Temporal Module

周浩

2018/11/23

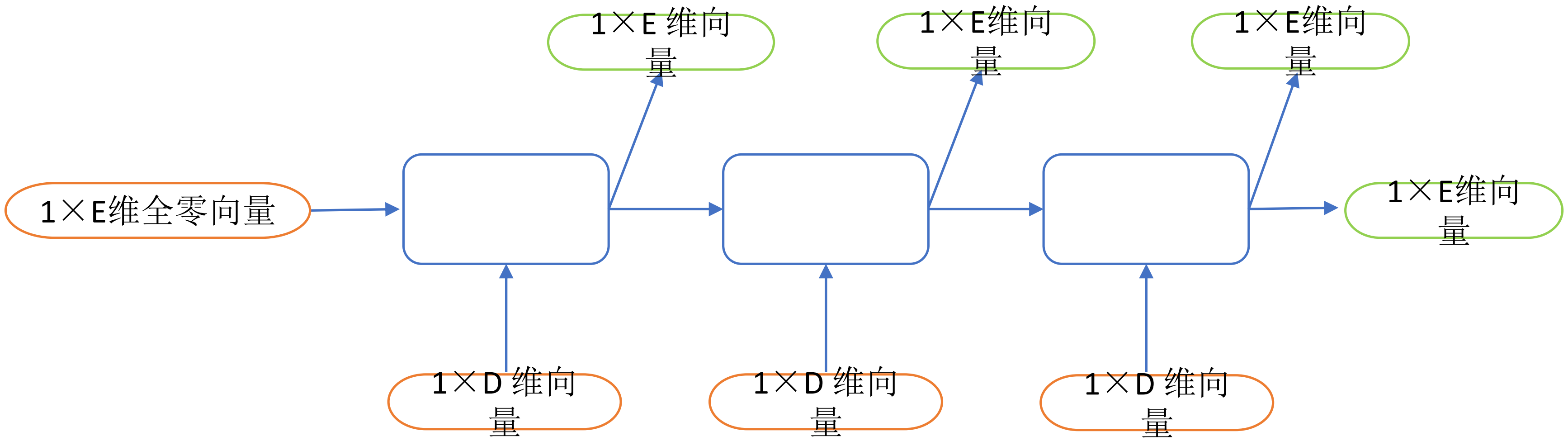
Outline:

- RNN
  - GRU

- Neural Machine Translation
  - Transformer

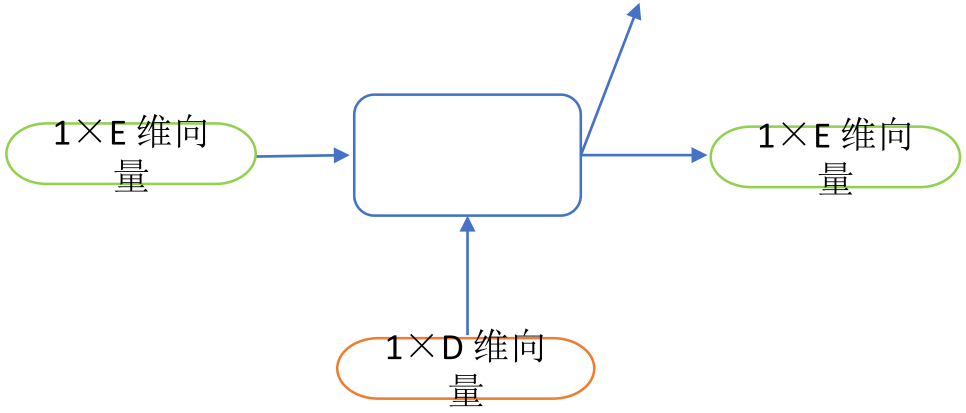
RNN

- 输入：　T个D维有序向量
- 输出：　T个E维有序向量
- 此处T=3



RNN Cell

- 输入： D维向量
- 输出： E维向量



class `torch.nn.RNNCell`*(input_size, hidden_size, bias=True, nonlinearity='tanh')* [source]

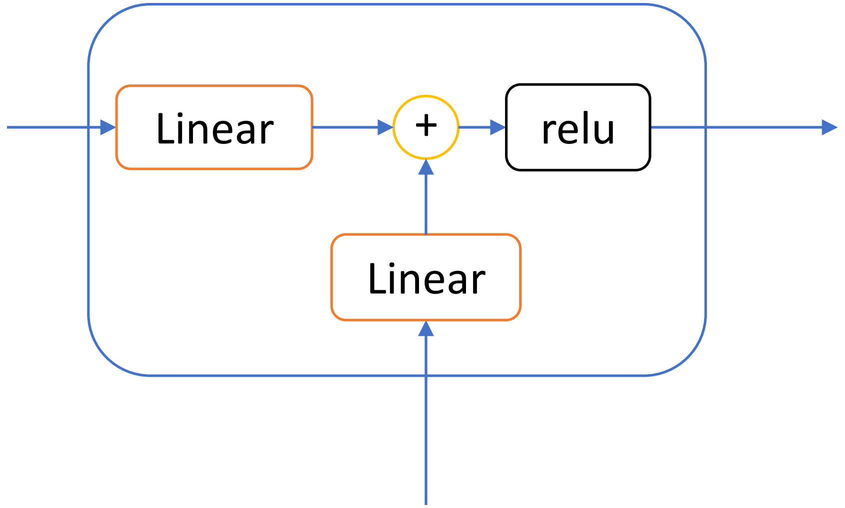
An Elman RNN cell with tanh or ReLU non-linearity.

$$h' = \tanh(w_{ih}x + b_{ih} + w_{hh}h + b_{hh})$$

If `nonlinearity` is *'relu'*, then ReLU is used in place of tanh.

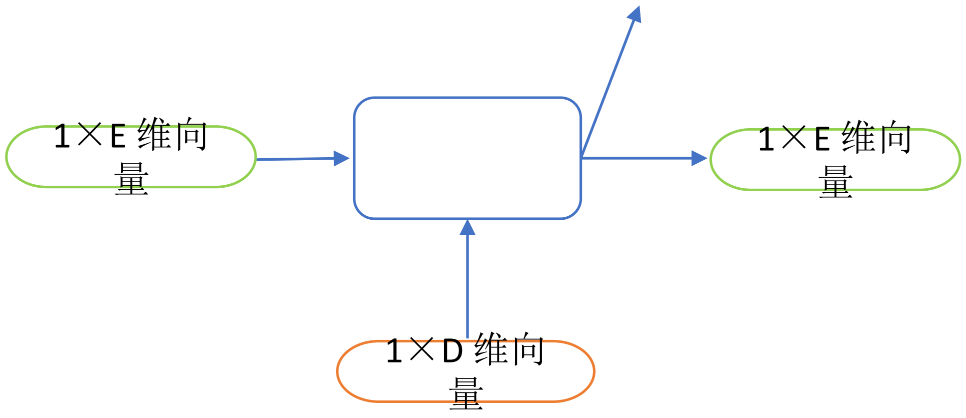
Parameters:

- **input_size** – The number of expected features in the input *x*
- **hidden_size** – The number of features in the hidden state *h*
- **bias** – If `False`, then the layer does not use bias weights *b_ih* and *b_hh*. Default: `True`
- **nonlinearity** – The non-linearity to use. Can be either 'tanh' or 'relu'. Default: 'tanh'



GRU Cell

- 输入： D维向量
- 输出： E维向量



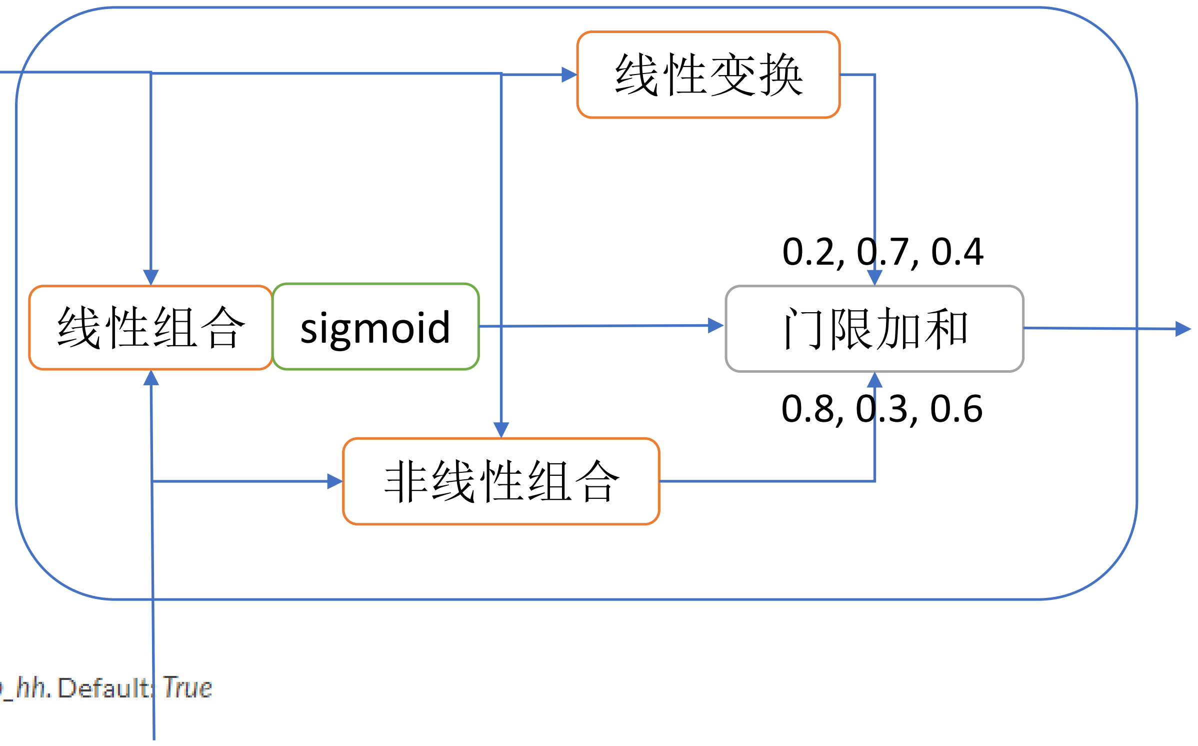
class `torch.nn.GRUCell`(*input_size, hidden_size, bias=True*) [source]

A gated recurrent unit (GRU) cell

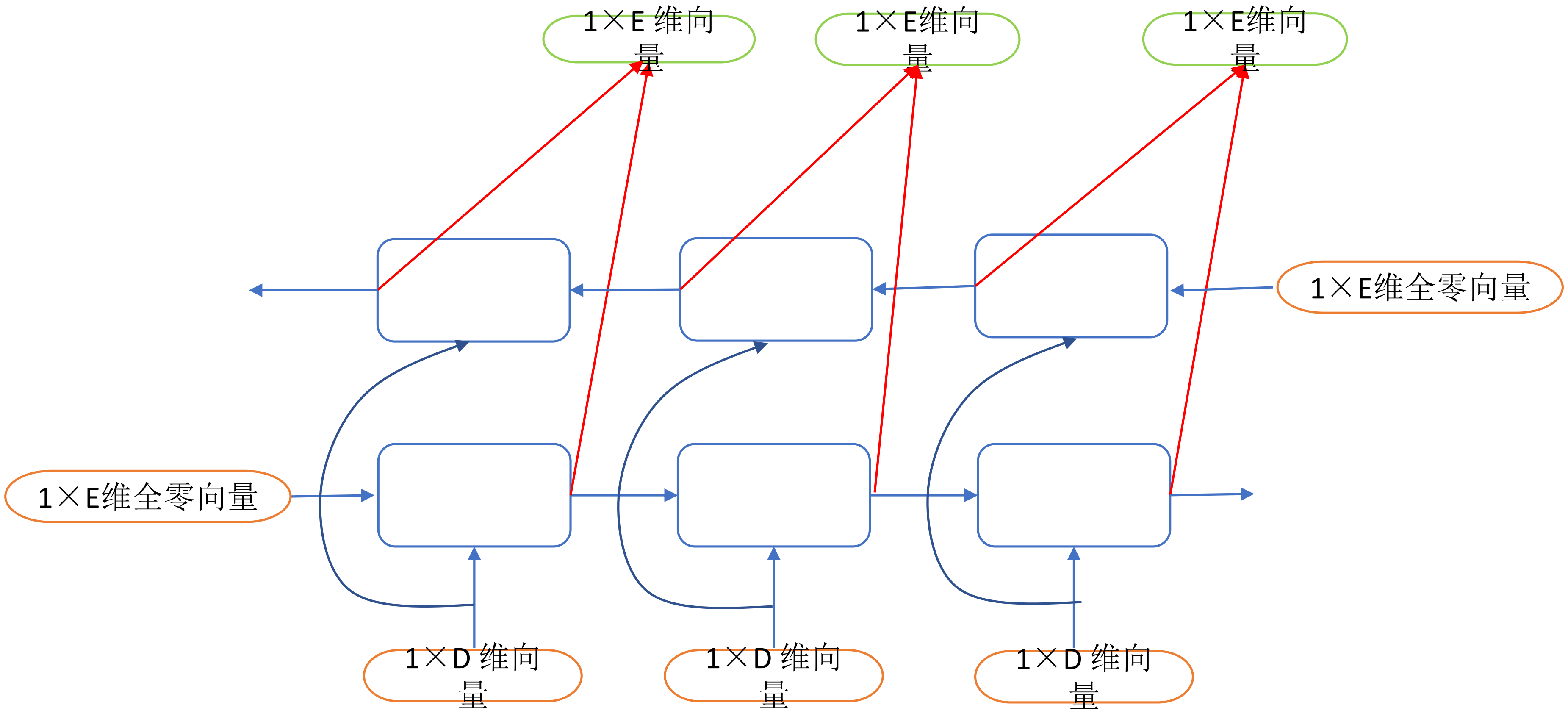
$$
\begin{aligned}
r &= \sigma(W_{ir} x + b_{ir} + W_{hr} h + b_{hr}) \\
z &= \sigma(W_{iz} x + b_{iz} + W_{hz} h + b_{hz}) \\
n &= \tanh(W_{in} x + b_{in} + r * (W_{hn} h + b_{hn})) \\
h' &= (1 - z) * n + z * h
\end{aligned}
$$

where σ is the sigmoid function.

Parameters:
- **input_size** – The number of expected features in the input *x*
- **hidden_size** – The number of features in the hidden state *h*
- **bias** – If *False*, then the layer does not use bias weights *b_ih* and *b_hh*. Default: *True*



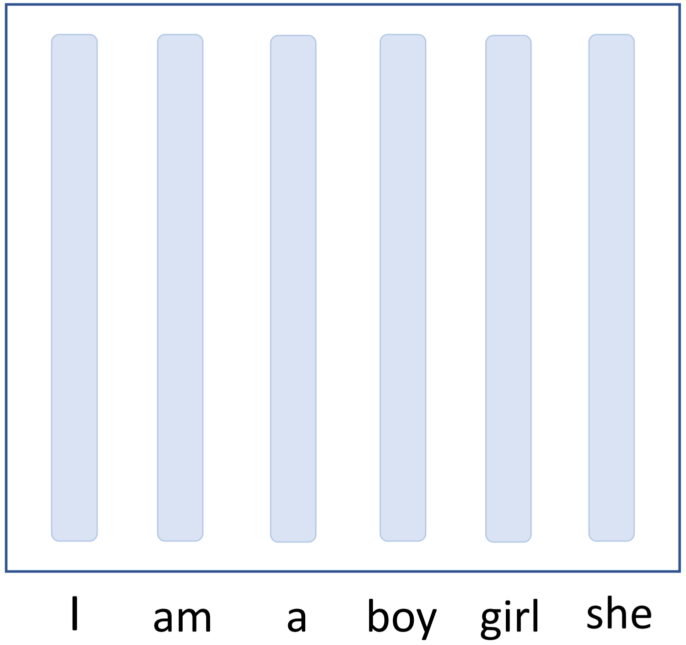
bidirectional RNN



1. Word Embedding and Prediction

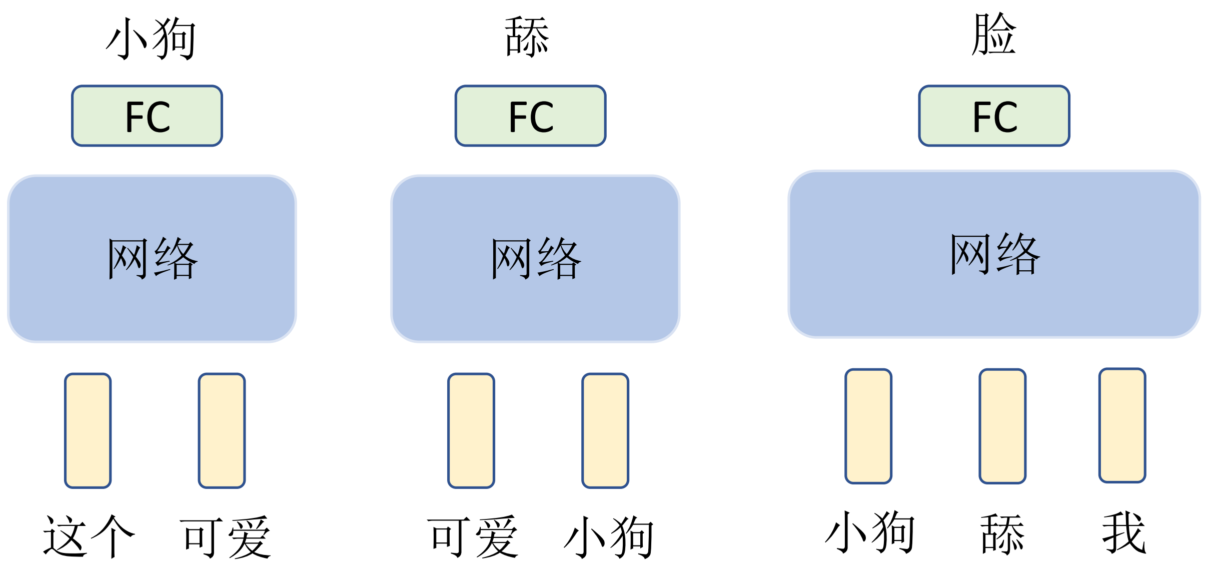
- embedding： W×D， 随机初始化，可训练的
  - 相似场景：个性化推荐
    - 给定对象和广告，学习特征的表达

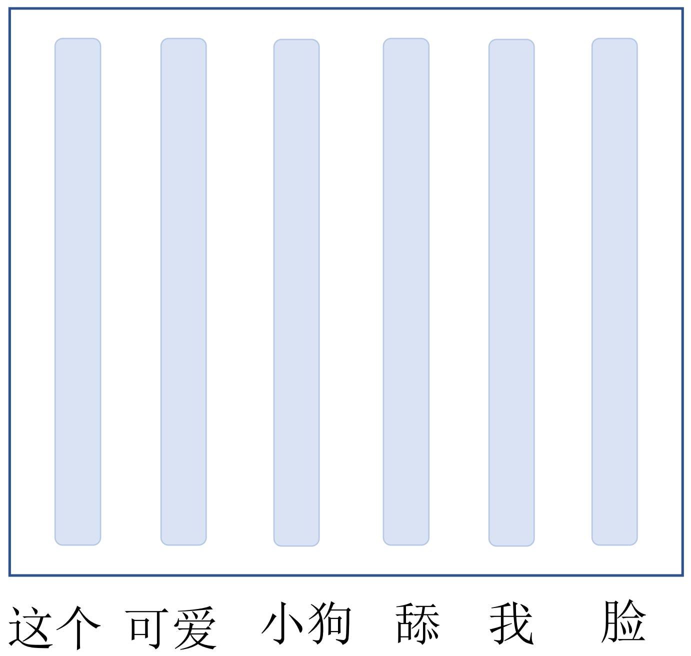
- prediction： D×W
  - 全连接层做分类，类别数为词的个数



Word Embedding and Prediction

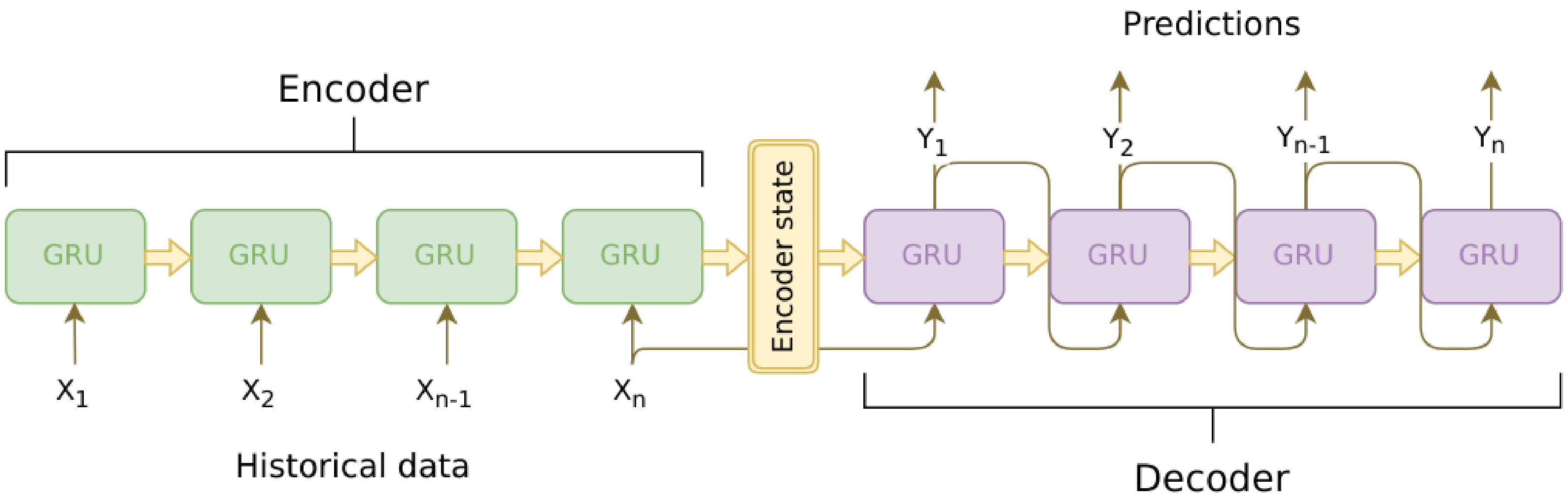
- 上下文预测：
- 这个 \ 可爱的 \小狗 \舔了 \我的 \脸





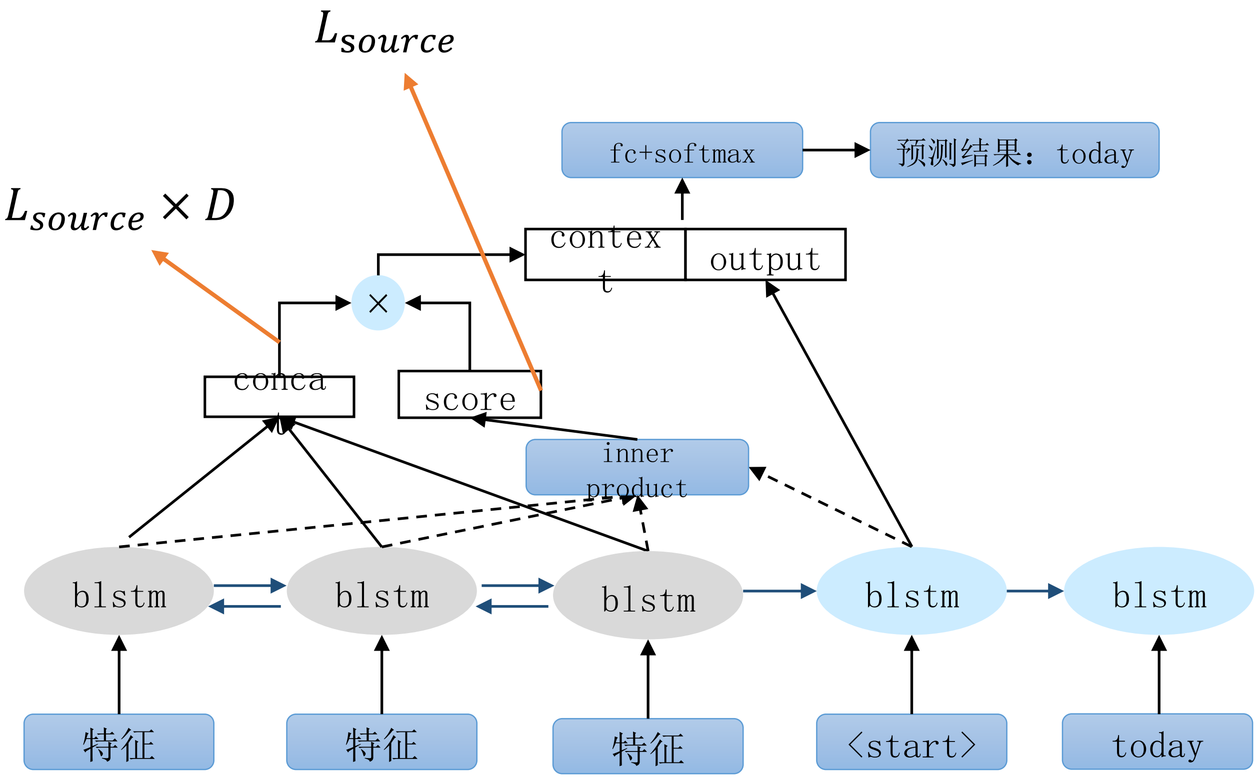
2. Neural Machine Translation（sys2sys）

- 编码： N1×D维向量
- 解码： ?×E维向量
  - 输入词表：添加 [begin] 标签(optional)
  - 输出此表：添加 [end] 标签，遇到则停止



sys2sys

- 基于attention的sys2sys
  - 把所有序列信息利用起来



3. Transformer

- 基于Attention模型的编码
  - RNN完成了N1×D到N1×E的变换
    - long term ?
  - 利用Attention模型完成相似的功能
    - non-local

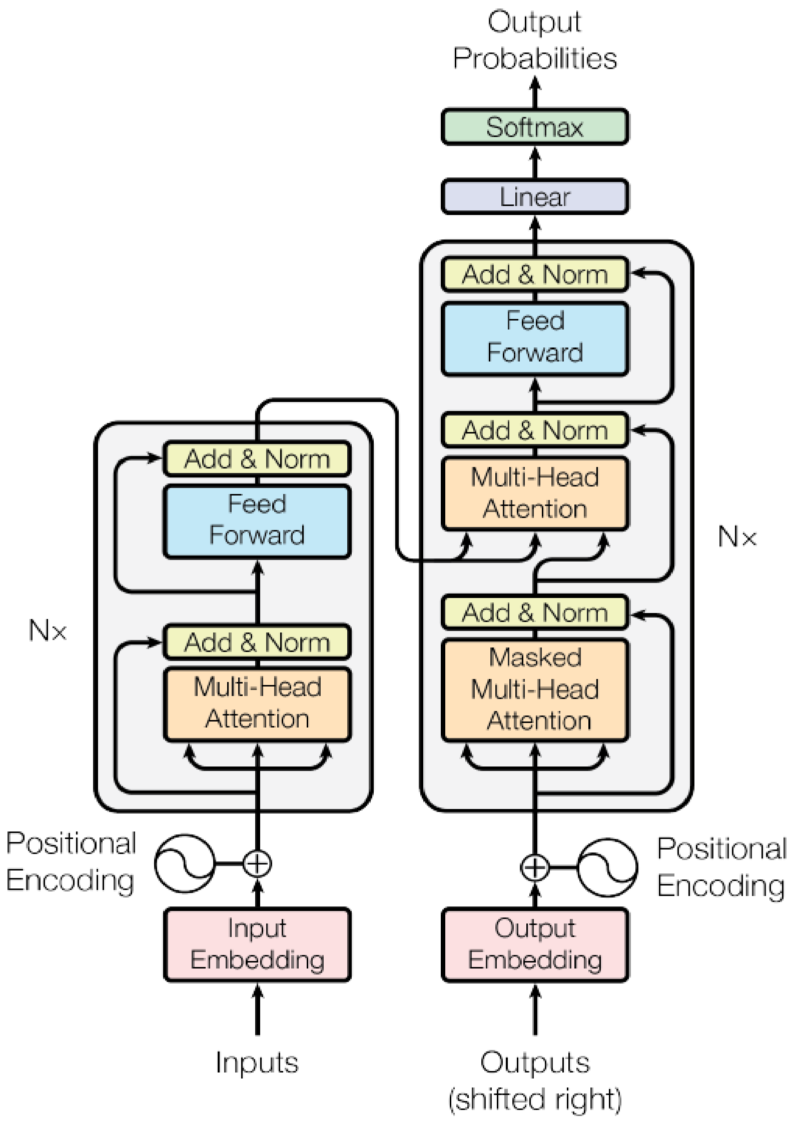
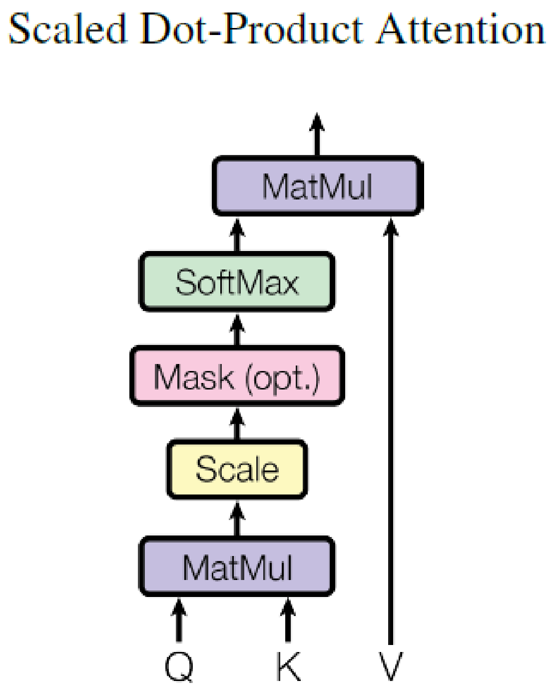


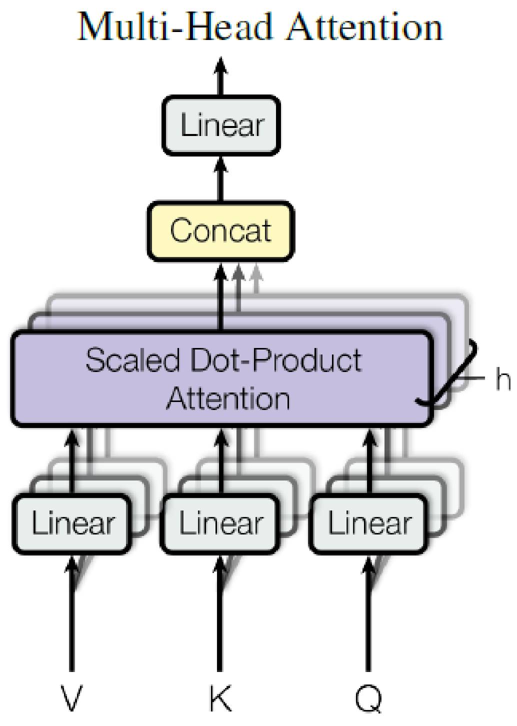
Figure 1: The Transformer - model architecture.

3. Transformer

- scaled dot-product attention
  - query: T1×D
  - key: T2×D
  - value: T2×E
  - 利用T1×T2的mask得到T1×E
    - 此处E和D可以相同，即为RNN的作用

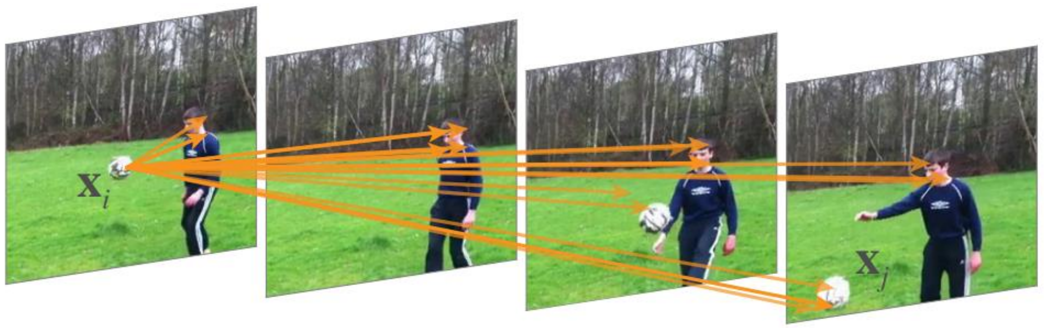
- multi-head attention
  - 分组，重复attention操作，把结果拼接

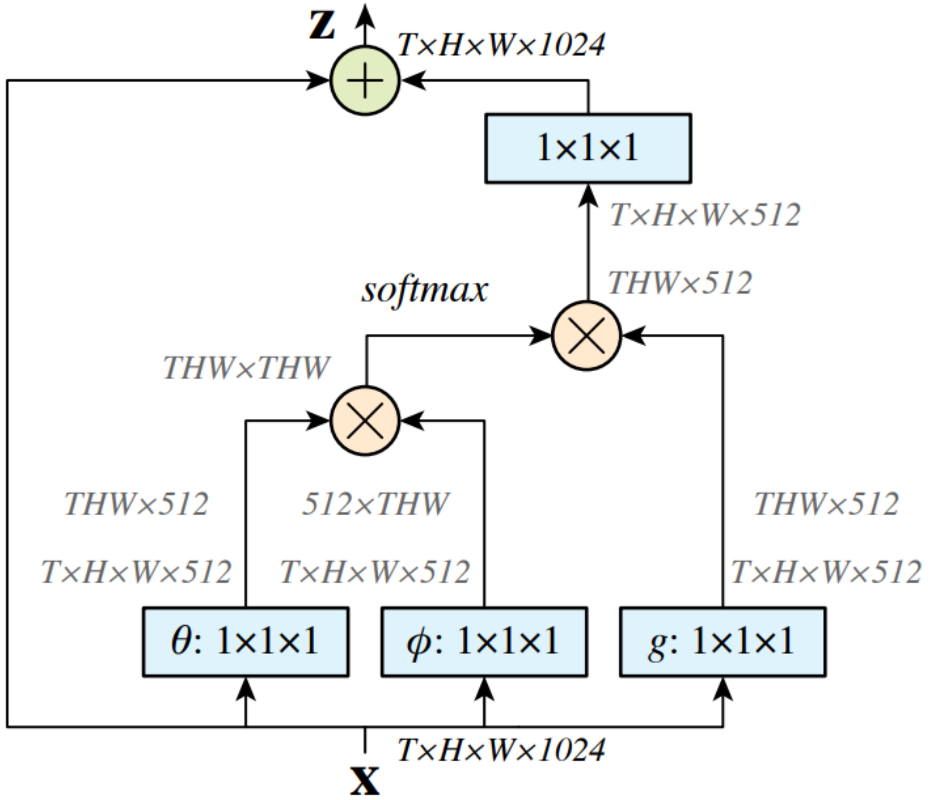




$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Non-local Neural Networks(best performance)





- Non-local Similarity
- Input-Dependent Matrix

Xiaolong Wang "Non-local Neural Networks", CVPR 2018