2022春

模式识别与深度学习 (33-34)

深度序列建模-2

左旺孟

综合楼309
机器学习研究中心
哈尔滨工业大学计算机学院
cswmzuo@gmail.com
13134506692

循环神经网络

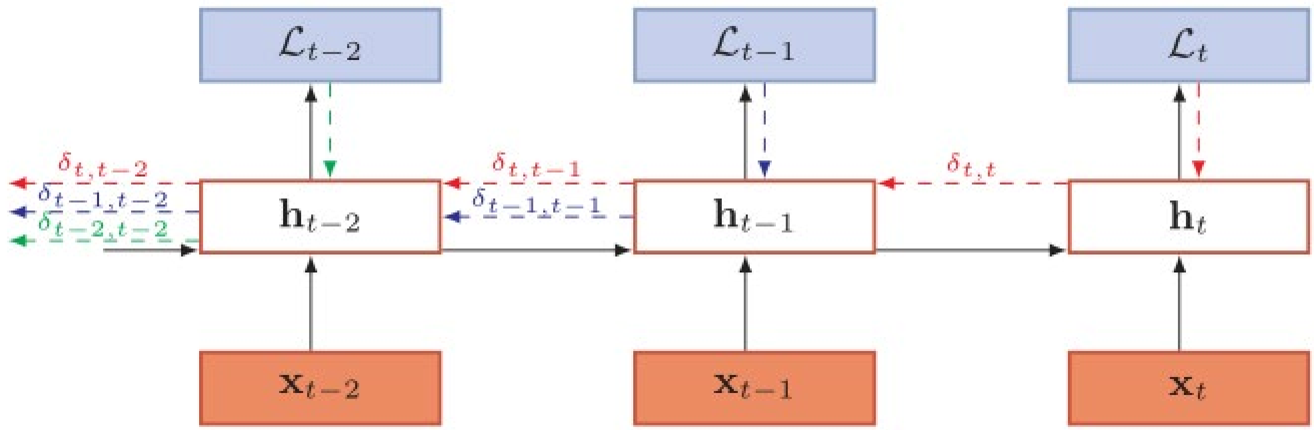
- 循环神经网络（Recurrent NN）
- 双向RNN
- 序列到序列模型
- 长短期记忆（LSTM）、GRU

长短期记忆

- 长期依赖
- 启发式解决方案
- GRU
- LSTM

BPTT

$$\mathbf{h}_{t+1} = f(\mathbf{z}_{t+1}) = f(U\mathbf{h}_t + W\mathbf{x}_{t+1} + \mathbf{b})$$

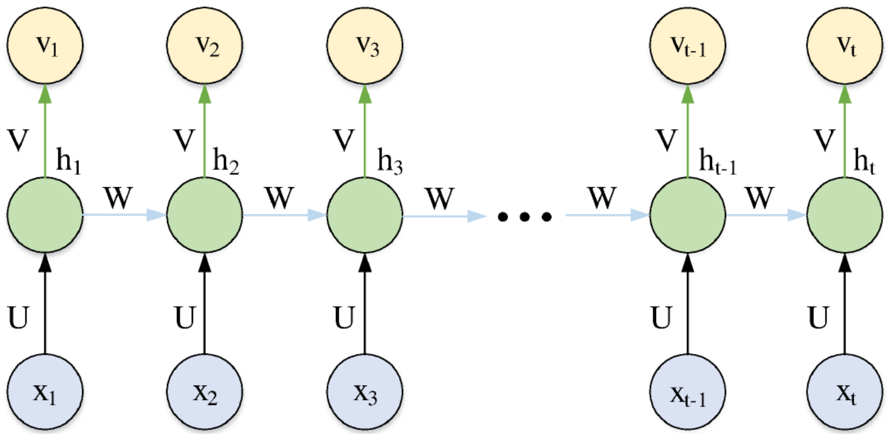


$$\frac{\partial \mathcal{L}}{\partial U} = \sum_{t=1}^{T}\sum_{k=1}^{t} \delta_{t,k}\mathbf{h}_{k-1}^{\mathrm{T}}$$

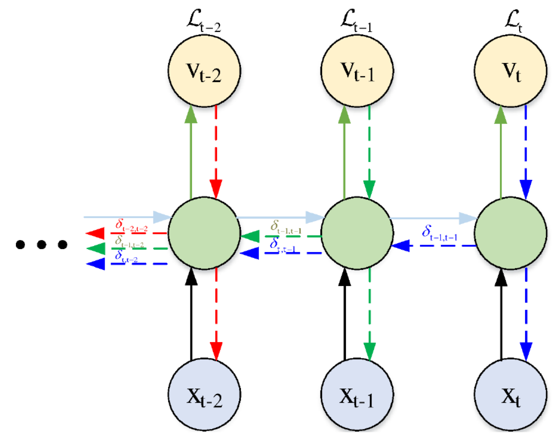
$$\delta_{t,k} = \prod_{\tau=k}^{t-1}\left(\mathrm{diag}(\boxed{f'(\mathbf{z}_\tau)})U^{\mathrm{T}}\right)\delta_{t,t}$$

$\delta_{t,k}$为第t时刻的损失对第k步隐藏神经元的净输出z_k的导数

长期依赖（Long-Term Denpendency）



前向



梯度

- 梯度计算

$$\begin{aligned}\nabla_{h^{(t)}} L &= \left(\frac{\partial \boldsymbol{h}^{(t+1)}}{\partial \boldsymbol{h}^{(t)}}\right)^{\top} (\nabla_{h^{(t+1)}} L) + \left(\frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{h}^{(t)}}\right)^{\top} (\nabla_{o^{(t)}} L) \\ &= \boldsymbol{W}^{\top} (\nabla_{h^{(t+1)}} L) \operatorname{diag}\left(1 - (\boldsymbol{h}^{(t+1)})^2\right) + \boldsymbol{V}^{\top} (\nabla_{o^{(t)}} L),\end{aligned}$$

$$\begin{aligned}\nabla_W L &= \sum_t \sum_i \left(\frac{\partial L}{\partial h_i^{(t)}}\right) \nabla_{W^{(t)}} h_i^{(t)} \\ &= \sum_t \operatorname{diag}\left(1 - (\boldsymbol{h}^{(t)})^2\right) (\nabla_{h^{(t)}} L) \boldsymbol{h}^{(t-1)^{\top}}\end{aligned}$$

$$\frac{\partial \mathcal{L}}{\partial W} = \sum_{t=1}^{T} \sum_{k=1}^{t} \left(\prod_{i=k}^{t-1} \operatorname{Diag}\left(f'\left(\mathbf{z}_i\right)\right) W^T \right) \delta_{t,t} \mathbf{h}_{k-1}^T$$

长期依赖（Long-Term Denpendency）

- 假设 $\gamma \approx \left\| \mathrm{Diag}\left(f'\left(\mathbf{z}_i\right)\right) W^T \right\|$

$$\frac{\partial \mathcal{L}}{\partial W} = \sum_{t=1}^{T} \sum_{k=1}^{t} \gamma^{t-k} \delta_{t,t} \mathbf{h}_{k-1}^T$$

  - 当 $\gamma > 1$，$t-k \to \infty$ 时，$\gamma^{t-k} \to \infty$, 此时会产生**梯度爆炸**
  - 当 $\gamma < 1$，$t-k \to \infty$ 时，$\gamma^{t-k} \to 0$，从而出现和前馈神经网络类似的梯度消失问题
  - 当 $t-k$ 较大时，时刻 t 损失函数 $\mathcal{L}_t$ 产生的梯度无法对 $t-k$ 时刻之前的参数 W 产生影响
  - 长期依赖：当间隔 k 较大时，网络无法对长时间间隔的数据依赖关系进行建模

长短期记忆

- 长期依赖
- 启发式解决方案
- GRU
- LSTM

缓解梯度爆炸

- 截断梯度 （Gradient clipping）
  - 是当参数的梯度大于一定阈值时，就将其截断为一个较小的数值
  - 方式1：在参数更新之前，**逐元素**地截断Mini-batch产生的参数梯度
  - 方式2：在参数更新之前，**整体**约束参数梯度大小（不改变梯度方向）

$$\mathbf{g} = \begin{cases} \frac{\mathbf{g}v}{\|\mathbf{g}\|}, & \text{if } \|\mathbf{g}\| > v \\ \mathbf{g}, & \text{else} \end{cases}$$

  - 实际应用中，两种方式性能表现类似

缓解梯度消失

- 时间维度的跳跃连接
  - 直接构造从t时刻单元到$t+d$时刻单元的连接

- 渗漏单元
  - 令 $W = \mathrm{I}$，$f'(\mathbf{z}_i) = \mathbf{1}$

$$\mathbf{h}_t = \mathbf{h}_{t-1} + \mathbf{g}(\mathbf{x}_t, \phi)$$

  - 丢失了神经元上存在的非线性激活性质，降低了网络的拟合能力

缓解梯度消失：渗漏单元

- 记忆容量（Memory Capacity）问题
  - 随着$\mathbf{h}_t$不断累积存储过去的输出状态，会发生“饱和”现象
- **渗漏单元**（Leaky Unit）

$$\mathbf{h}_t = \mu \mathbf{h}_{t-1} + (1-\mu)\,\mathbf{g}\left(\mathbf{x}_t, \mathbf{h}_{t-1}, \phi\right)$$

  - 当μ 接近于1 时，神经网络能够记住过去很长一段时间的信息；
  - 当μ 接近于0 时，关于过去的信息会被快速丢弃

  - 超参数μ：可以预设，也通过数据驱动的方式学习

长短期记忆

- 长期依赖
- 启发式解决方案
- GRU
- LSTM

K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.

门控循环单元（GRU）

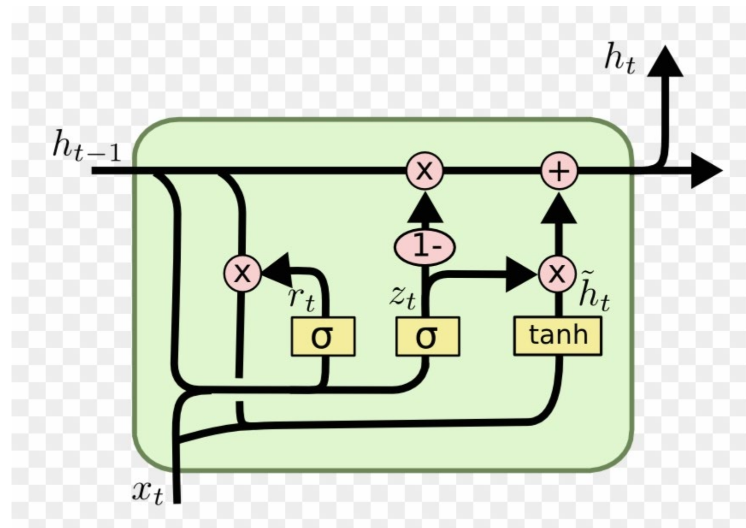
- 引入门控机制：由神经网络学会决定何时清除状态

$$\mathbf{h}_t = \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \tilde{\mathbf{h}}_t,$$

$$\tilde{\mathbf{h}}_t = \tanh\left(W_h \mathbf{x}_t + U_h \left(\mathbf{r}_t \odot \mathbf{h}_{t-1}\right) + b_h\right),$$

$$\mathbf{z}_t = \delta\left(W_z \mathbf{x}_t + U_z \mathbf{h}_{t-1} + b_z\right),$$

$$\mathbf{r}_t = \delta\left(W_r \mathbf{x}_t + U_r \mathbf{h}_{t-1} + b_r\right),$$



  - $\mathbf{z}_t$: 更新门（Update Gate）
  - $\mathbf{r}_t$: 重置门（Reset Gate）

解释

- 当$\mathbf{z}_t = 0$ 时，当前状态$\mathbf{h}_t$ 和历史状态$\mathbf{h}_{t-1}$ 只存在非线性关系

- 当$\mathbf{z}_t = 0$，$\mathbf{r}_t = 1$，GRU 网络则退化为简单循环神经网络

- 当$\mathbf{z}_t = 0$，$\mathbf{r}_t = 0$， GRU 网络退化为传统的前馈神经网络

- 当$\mathbf{z}_t = 1$ 时，当前时刻的隐藏层输出$\mathbf{h}_{t+1}$ 等于上一时刻的隐藏层输出$\mathbf{h}_t$，而与当前输入$\mathbf{x}_t$ 无关

GRU与优化算法的联系

$$\min_{\mathbf{s}} \sum_i \|\mathbf{x}_i - \mathbf{B}\mathbf{s}_i\|_F^2 + \lambda\|\mathbf{s}_i\|_1$$

- 投影梯度下降

$$\begin{aligned}\mathbf{s}^{(t)} &= sh_{(\lambda\tau)}\left(\mathbf{s}^{(t-1)} - \tau\left(\mathbf{B}^T\left(\mathbf{B}\mathbf{s}^{(t-1)} - \mathbf{X}\right)\right)\right) \\ &= sh_{(\lambda\tau)}\left(\mathbf{W}_e\mathbf{s}^{(t-1)} + \mathbf{W}_d\mathbf{x}\right),\end{aligned}$$

- Nesterov加速梯度

$$\mathbf{x}_k = \mathbf{y}_{k-1} - \varepsilon\nabla F(\mathbf{y}_{k-1}) \quad (\varepsilon \le 1/L_F)$$

$$t_{k+1} \leftarrow (1 + \sqrt{1 + 4t_k^2})/2,$$

$$\boxed{\mathbf{y}_{k+1} \leftarrow \mathbf{x}_k + (t_k - 1)/t_{k+1}(\mathbf{x}_k - \mathbf{x}_{k-1})}$$

GRU与优化算法的联系

- 问题 $\min_{\mathbf{s}} \sum_i \|\mathbf{x}_i - \mathbf{B}\mathbf{s}_i\|_F^2 + \lambda\|\mathbf{s}_i\|_1$
- 改进Nesterov加速

$$\begin{aligned}\tilde{\mathbf{c}}^{(t)} &= \mathbf{W}_e \mathbf{s}^{(t-1)} + \mathbf{W}_d \mathbf{x},\\ \mathbf{c}^{(t)} &= \mathbf{f}^{(t)} \odot \mathbf{c}^{(t-1)} + \mathbf{i}^{(t)} \odot \tilde{\mathbf{c}}^{(t)}\\ \mathbf{s}^{(t)} &= sh_{(\lambda\tau)}(\mathbf{c}^{(t)}),\end{aligned}$$

$$\begin{aligned}\mathbf{h}_t &= \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1-\mathbf{z}_t) \odot \tilde{\mathbf{h}}_t,\\ \tilde{\mathbf{h}}_t &= \tanh\left(W_h \mathbf{x}_t + U_h\left(\mathbf{r}_t \odot \mathbf{h}_{t-1}\right) + b_h\right),\end{aligned}$$

- 对比GRU

$$\begin{aligned}\mathbf{i}^{(t)} &= \sigma(\mathbf{W}_{is}\mathbf{s}^{(t-1)} + \mathbf{W}_{ix}\mathbf{x}),\\ \mathbf{f}^{(t)} &= \sigma(\mathbf{W}_{fs}\mathbf{s}^{(t-1)} + \mathbf{W}_{fx}\mathbf{x}),\\ \tilde{\mathbf{c}}^{(t)} &= \mathbf{W}_e\mathbf{s}^{(t-1)} + \mathbf{W}_d\mathbf{x},\\ \mathbf{c}^{(t)} &= \mathbf{f}^{(t)} \odot \mathbf{c}^{(t-1)} + \mathbf{i}^{(t)} \odot \tilde{\mathbf{c}}^{(t)}\\ \mathbf{s}^{(t)} &= h_{(\mathbf{D},\mathbf{u})}(\mathbf{c}^{(t)}),\end{aligned}$$

$$\begin{aligned}\mathbf{z}_t &= \delta\left(W_z\mathbf{x}_t + U_z\mathbf{h}_{t-1} + b_z\right),\\ \mathbf{r}_t &= \delta\left(W_r\mathbf{x}_t + U_r\mathbf{h}_{t-1} + b_r\right),\\ \tilde{\mathbf{h}}_t &= \tanh\left(W_h\mathbf{x}_t + U_h\left(\mathbf{r}_t \odot \mathbf{h}_{t-1}\right) + b_h\right),\\ \mathbf{h}_t &= \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1-\mathbf{z}_t) \odot \tilde{\mathbf{h}}_t,\end{aligned}$$

Joey Tianyi Zhou et al., SC2Net: Sparse LSTMs for Sparse Coding, AAAI 2018.

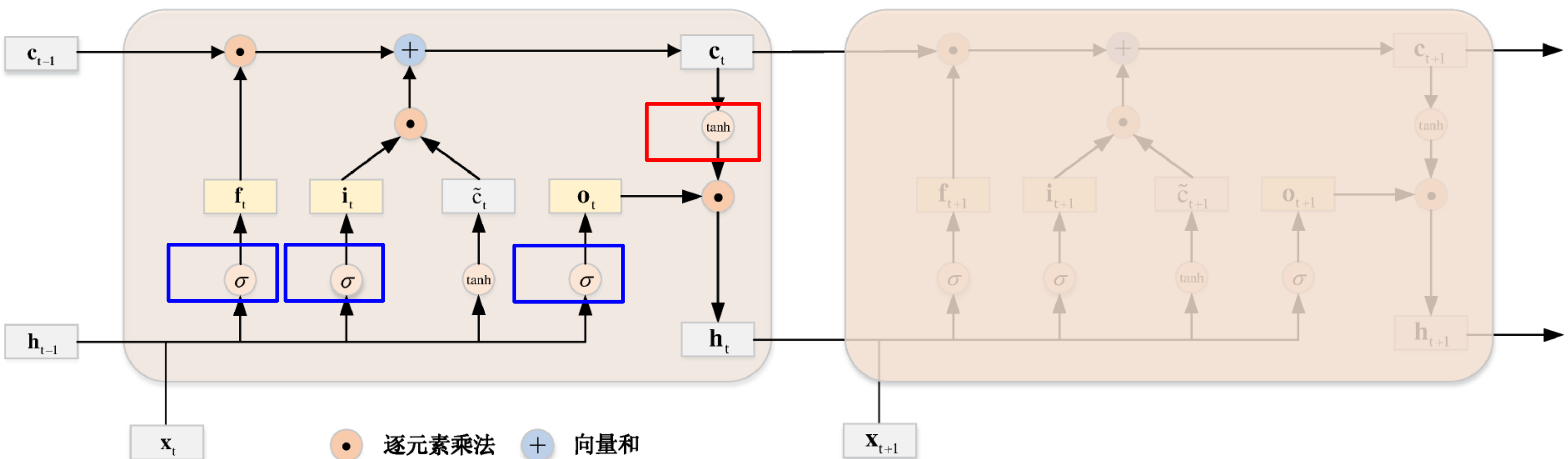
长短期记忆

- 长期依赖
- 启发式解决方案
- GRU
- LSTM

S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997

长短期记忆（long short-term memory）

- 新的记忆单元$\mathbf{c}_t$，用于控制信息的线性传递
  - 候选内部状态 $\tilde{\mathbf{c}}_t$
- 三个门组件
  - 输入门（Input Gate）$\mathbf{i}_t$，
  - 遗忘门（Forget Gate）$\mathbf{f}_t$
  - 输出门（Output Gate）$\mathbf{o}_t$



长短期记忆（LSTM）

- 计算过程
  - 更新门组件

$$\mathbf{i}_t = \delta\left(W_i \mathbf{x}_t + U_i \mathbf{h}_{t-1} + b_i\right),$$

$$\mathbf{f}_t = \delta\left(W_f \mathbf{x}_t + U_f \mathbf{h}_{t-1} + b_f\right),$$

$$\mathbf{o}_t = \delta\left(W_o \mathbf{x}_t + U_o \mathbf{h}_{t-1} + b_o\right).$$

  - 候选内部状态更新

$$\tilde{\mathbf{c}}_t = \mathbf{tanh}\left(W_c \mathbf{x}_t + U_c \mathbf{h}_{t-1} + b_c\right)$$

  - 记忆单元和隐藏单元更新

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t,$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \mathbf{tanh}\left(\mathbf{c}_t\right),$$

长短期记忆（LSTM）

- 通过前一时刻的输出状态$\mathbf{h}_{t-1}$和当前时刻输入$\mathbf{x}_t$计算当前时刻三个门的输出$\mathbf{f}_t$，$\mathbf{i}_t$和$\mathbf{o}_t$;
- 计算当前时刻候选内部状态 $\tilde{\mathbf{c}}_t$， 同时结合上一时刻的记忆单元输出$\mathbf{c}_{t-1}$和$\mathbf{f}_t$，计算当前时刻的记忆单元输出$\mathbf{c}_t$。
- 结合输出门$\mathbf{o}_t$，计算当前时刻隐藏单元的最终输出$\mathbf{h}_t$。

门机制解释

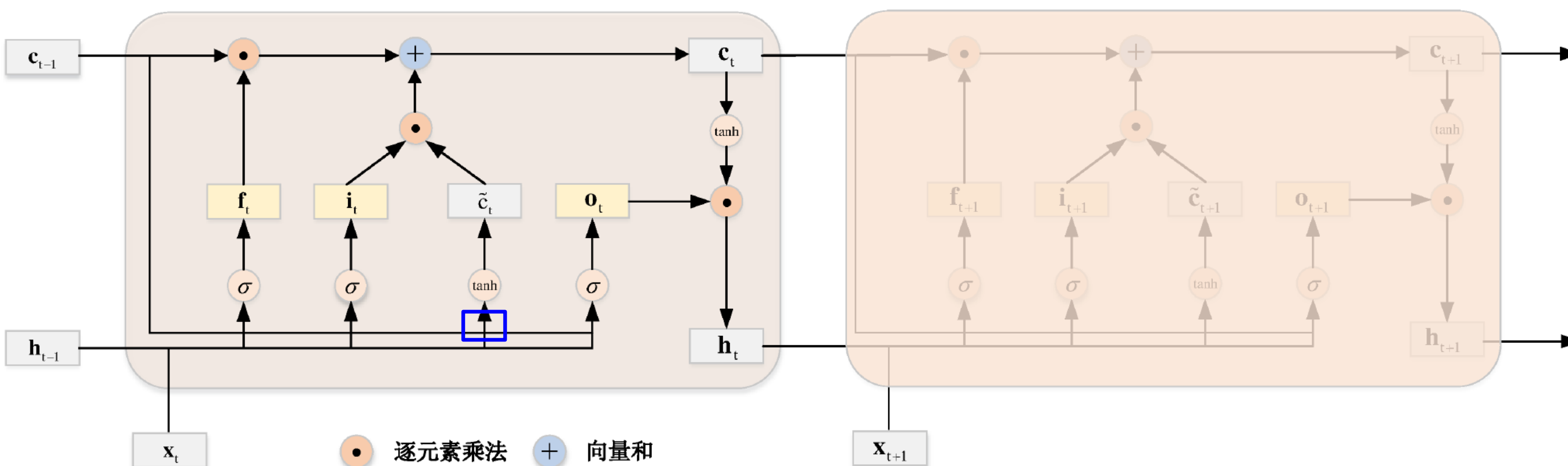
- 当$\mathbf{f}_t = 0$，$\mathbf{i}_t = 1$ 时，记忆单元将历史信息清空，并将候选内部状态 $\tilde{\mathbf{c}}_t$ 写入；
- 当$\mathbf{f}_t = 1$，$\mathbf{i}_t = 0$ 时，记忆单元将复制上一时刻的内容，不写入新的信息。
- 外部的RNN 循环+内部的LSTM 细胞循环（自环）

变体1：带有peephole 连接的LSTM

$$\mathbf{i}_t = \delta\left(W_i \mathbf{x}_t + U_i \mathbf{h}_{t-1} + V_i \mathbf{c}_{t-1} + b_i\right),$$

$$\mathbf{f}_t = \delta\left(W_f \mathbf{x}_t + U_f \mathbf{h}_{t-1} + V_f \mathbf{c}_{t-1} + b_f\right),$$

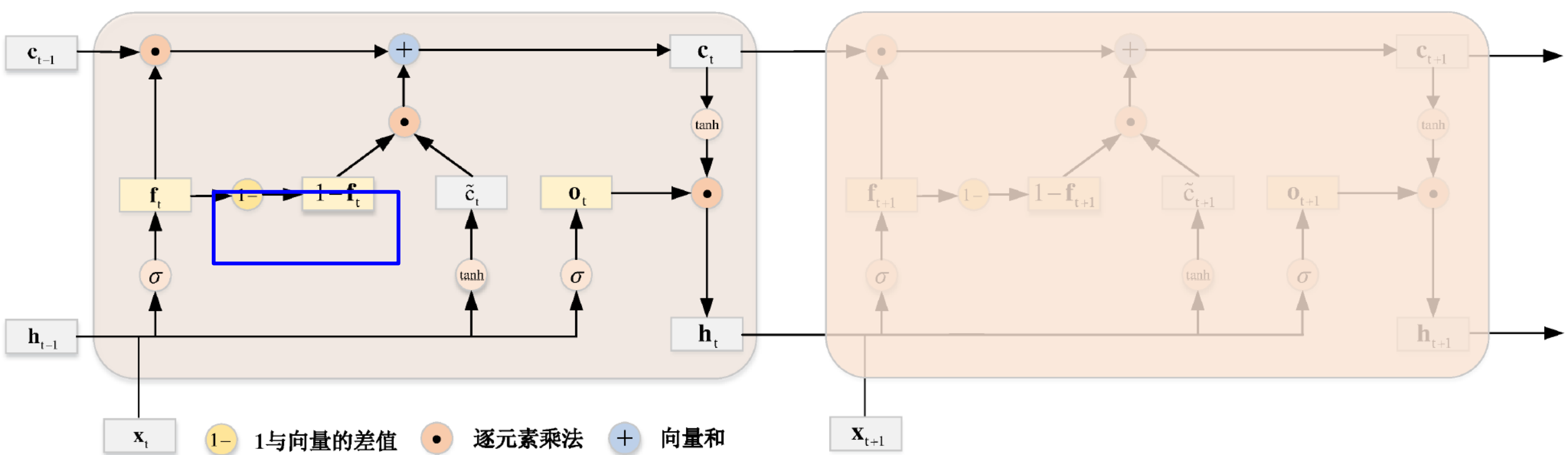
$$\mathbf{o}_t = \delta\left(W_o \mathbf{x}_t + U_o \mathbf{h}_{t-1} + V_o \mathbf{c}_{t-1} + b_o\right).$$



变体2：耦合输入门和遗忘门的LSTM

$$\mathbf{i}_t = 1 - \mathbf{f}_t$$

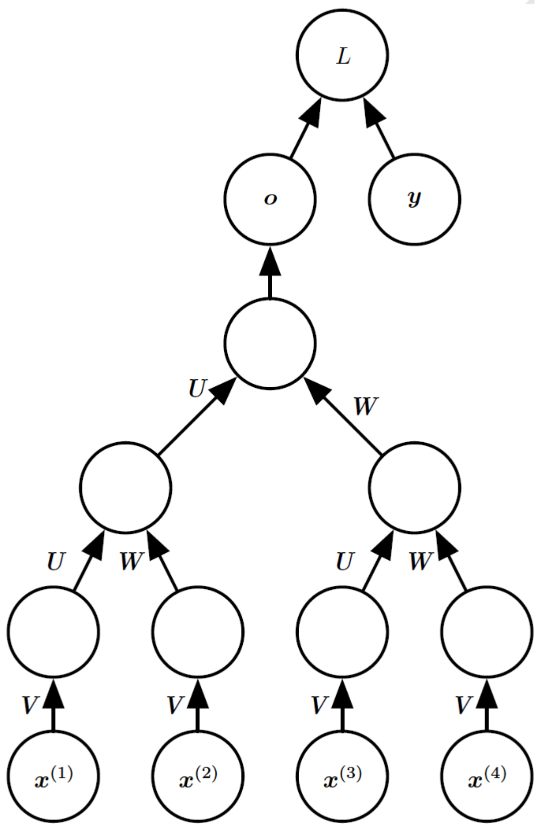
$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + (1 - \mathbf{f}_t) \odot \tilde{\mathbf{c}}_t$$

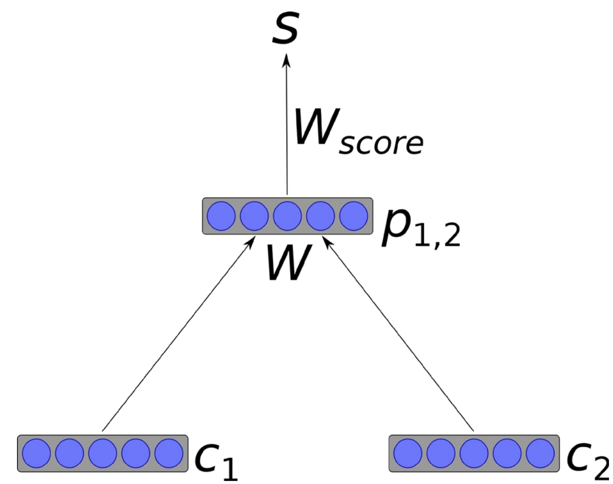


序列建模

- 循环神经网络
- 递归神经网络
- 记忆网络
- 图神经网络

递归神经网络





递归网络基本单元

$$p_{1,2} = \tanh(W[c_1; c_2])$$

递归网络将循环网络的链状计算图推广到树状计算图

给定树结构，网络深度可从O(T)降至O(logT)

递归神经网络

- 如何以最佳的方式构造树
  - 使用不依赖于数据的树结构
  - 借鉴外部方法选择适当的树结构（语法树）
  - 自行发现和推断适合于任意给定输入的树结构（层次聚类）

序列建模

- 循环神经网络
- 递归神经网络
- 记忆网络
- 图神经网络

知识的种类与表达

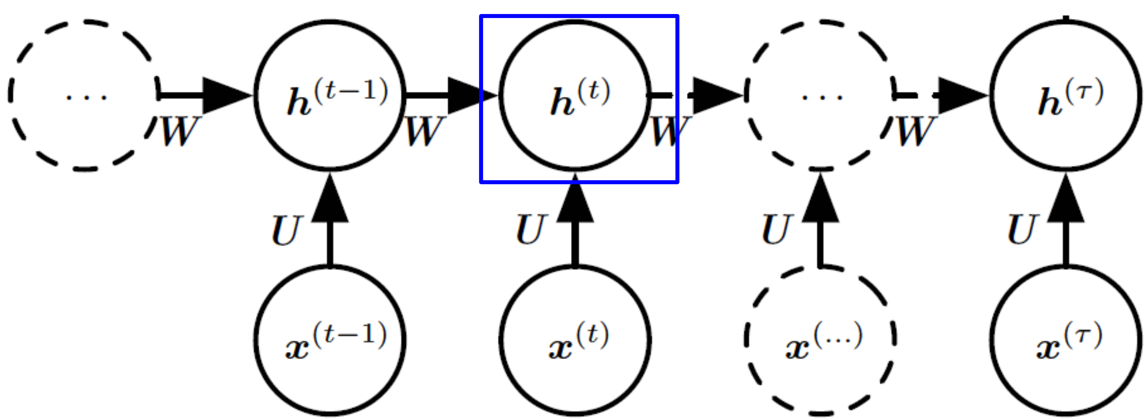
- 隐性知识：隐含的、潜意识的并且难以用语言表达
  - 如：怎么行走或狗与猫的样子有什么不同
- 明确的、可陈述的以及可以相对简单地使用词语表达
  - 常识性的知识：猫是一种动物
  - 具体的事实：与销售团队会议在141室于下午3:00 开始

词语、概念和概念间的关系

记忆网络

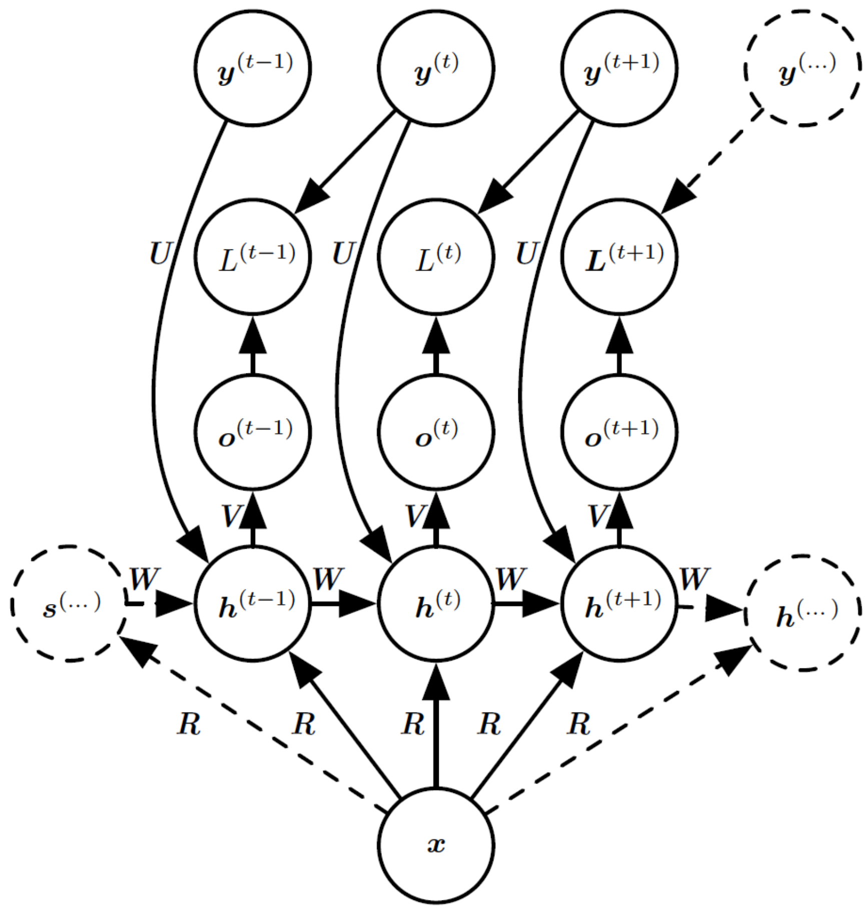
- 神经网络擅长存储隐性知识，但很难记住事实
  - 缺乏工作存储系统：外显记忆组件

- 如果在神经网络中引入外部知识？



Revisit: 基于上下文的RNN 序列建模

- 只使用单个向量$\boldsymbol{x}$ 作为输入



记忆网络

- 记忆网络：引入记忆单元
  - 需要监督信号指示他们如何使用自己的记忆单元

    Weston, J., Chopra, S., and Bordes, A. (2014). Memory networks. *arXiv preprint arXiv:1410.3916* .

- 神经网络图灵机：不需要明确的监督指示而能学习从记忆单元读写任意内容

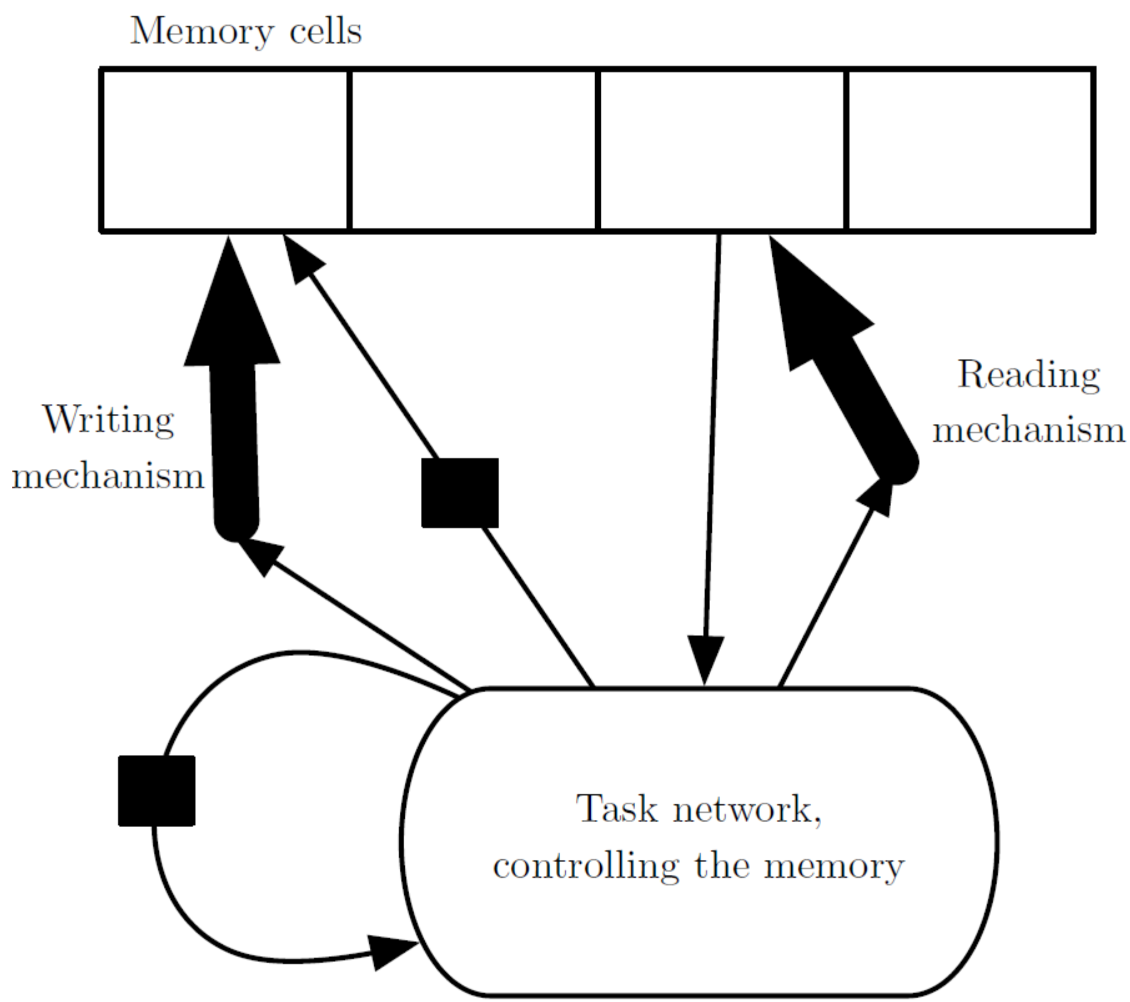
  Graves, A., Wayne, G., and Danihelka, I. (2014). Neural Turing machines. arXiv:1410.5401.

- 基于内容的软注意机制：端到端训练

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In ICLR'2015, arXiv:1409.0473 .

具有外显记忆的神经网络

- 记忆网络
- 神经网络图灵机 (NTM)



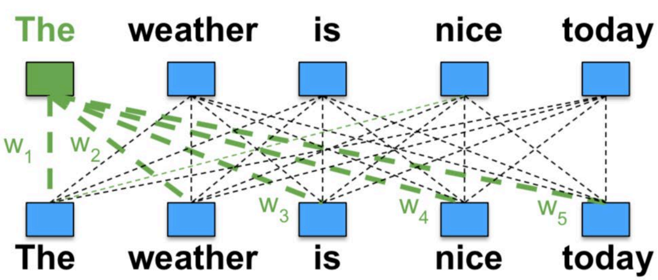
具有外显记忆的神经网络

- 记忆单元的写入和读取：
  - 避免整数寻址
  - 以概率形式同时从多个记忆单元写入或读取
    - 读取时，采取许多单元的加权平均值
    - 写入时，同时修改多个单元

- 使用向量值的记忆单元
  - 基于内容的寻址(content-based addressing)

  检索一首副歌歌词中带有’We all live in a yellow submarine’ 的歌

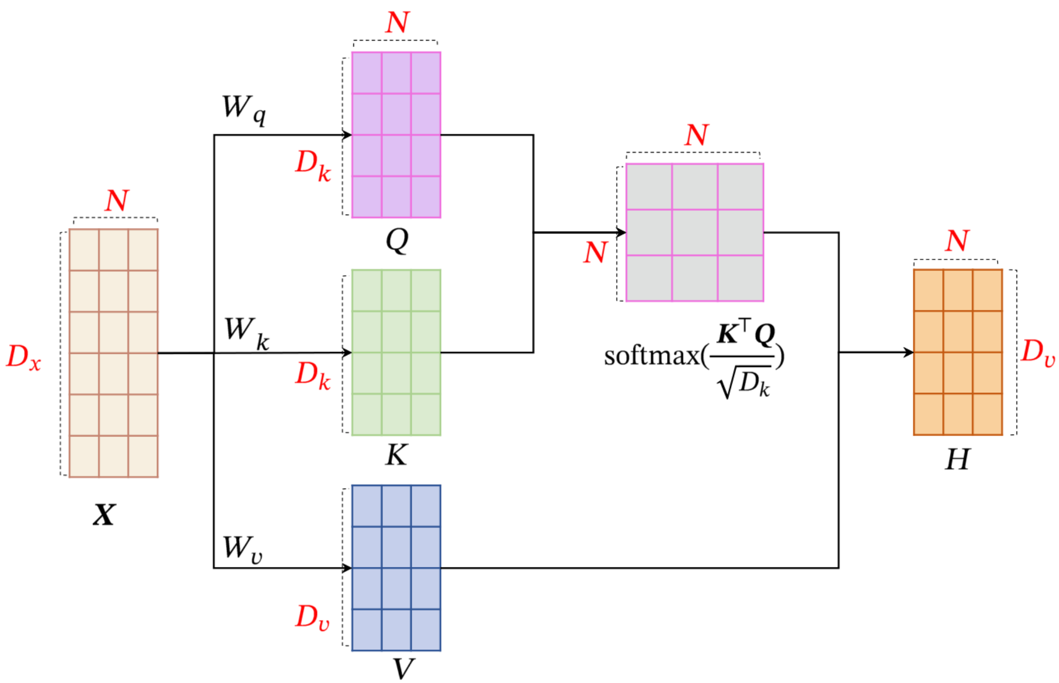
自注意力示例



$$w_1, w_2, w_3, w_4, w_5 = \text{softmax}\left(\underset{\text{The}}{\begin{bmatrix} 0.6 & 0.2 & 0.8 \end{bmatrix}} \times \begin{array}{ccccc} \begin{bmatrix} 0.6 \\ 0.2 \\ 0.8 \end{bmatrix} & \begin{bmatrix} 0.2 \\ 0.3 \\ 0.1 \end{bmatrix} & \begin{bmatrix} 0.9 \\ 0.1 \\ 0.8 \end{bmatrix} & \begin{bmatrix} 0.4 \\ 0.1 \\ 0.4 \end{bmatrix} & \begin{bmatrix} 0.4 \\ 0.1 \\ 0.6 \end{bmatrix} \\ \text{The} & \text{weather} & \text{is} & \text{nice} & \text{today} \end{array}\right)$$

$$\underset{\text{The}}{\begin{bmatrix} 1.8 \\ 2.3 \\ 0.4 \end{bmatrix}} = w_1 \times \underset{\text{The}}{\begin{bmatrix} 0.6 \\ 0.2 \\ 0.8 \end{bmatrix}} + w_2 \times \underset{\text{weather}}{\begin{bmatrix} 0.2 \\ 0.3 \\ 0.1 \end{bmatrix}} + w_3 \times \underset{\text{is}}{\begin{bmatrix} 0.9 \\ 0.1 \\ 0.8 \end{bmatrix}} + w_4 \times \underset{\text{nice}}{\begin{bmatrix} 0.4 \\ 0.1 \\ 0.4 \end{bmatrix}} + w_5 \times \underset{\text{today}}{\begin{bmatrix} 0.4 \\ 0.1 \\ 0.6 \end{bmatrix}}$$

QKV模式（Query-Key-Value）



自注意力模型

- 输入序列为$X = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N] \in R^{D_x \times N}$
- 首先生成三个向量序列

$$\boldsymbol{Q} = \boldsymbol{W}_q \boldsymbol{X} \in \mathbb{R}^{D_k \times N},$$

$$\boldsymbol{K} = \boldsymbol{W}_k \boldsymbol{X} \in \mathbb{R}^{D_k \times N},$$

$$\boldsymbol{V} = \boldsymbol{W}_v \boldsymbol{X} \in \mathbb{R}^{D_v \times N},$$

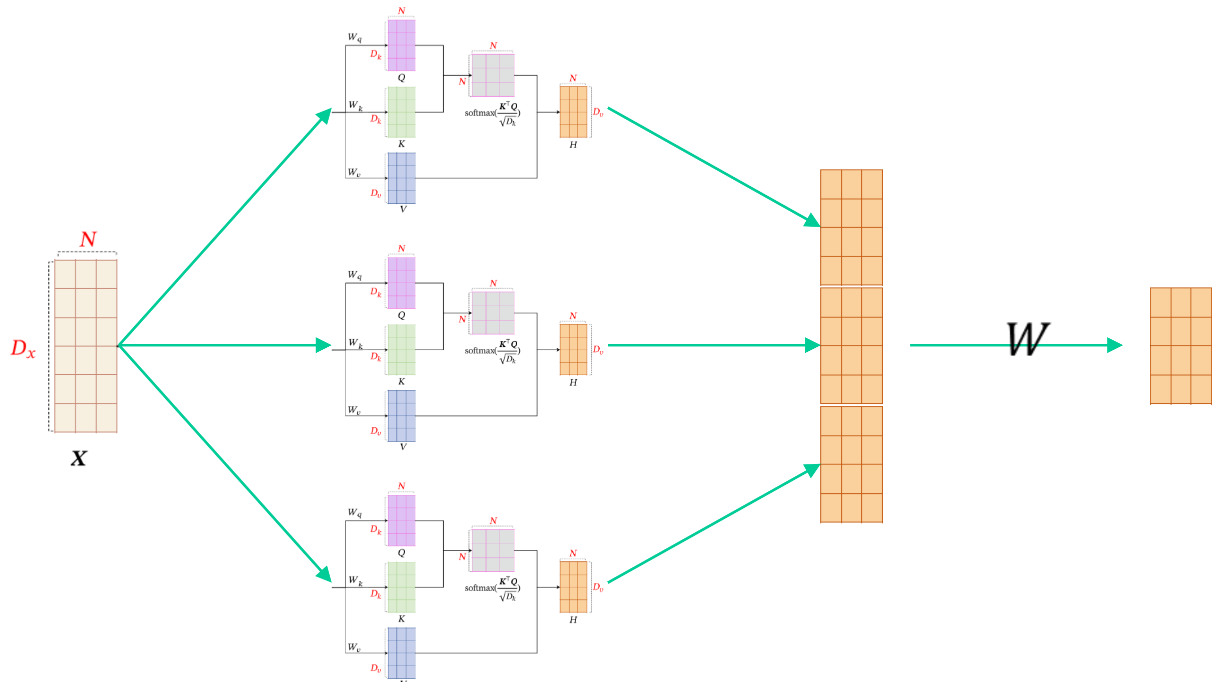
- 计算$\boldsymbol{h}_n$

$$\boldsymbol{h}_n = \mathrm{att}\Big((\boldsymbol{K}, \boldsymbol{V}), \boldsymbol{q}_n\Big)$$

- 如果使用缩放点积来作为注意力打分函数，输出向量序列可以简写为

$$\boldsymbol{H} = \boldsymbol{V}\,\mathrm{softmax}\Big(\frac{\boldsymbol{K}^\top \boldsymbol{Q}}{\sqrt{D_k}}\Big),$$

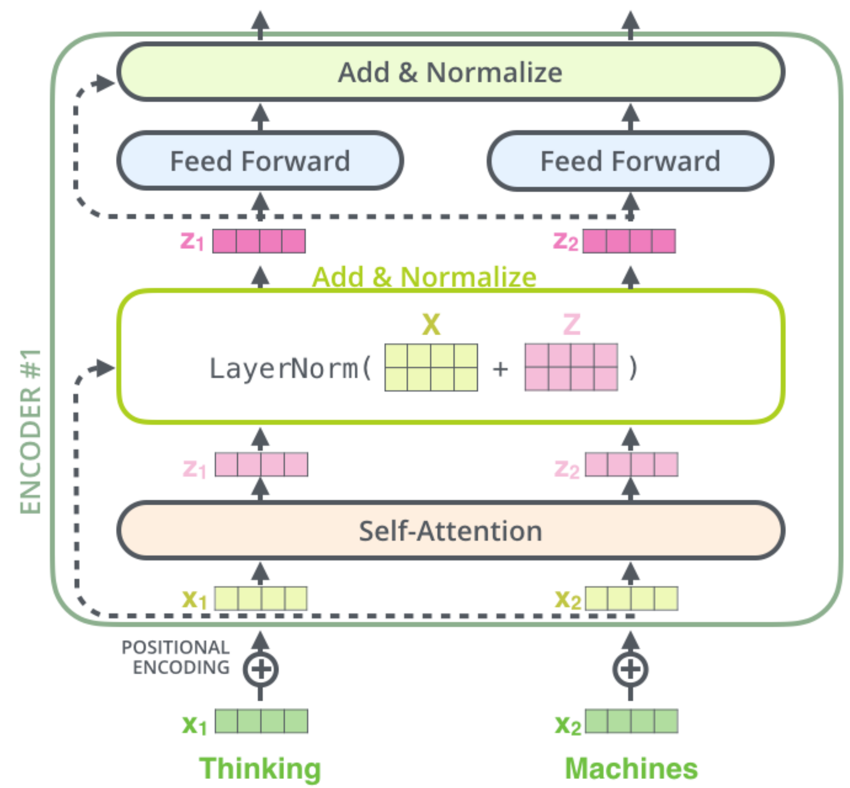
多头（multi-head）自注意力模型



Transformer Encoder

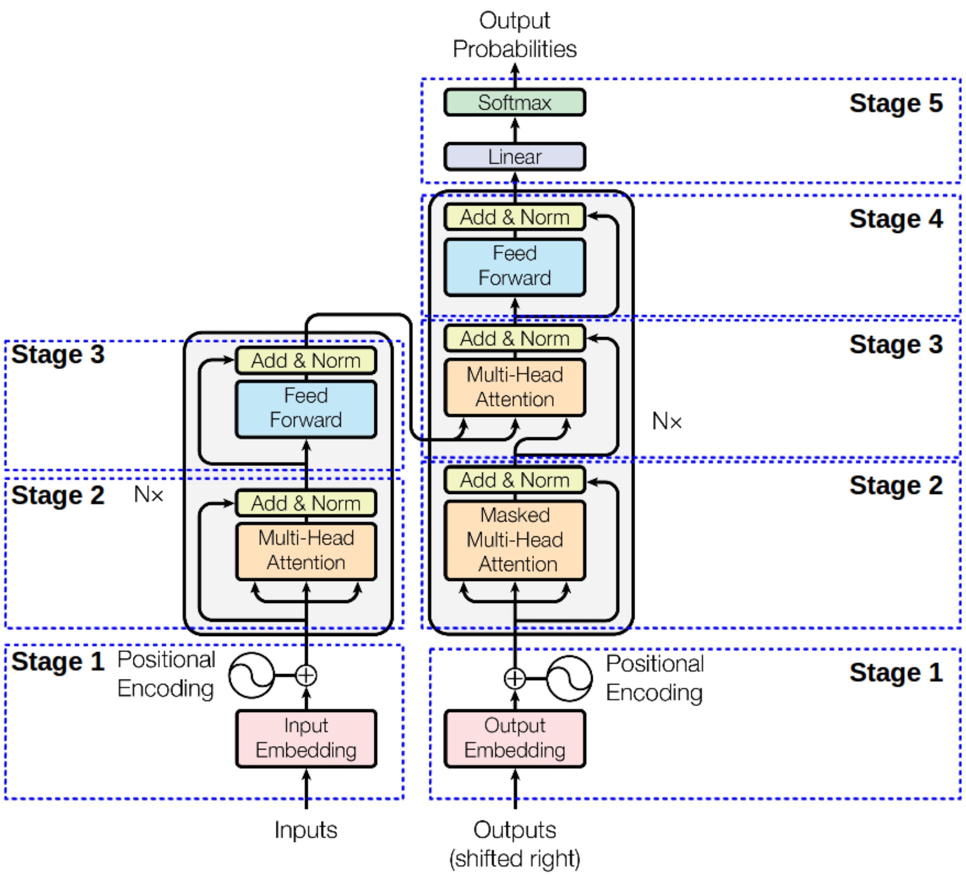
图片来源：http://jalammar.github.io/illustrated-transformer/

- 仅仅自注意力还不够
- 其它操作
  - 位置编码
  - 层归一化
  - 直连边
  - 逐位的FNN



Transformer

https://mchromiak.github.io/articles/2017/Sep/12/Transformer-Attention-is-all-you-need/#.W90QB5Mzabg

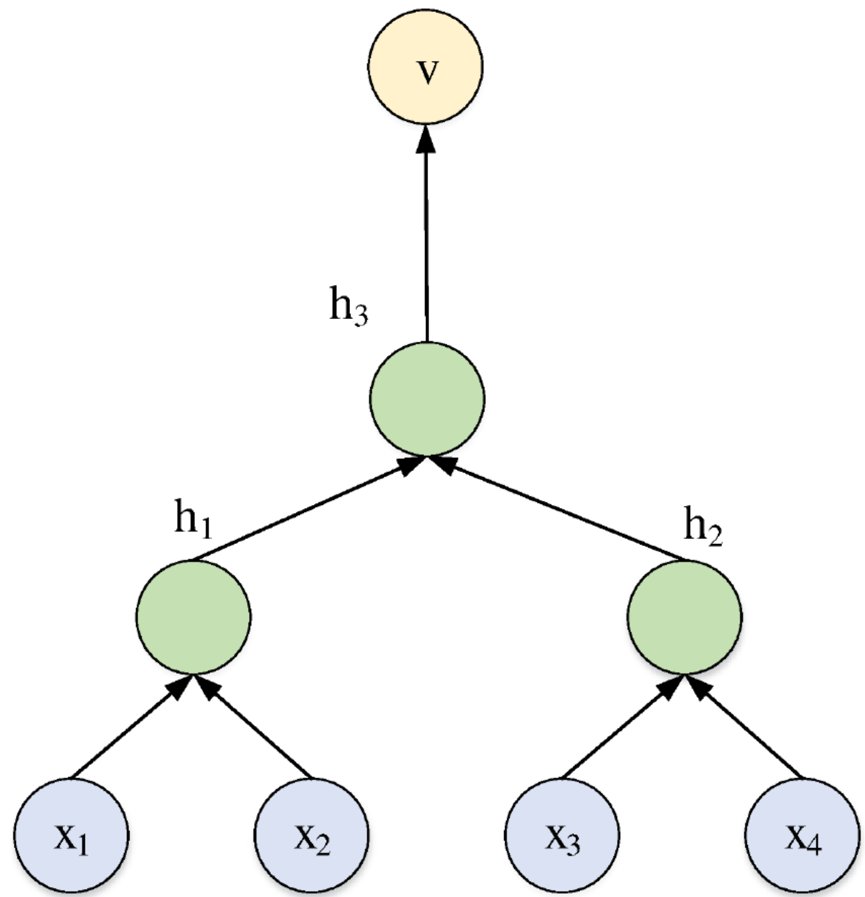


序列建模

- 循环神经网络
- 递归神经网络
- 记忆网络
- 图神经网络

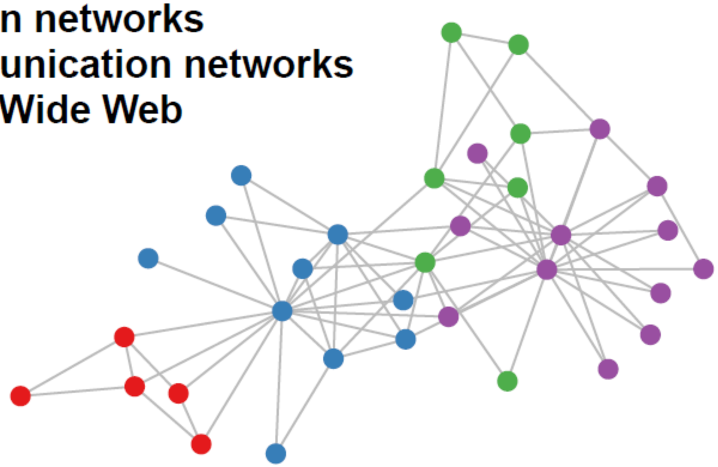
递归神经网络

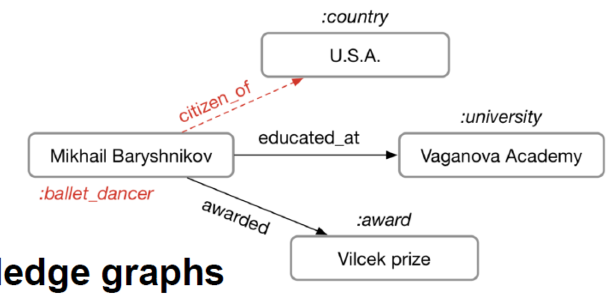
- 当输入数据为更为复杂的结构化数据
  - 节点分类
  - 关系预测



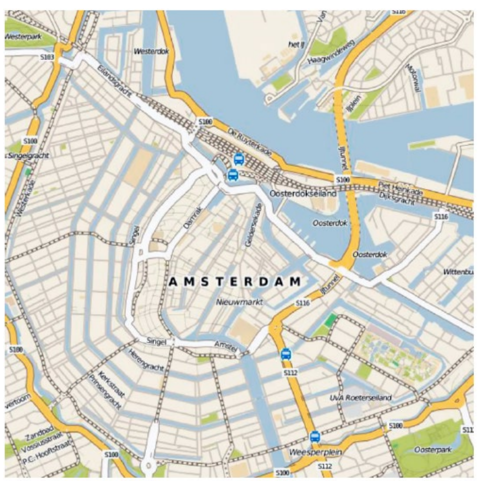
图数据

Social networks
Citation networks
Communication networks
World Wide Web

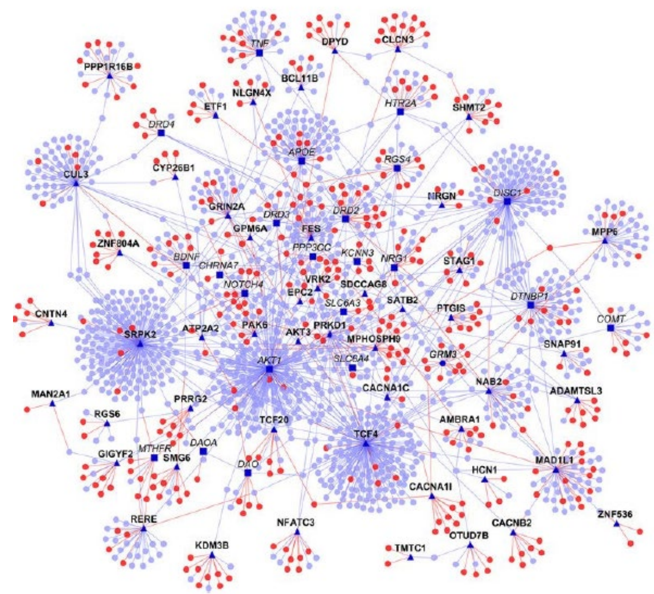




Knowledge graphs

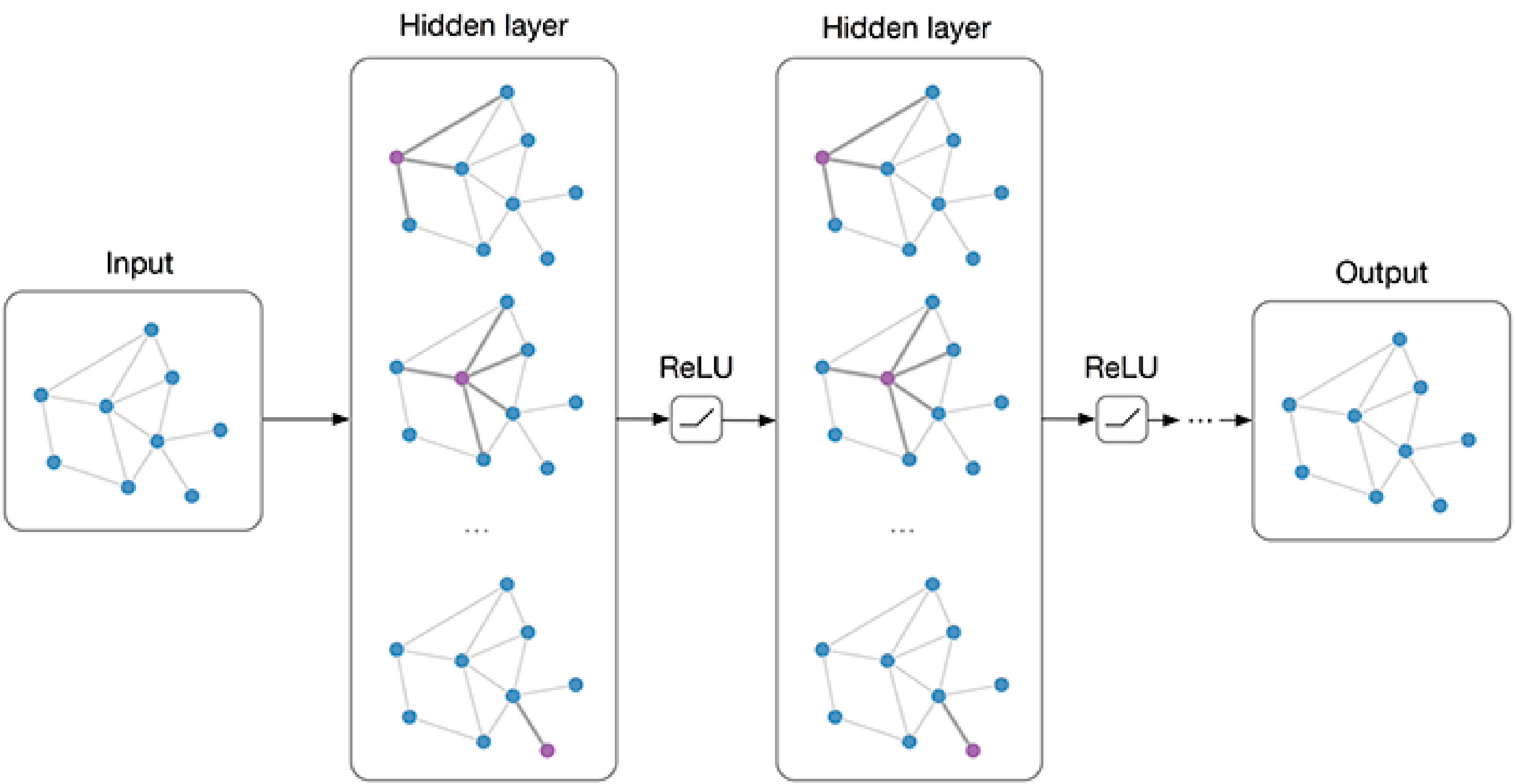


Road maps



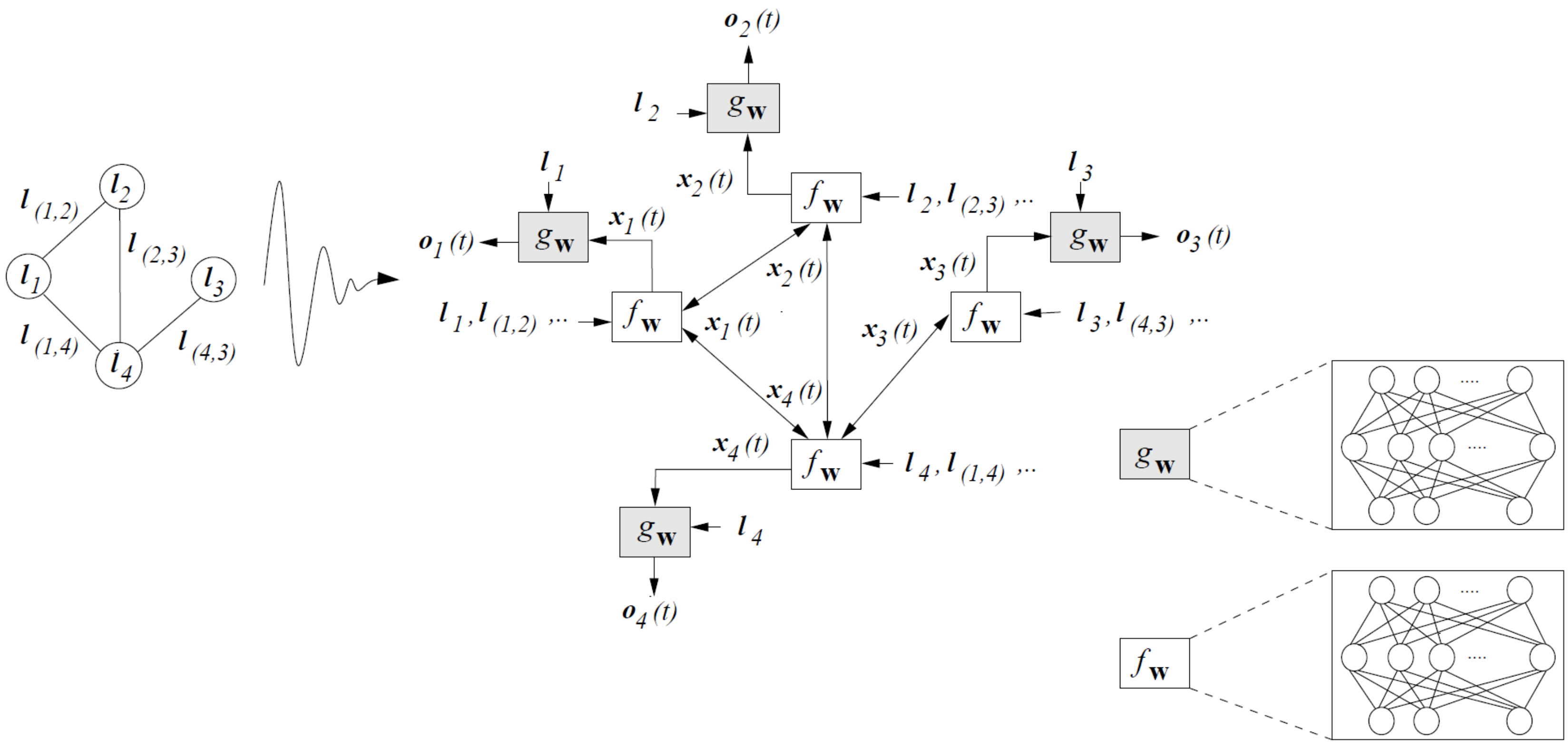
Protein interaction networks

图神经网络



T. N. Kipf, M. Welling, Semi-Supervised Classification with Graph Convolutional Networks (ICLR 2017)

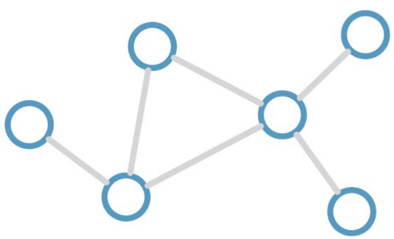
图神经网络



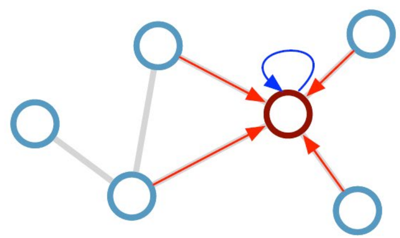
S. Franco, M. Gori, T. Ah Chung, H. Markus, and M. Gabriele, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, p. 61, 2009.

图神经网络

图



节点



局限性：

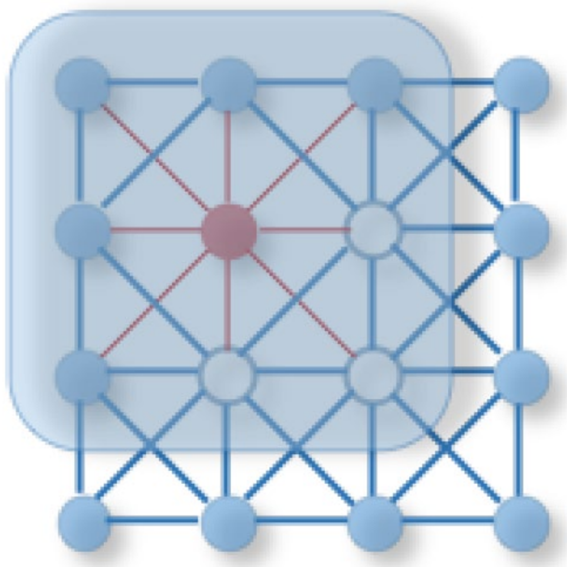
1. 参数不能共享
2. 局部 -> 全局

- 计算公式

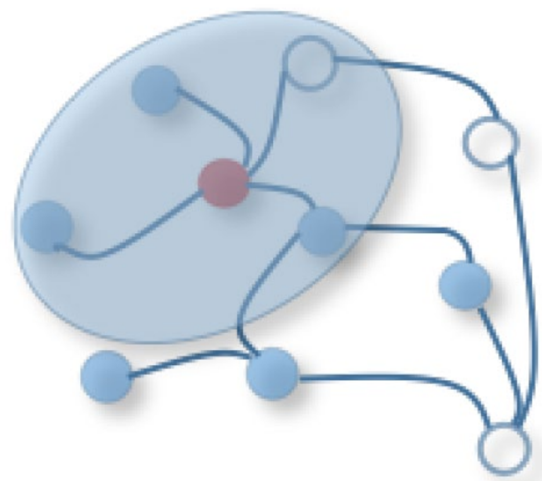
$$\mathbf{h}_i' = \sigma\left(\mathbf{W}_0\mathbf{h}_i + \sum_{j\in\mathcal{N}_i}\alpha_{ij}\mathbf{W}_1\mathbf{h}_j\right)$$

图神经网络

- 图卷积



2D卷积

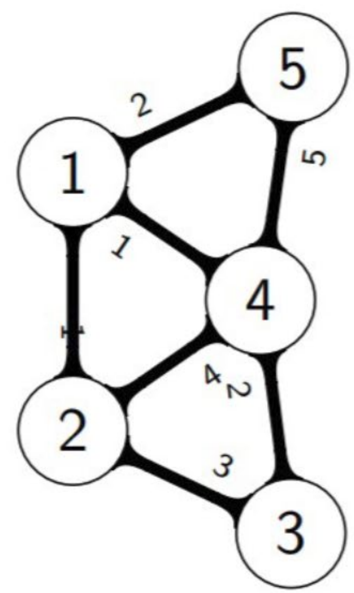


图卷积

Graph Laplacian Matrix

$\mathbf{A}$	adjacency matrix
$\mathbf{W}$	weight matrix
$\mathbf{D}$	(diagonal) degree matrix
$\mathbf{L} = \mathbf{D} - \mathbf{W}$	graph Laplacian matrix

$$\mathbf{L} = \begin{pmatrix} 4 & -1 & 0 & -1 & -2 \\ -1 & 8 & -3 & -4 & 0 \\ 0 & -3 & 5 & -2 & 0 \\ -1 & -4 & -2 & 12 & -5 \\ -2 & 0 & 0 & -5 & 7 \end{pmatrix}$$



$$\mathbf{D_{ii}} = \sum_j (\mathbf{A_{i,j}})$$
$$\mathbf{L} = \mathbf{I_n} - \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}$$

卷积 -> 图卷积

- 卷积
- **w*x** = IFFT(FFT(**w**) • FFT(**x**))

$$\begin{bmatrix} a & b & c & d \\ b & c & d & e \\ c & d & e & f \\ d & e & f & g \end{bmatrix}$$

周期信号
循环矩阵

- 图卷积
  - 奇异值分解 $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$

    $\mathscr{F}(\mathbf{x}) = \mathbf{U}^T \quad \mathscr{F}^{-1}(\hat{\mathbf{x}}) = \mathbf{U}\hat{\mathbf{x}}$
  - 滤波器 $\mathbf{g} \in \mathbf{R}^N \quad \mathbf{g}_\theta = diag(\mathbf{U}^T\mathbf{g})$
  - 图卷积 $\mathbf{x} *_G \mathbf{g}_\theta = \mathbf{U}\mathbf{g}_\theta\mathbf{U}^T\mathbf{x}$

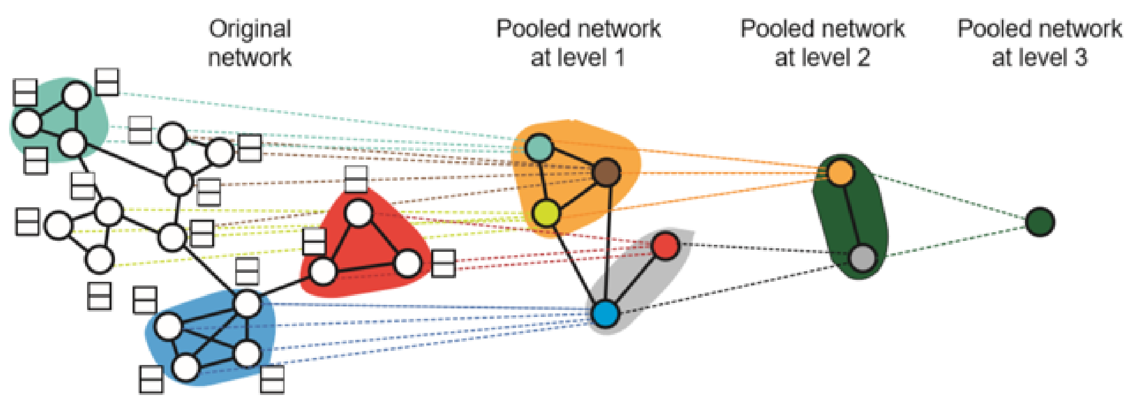
Graph Pooling

- Mean、Max、Sum

$$\mathbf{h}_G = mean/max/sum(\mathbf{h}_1^T, \mathbf{h}_2^T, ..., \mathbf{h}_n^T)$$

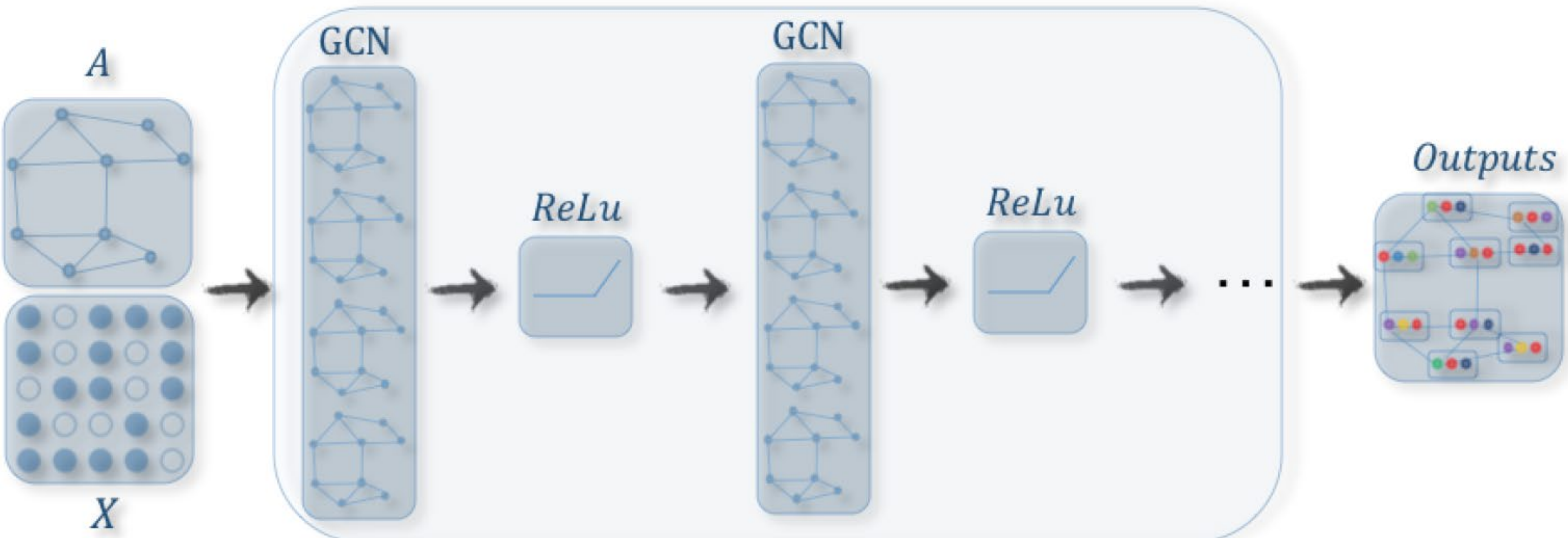
- SortPooling

- DIFFPOOL



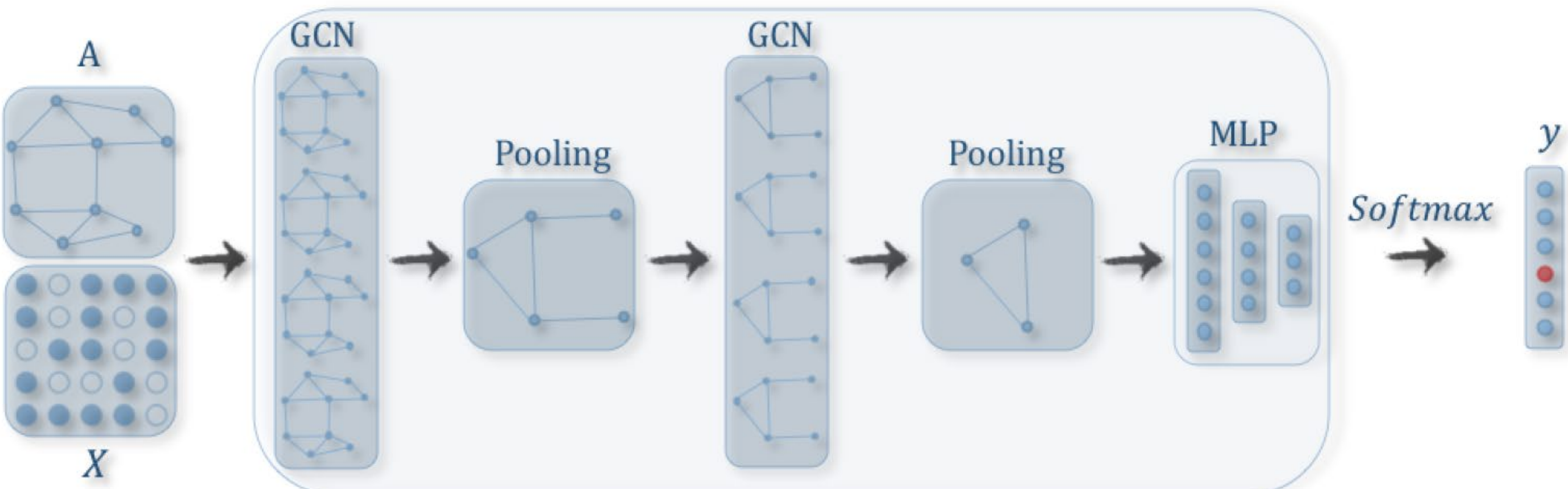
典型图卷积网络

- 节点分类



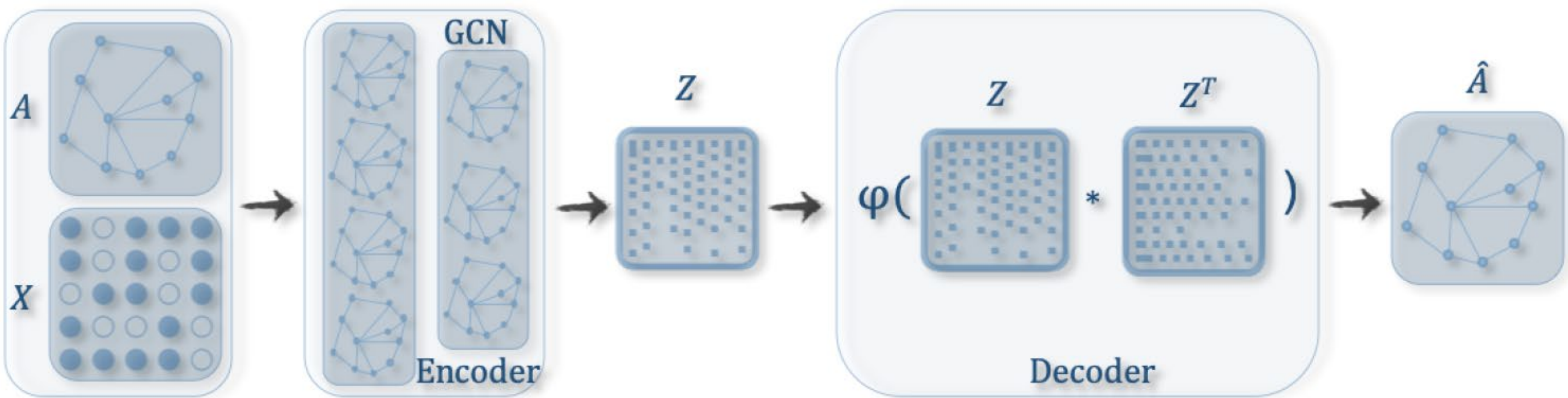
典型图卷积网络

- 图分类



典型图卷积网络

- 自编码网络



图卷积网络

- 其它进展
  - 图时空网络（Graph Spatial-Temporal Networks）
  - 图注意力机制
  - 图生成和对抗网络
  - …