NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

Dzmitry Bahdanau, KyungHyun Cho, Yoshua Bengio

讲解人：刘逸博

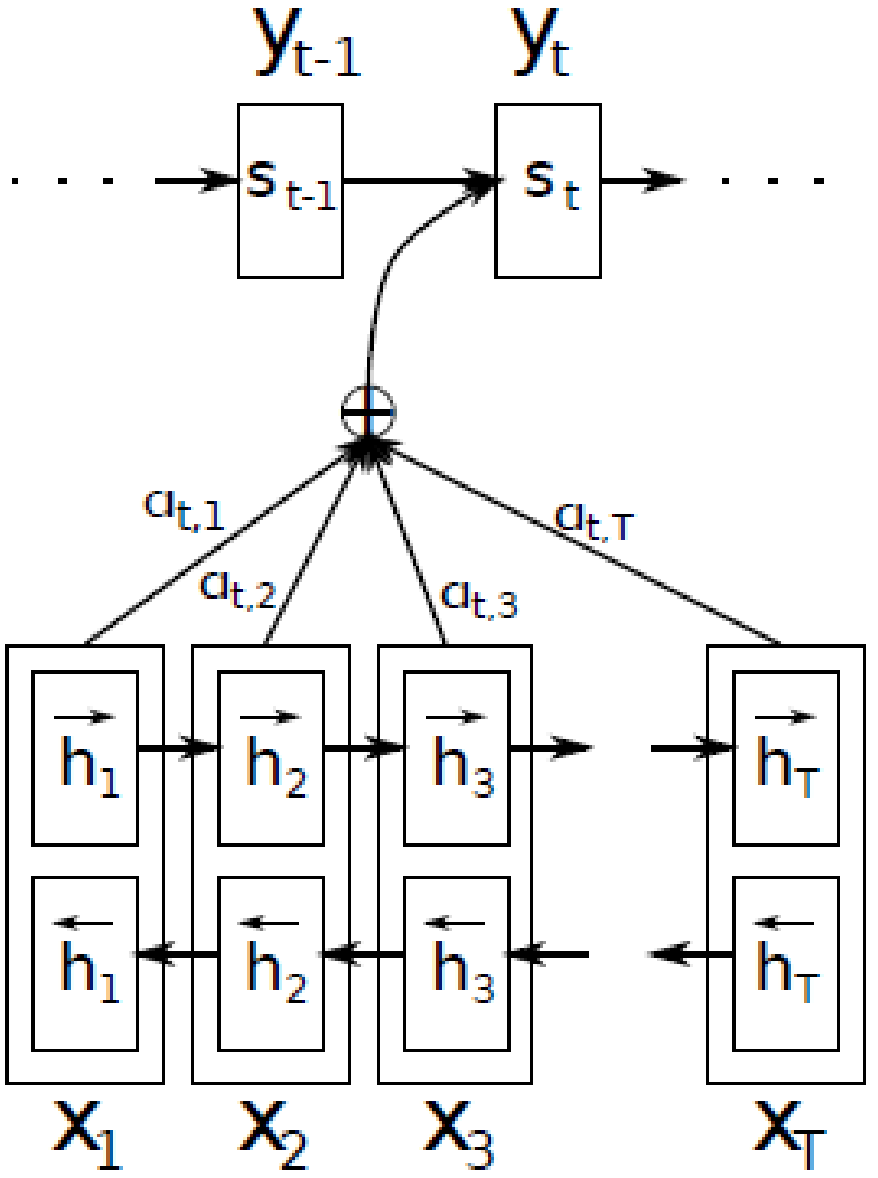
2018/7/22

- 模型结构
- 训练方法
- 实验结果

模型结构

整体结构

- 带attention机制的seq2seq架构
- Seq2seq: 输入序列映射为不等长的输出序列
- Encoder： 输入序列->c
- Decoder： c->输出序列



模型优点

- C长度可变：每个时刻而都不同
- Attention机制：c和序列输出相关联

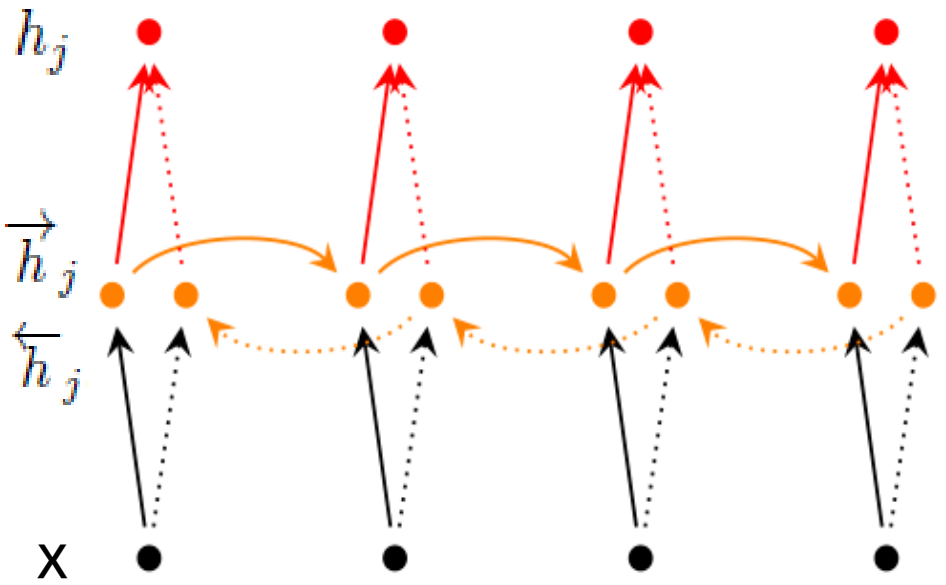
Encoder结构

- BiRNN

- 好处：单向RNN只能考虑t时刻之前的序列，而双向RNN考虑过去和未来的输入序列

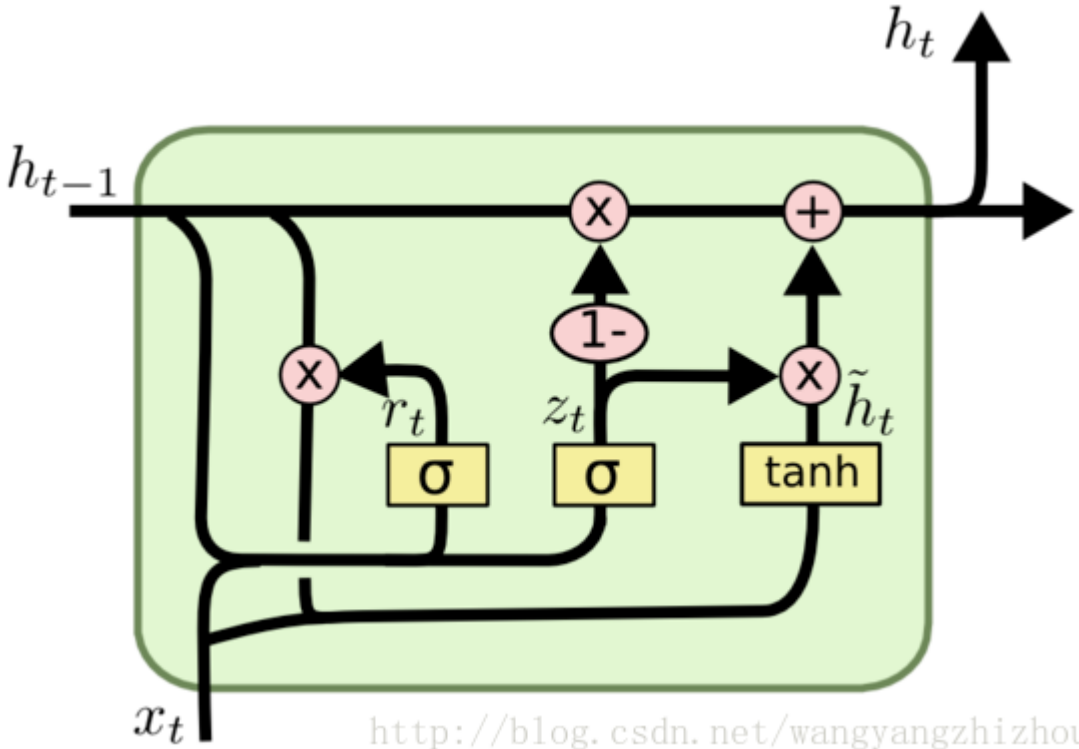
$$h_i = \begin{bmatrix} \overrightarrow{h}_i \\ \overleftarrow{h}_i \end{bmatrix}$$

- 维度2n



- GRU：门控循环单元

- 更新门z：控制前一时刻的状态信息被带入到当前状态中的程度。值越大带入越多。
- 复位门r：控制忽略前一时刻的状态信息的程度。值越小忽略得越多。



X->h

$$\overrightarrow{h}_i = \begin{cases} (1 - \overrightarrow{z}_i) \circ \overrightarrow{h}_{i-1} + \overrightarrow{z}_i \circ \underline{\overrightarrow{h}}_i & , \text{if } i > 0 \\ 0 & , \text{if } i = 0 \end{cases}$$

- 输入： $\mathbf{x} = (x_1, \ldots, x_{T_x}),\ x_i \in \mathbb{R}^{K_x}$
- 输出： $\mathbf{y} = (y_1, \ldots, y_{T_y}),\ y_i \in \mathbb{R}^{K_y}$
- Kx, Ky： 词表大小
- Tx, Ty： 句子长度

$$\underline{\overrightarrow{h}}_i = \tanh\left(\overrightarrow{W}\overrightarrow{E}x_i + \overrightarrow{U}\left[\overrightarrow{r}_i \circ \overrightarrow{h}_{i-1}\right]\right)$$

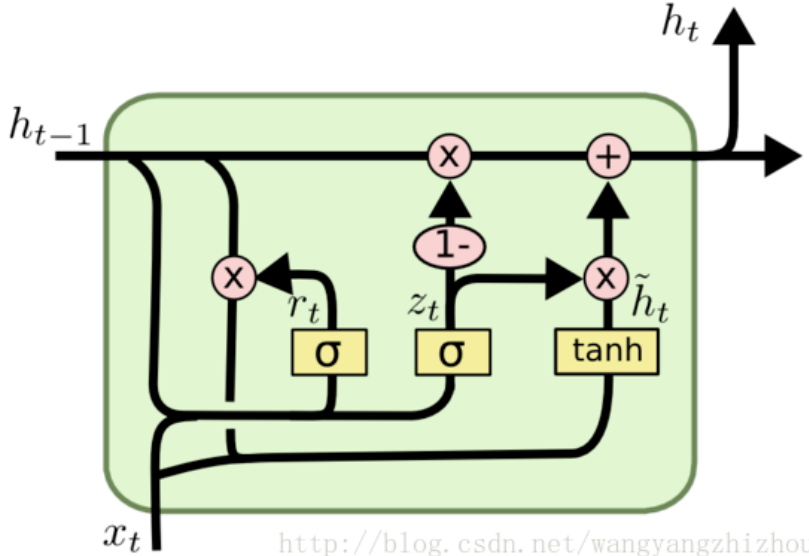
$$\overrightarrow{z}_i = \sigma\left(\overrightarrow{W}_z\overrightarrow{E}x_i + \overrightarrow{U}_z\overrightarrow{h}_{i-1}\right)$$

$$\overrightarrow{r}_i = \sigma\left(\overrightarrow{W}_r\overrightarrow{E}x_i + \overrightarrow{U}_r\overrightarrow{h}_{i-1}\right).$$

$$h_i = \begin{bmatrix} \overrightarrow{h}_i \\ \overleftarrow{h}_i \end{bmatrix}$$

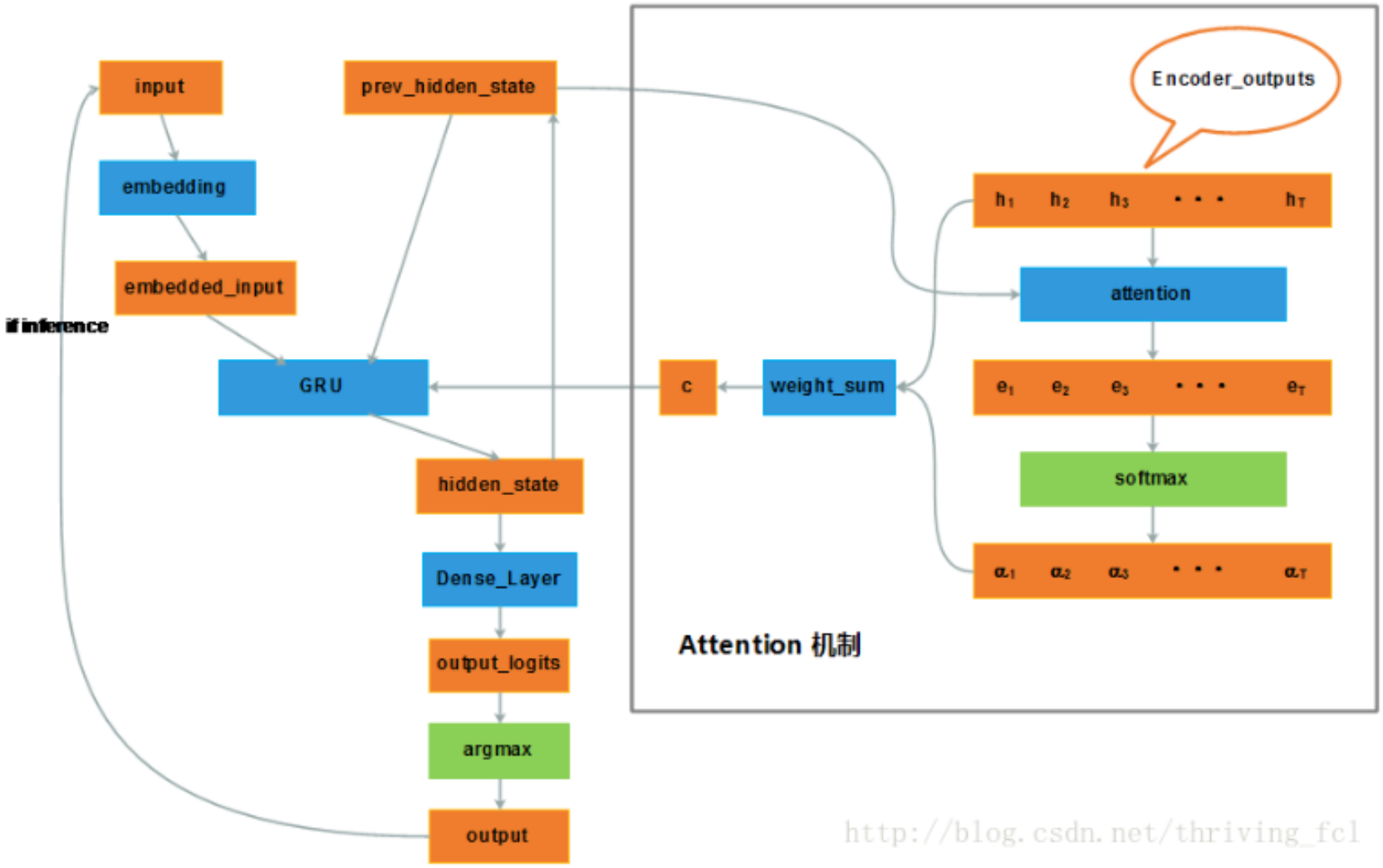
- 词嵌入矩阵 E: m*Kx维，双向共享
- 权重 W, U: n*m维，不共享

- m: 词嵌入维数， n: 隐层单元数



Decoder: Attention机制

- 标号变化：i指时刻，j指输入序列



- 1. 离散的词ID转换为词向量

- **2. 由encoder的输出结合decoder的prev_hidden_state生成energy**

$$e_{ij} = v_a^\top \tanh\left(W_a s_{i-1} + U_a h_j\right)$$

- Energy的含义：源语言输入词xj对目标词yi的影响力。
- Uahj可以提前算好。

- **3. 由energy 到概率（即归一化）**

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

- 如果直接把能量e作为权值，可能会将context向量缩放若干倍。需要归一化。Softmax。

- **4. context 向量合成（加权平均）**

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

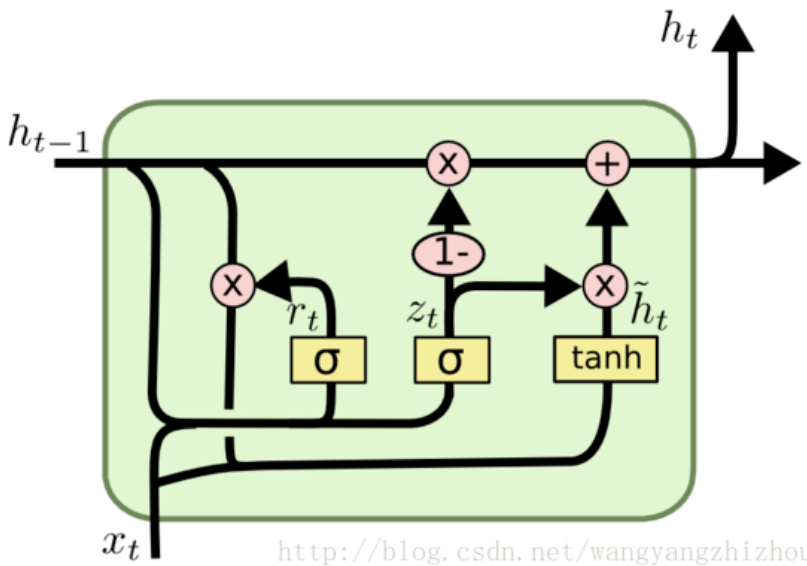
Attention机制的优点

- 减小处理高维输入数据的计算负担，通过结构化的选取输入的子集，降低数据维度。

- “去伪存真”，让任务处理系统更专注于找到输入数据中显著的与当前输出相关的有用信息，从而提高输出的质量。

- **5. prev_hidden_state, 词向量, context向量通过GRU单元生成下一时刻hidden_state**

- 只能单向，因为未来是未知的。



$$s_i = (1 - z_i) \circ s_{i-1} + z_i \circ \tilde{s}_i,$$

$$\tilde{s}_i = \tanh\left(WEy_{i-1} + U\left[r_i \circ s_{i-1}\right] + Cc_i\right)$$
$$z_i = \sigma\left(W_z Ey_{i-1} + U_z s_{i-1} + C_z c_i\right)$$
$$r_i = \sigma\left(W_r Ey_{i-1} + U_r s_{i-1} + C_r c_i\right)$$

- **6. 使用全连接层将hidden_state映射为vocabulary size的向量**
- 生成的hidden_state已经包含了待生成的词的信息了，但是要生成具体的词，我们还需要知道目标语言中每个词的条件概率p(yi|si)，如果si的维度就是目标语言的词典大小，那么使用softmax就可以算出每个词的概率，但是si的维度也属于模型的一个参数，通常是不会等于目标语言词典的大小的，因此再增加一个全连接层，将si映射为维度等于词典大小L的向量vi，每个维度代表一个词，使用softmax计算出每个词的概率。

$$p(y_i|s_i, y_{i-1}, c_i) \propto \exp\left(y_i^\top W_o t_i\right),$$

$$t_i = \left[\max\left\{\tilde{t}_{i,2j-1}, \tilde{t}_{i,2j}\right\}\right]_{j=1,\ldots,l}^\top$$

$$\tilde{t}_i = U_o s_{i-1} + V_o E y_{i-1} + C_o c_i.$$

测试

- 选择最大概率的输出：beam search算法
- 例如: 词表大小3: a,b,c, beam size = 2
- 解码时，生成第一个词时，选择概率最大2个词a,c,现在的序列是a和c。（两个，beam size）
- 选择第二个词时，将ac与此表里所有词组合，再选择最大概率的两个作为当前序列，例如aa和cb。
- 循环直到遇到结束符。

实验结果

数据信息

- 数据集：europarl-v8.lv-en
- 数据处理：标点符号用空格隔开，大写->小。
- 数据集划分：随机抽取。
- 语种：Latvian-English
- 训练集句对数：632789
- 训练集平均句子长度：27.1
- 训练集词表大小：30000
- 测试集句对数：3000
- 测试集平均句子长度：27.2
- 测试集词表大小：30000
- 测试集和训练集重合度：0

训练结果

- 训练参数:
- hidden_units=200, hidden_edim=200 , num_layer=1, vocab_size=30000, batch_size=10, beam_size=1

- 训练结果:
- checkpoint=102000, bleu=28.1。翻译结果见文档。