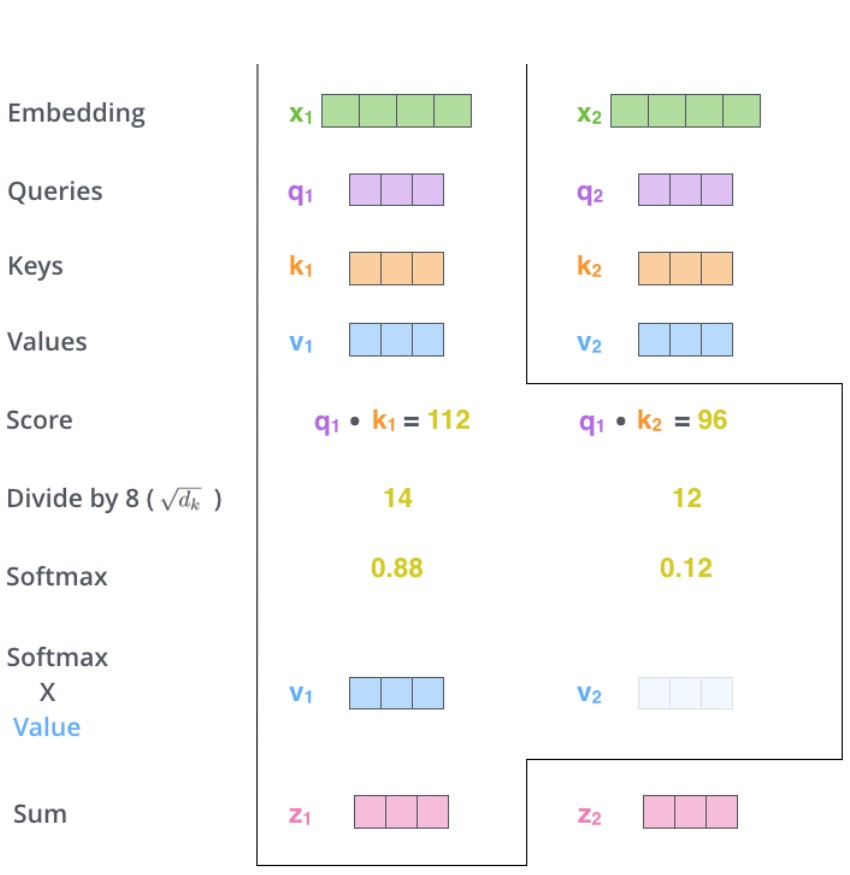


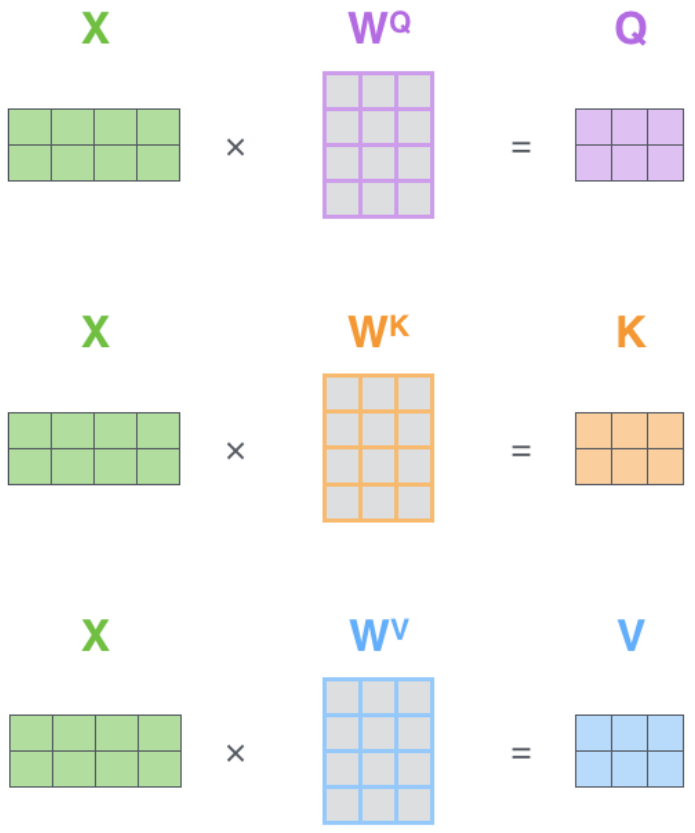

ENCODER
Feed Forward
Self-Attention
DECODER
Feed Forward
Encoder-Decoder Attention
Self-Attention

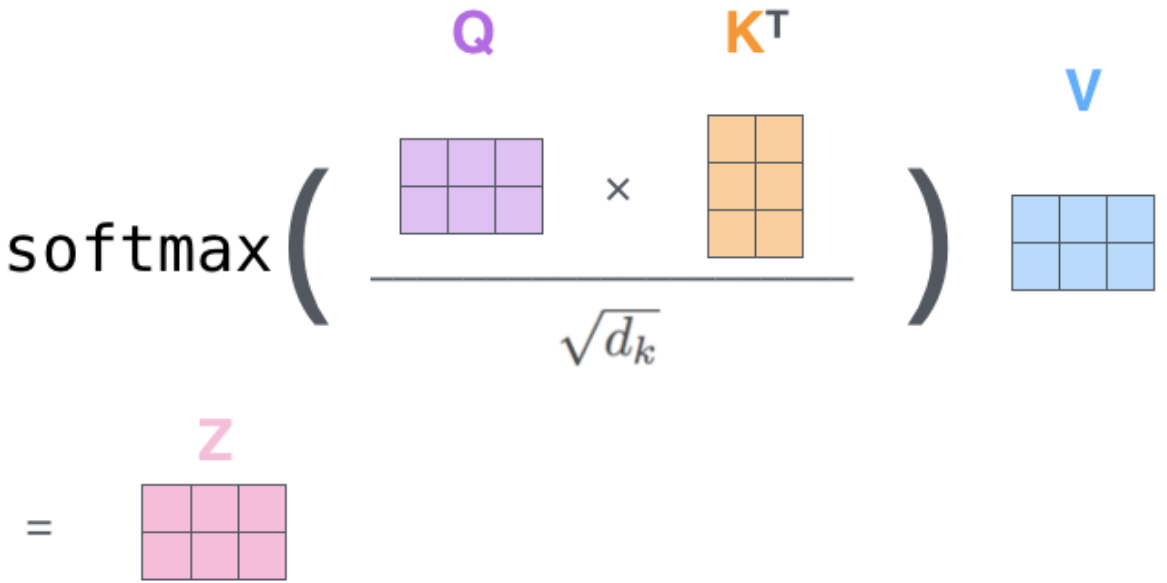
# Encoder layers



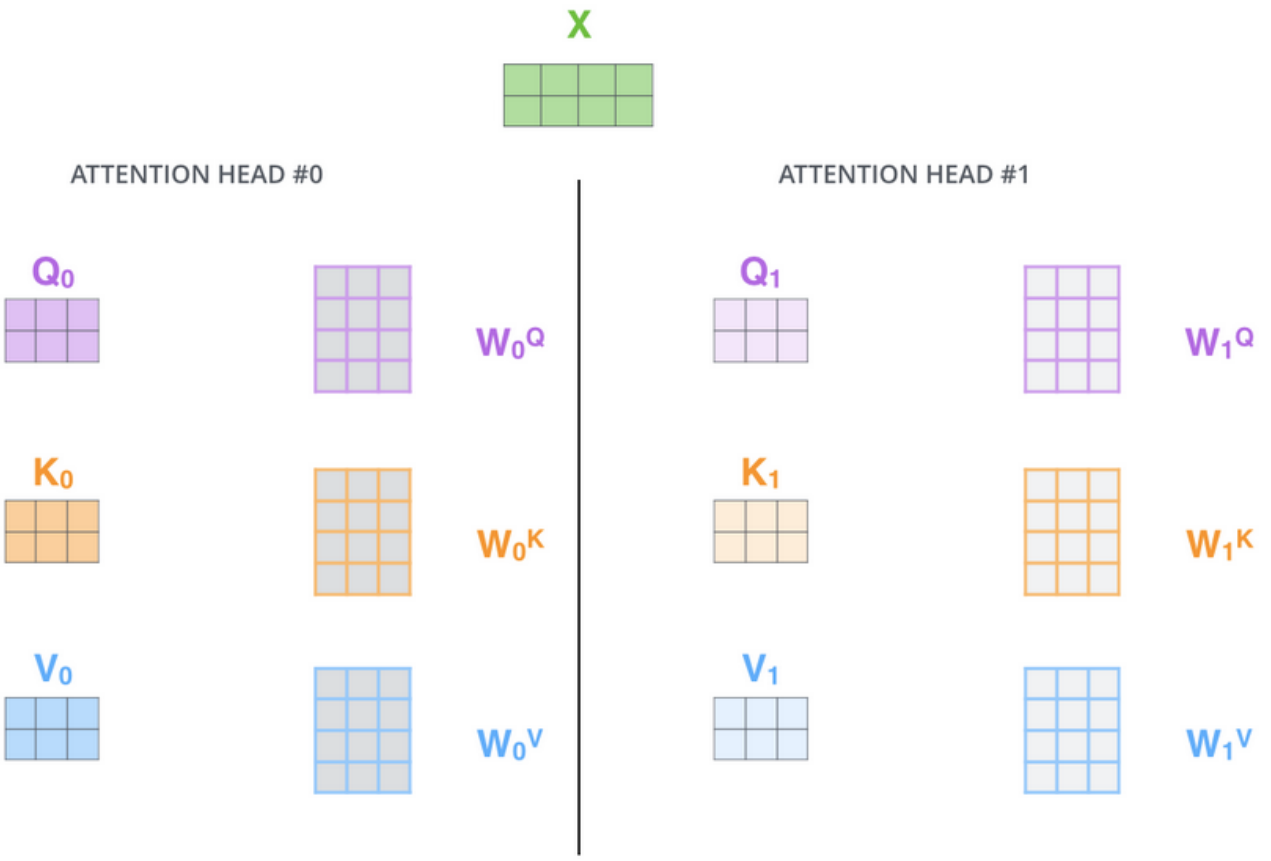
# Self attention




Q
K^T
V
softmax
×
$\sqrt{d_k}$
Z
=

# Multi-head attention



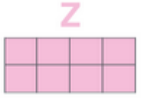
1) Concatenate all the attention heads


2) Multiply with a weight matrix $W^O$ that was trained jointly with the model

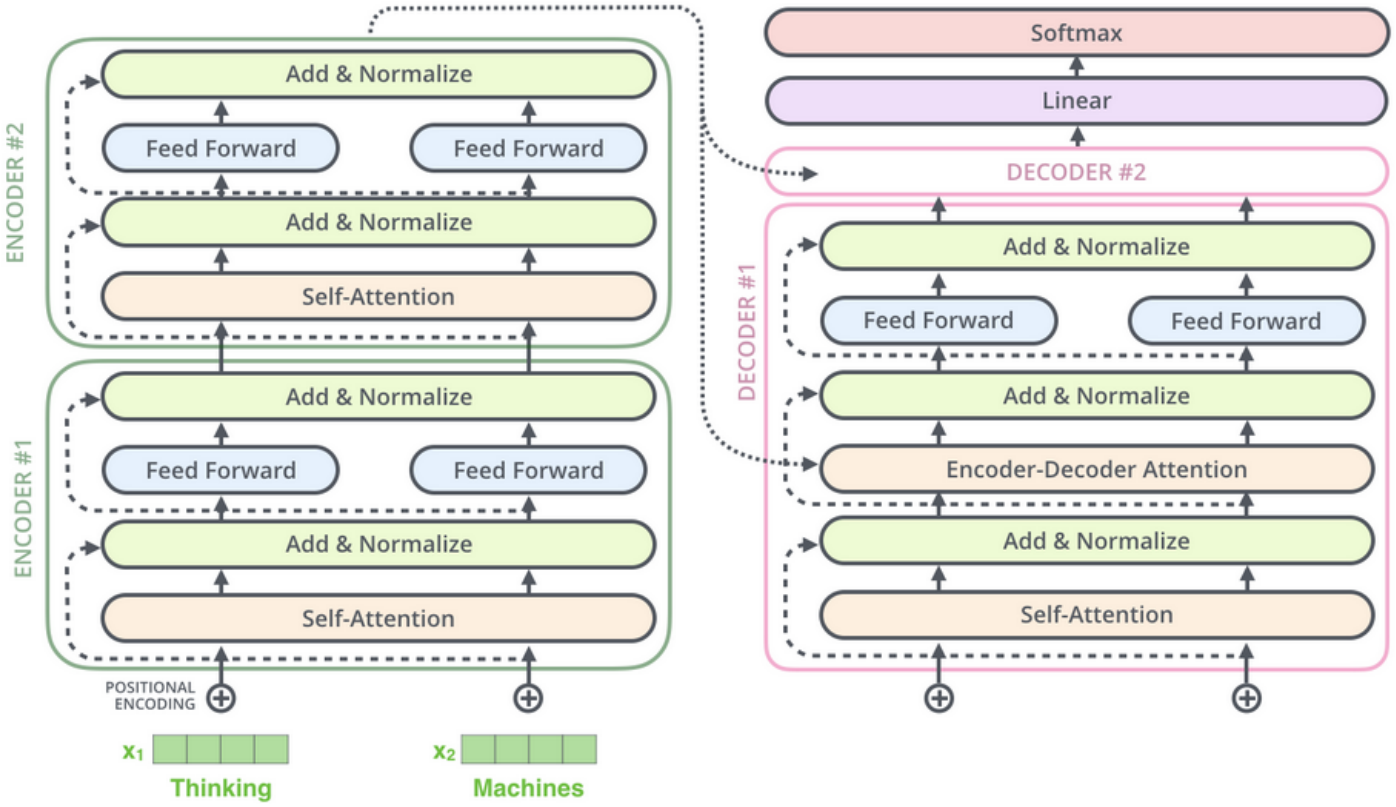
X

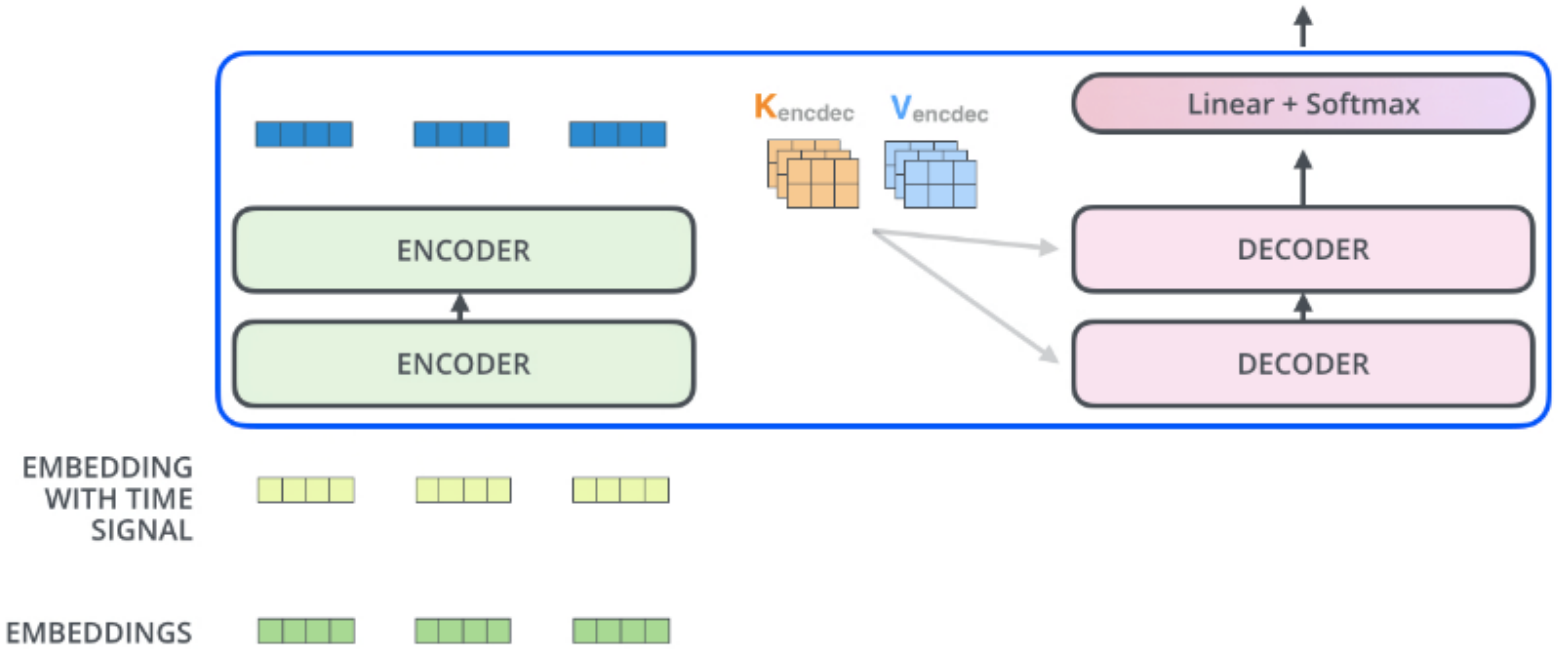

3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

=



# Architecture




Linear + Softmax
$K_{encdec}$
$V_{encdec}$
ENCODER
ENCODER
DECODER
DECODER
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS

# 2 encoder layers

- 10-fold cross-validation (CV)
- Stochastic gradient descent (SGD) algorithm with Nesterov momentum:
  - learning rate is 0.000002
  - momentum is 0.009
- Loss function: root mean squared error (RMSE)
- **Result: RMSE 1.08**

# 1 encoder layer + shared attention

- 10-fold cross-validation (CV)
- Stochastic gradient descent (SGD) algorithm with Nesterov momentum:
  - learning rate is 0.000002
  - momentum is 0.009
- Loss function: root mean squared error (RMSE)
- **Result: RMSE 0.85**

# 1 encoder layer + multihead

- 10-fold cross-validation (CV)
- Stochastic gradient descent (SGD) algorithm with Nesterov momentum:
  - learning rate is 0.000002
  - momentum is 0.009
- Loss function: root mean squared error (RMSE)
- **Result: Test RMSE 2, 2, 1.5..**