法律声明

- 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

- 课程详情请咨询
  - 微信公众号：小象
  - 新浪微博：ChinaHadoop

神经序列模型 V

主讲人： 史兴

07/26/2017

提纲

- 勘误：RNNLM实现细节
- Seq2Seq 实现细节
- Attention 实现细节
- 勘误：LSTM 与 beam_search
- Beam Search 实现细节

勘误：RNNLM实现细节

- 多层lstm的实现
- 错误的代码：
  - seqModel.py:108
  - single_cell = tf.contrib.rnn.MultiRNNCell([single_cell] * num_layers, state_is_tuple=True)
  - 导致每层的参数共享

勘误：RNNLM实现细节

- 多层lstm的实现
- 正确的代码：
  - seqModel.py:103-116
  - single_cell = tf.contrib.rnn.MultiRNNCell([lstm_cell() for _ in xrange(num_layers)], state_is_tuple=True)

Seq2Seq 代码实现

- 地址：
  https://github.com/shixing/xing_nlp/tree/master/Seq2Seq
- 文件夹说明

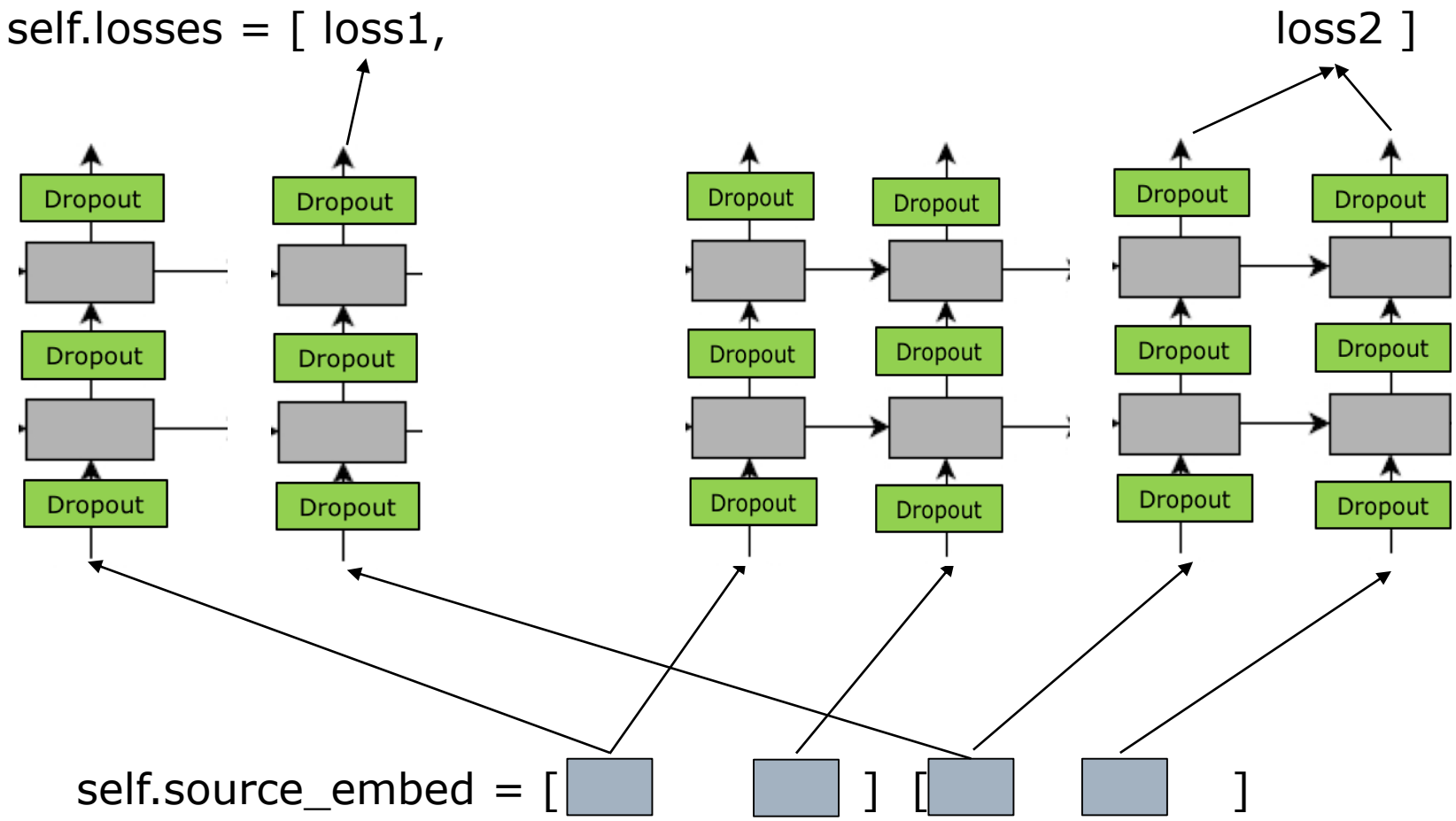
Seq2Seq 代码实现

- run.py
  - _buckets
  - def train() #训练
  - def read_data() #数据填入buckets
    - 输入语句的反向
    - 在输出语句末尾添加EOS
  - def beam_decode() #beam search
  - def read_data_test() #测试数据读取

Seq2Seq 代码实现

- seqModel.py
  - def __init__() # 初始化
    - 注意多层LSTM的实现

Seq2Seq 代码实现



Seq2Seq 代码实现

- seqModel.py
  - def get_batch() #增加padding
    - source在前面加padding, target在后面加padding
    - a b c -> 1 2 3 _EOS
    - _PAD _PAD a b c -> 1 2 3 _EOS _PAD
  - def basic_seq2seq() # 连接encoder decoder
    - 注意scope问题

Seq2Seq 代码实现

- 运行代码
  - /sh: bash train_small.sh
  - 字符串复制问题
    - a, b, c → a, b, c
  - 理想的PPT应该是多少？

Attention 代码实现

Attention

$$\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) = \begin{cases} \boldsymbol{h}_t^\top \bar{\boldsymbol{h}}_s & \textit{dot} \\ \boldsymbol{h}_t^\top \boldsymbol{W}_a \bar{\boldsymbol{h}}_s & \textit{general} \\ \boldsymbol{v}_a^\top \tanh\left(\boldsymbol{W}_a[\boldsymbol{h}_t; \bar{\boldsymbol{h}}_s]\right) & \textit{concat} \end{cases}$$

$$\begin{aligned} \boldsymbol{a}_t(s) &= \text{align}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) \\ &= \frac{\exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)\right)}{\sum_{s'} \exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_{s'})\right)} \end{aligned}$$

$$c_t = \sum_{s'} a_t(s')\, \overline{h_{s'}}$$

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_c[\boldsymbol{c}_t; \boldsymbol{h}_t])$$

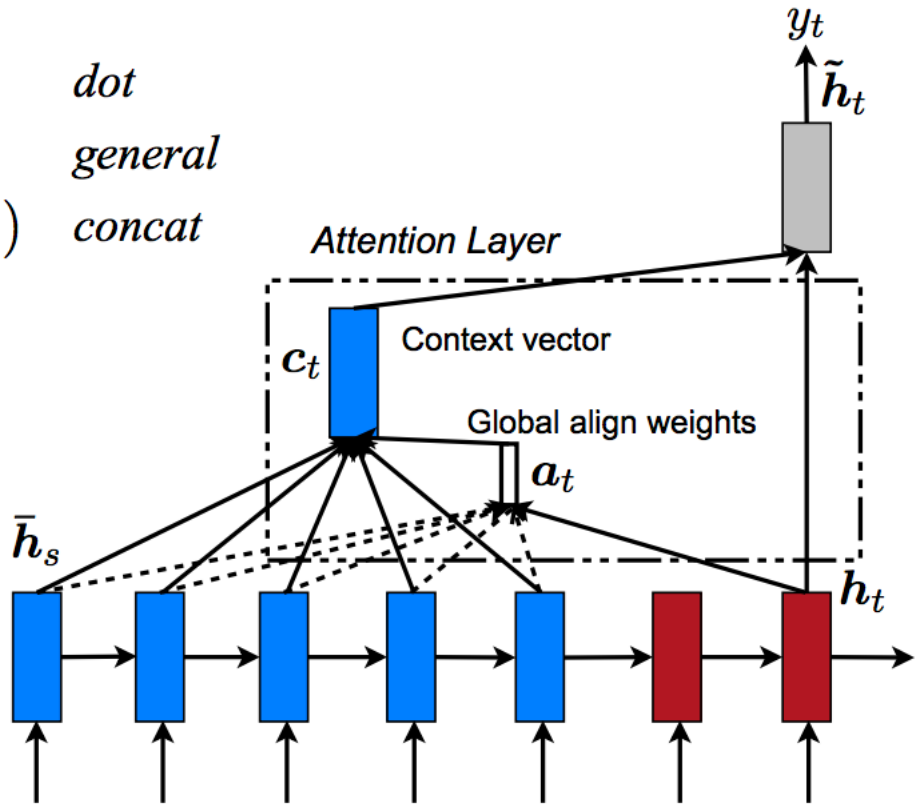


Figure from https://arxiv.org/pdf/1508.04025.pdf

Attention

- Attention
  - feed-input：下一个单词知道上一个单词的attention

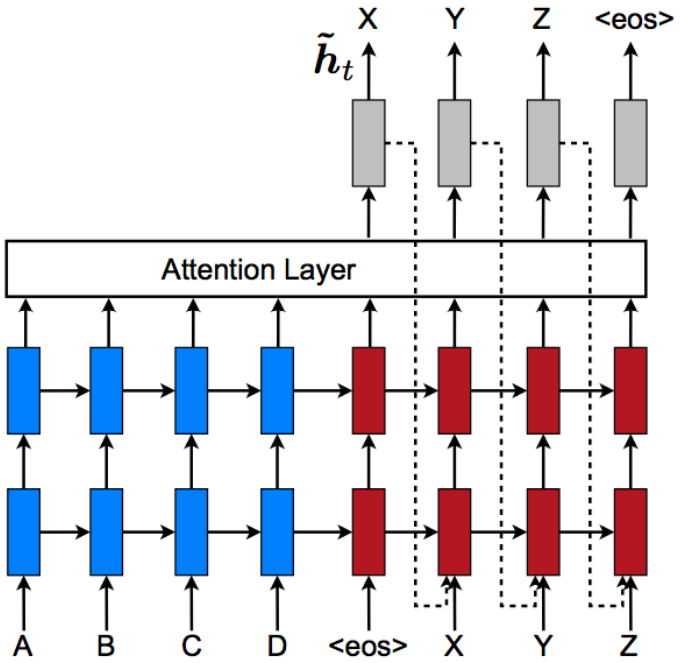


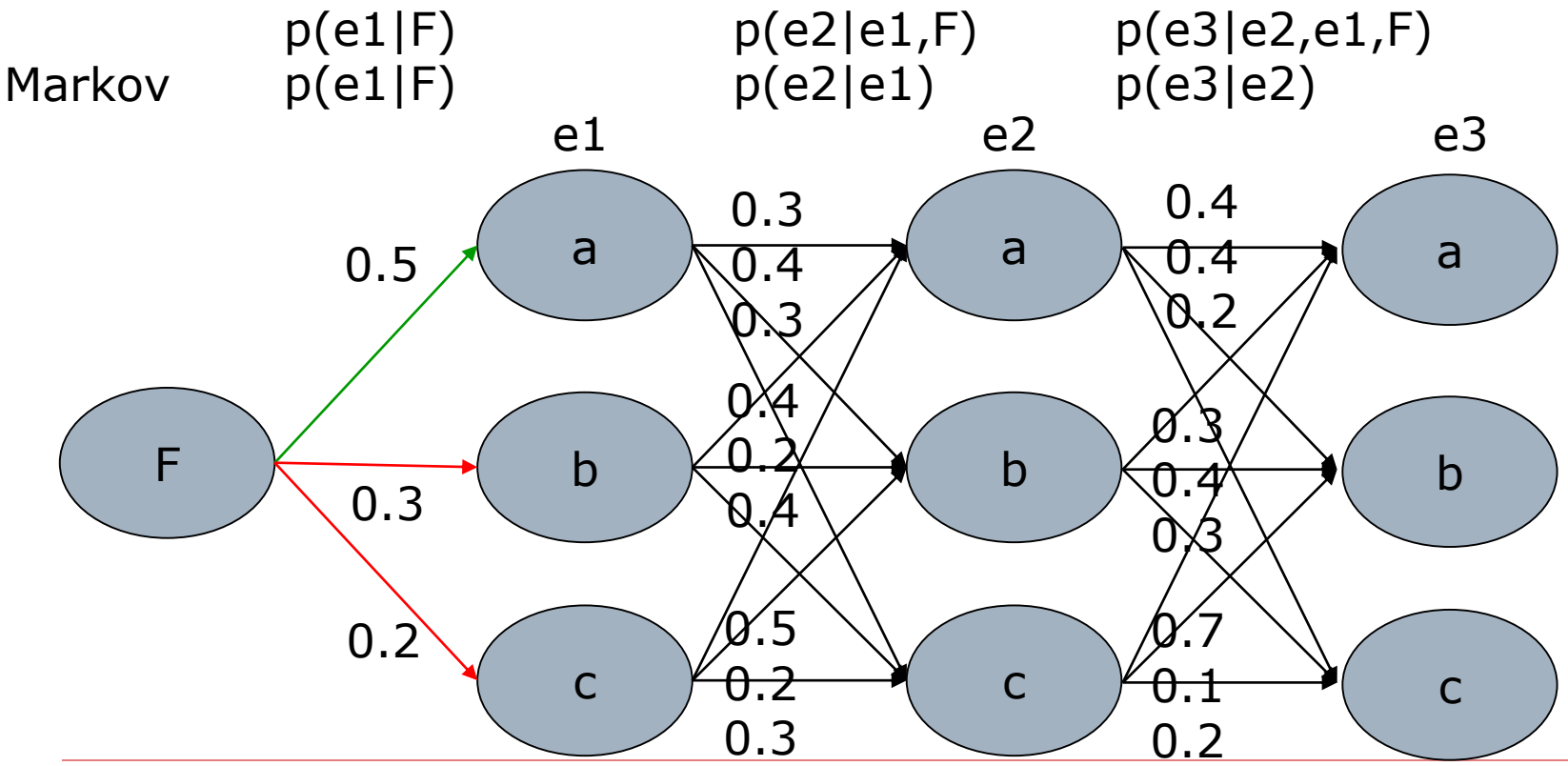
Figure from https://arxiv.org/pdf/1508.04025.pdf

Attention 代码实现

- seqModel.py
  - def attention_seq2seq()
    - conv2d

勘误：LSTM 与 beam_search

- 下图表示实际是带有markov假设的



勘误：LSTM 与 beam_search

- LSTM并没有markov假设

F	p(e1|F)	e1	p(e2|e1,F)	e2	p(e3|e2,e1,F)	e3
F	0.3	a	0.8	a	0.3	a
					0.7	b
			0.2	b	0.8	a
					0.2	b
	0.7	a	0.5	a	0.3	a
					0.7	b
			0.5	b	0.5	a
					0.5	b

勘误：LSTM 与 beam_search

- 最优解：穷举法 $O(|V|^N)$

F	p(e1|F)	e1	p(e2|e1,F)	e2	p(e3|e2,e1,F)	e3
F	0.3	a	0.8	a	0.3	a
					0.7	b
			0.2	b	0.8	a
					0.2	b
	0.7	a	0.5	a	0.3	a
					0.7	b
			0.5	b	0.5	a
					0.5	b

勘误：LSTM 与 beam_search

- Beam Search (beam_size = 2)

F	p(e1|F)	e1	p(e2|e1,F)	e2	p(e3|e2,e1,F)	e3
F	0.3	a	0.8	a	0.3	a
					0.7	b
			0.2	b	0.8	a
					0.2	b
	0.7	b	0.5	a	0.3	a
					0.7	b
			0.5	b	0.5	a
					0.5	b

b : 0.7			
a : 0.3			

勘误：LSTM 与 beam_search

- Beam Search (beam_size = 2)

F	p(e1|F)	e1	p(e2|e1,F)	e2	p(e3|e2,e1,F)	e3
F	0.3	a	0.8	a	0.3	a
					0.7	b
			0.2	b	0.8	a
					0.2	b
	0.7	b	0.5	a	0.3	a
					0.7	b
			0.5	b	0.5	a
					0.5	b

b : 0.7	ba: 0.35		
a : 0.3	bb: 0.35		

勘误：LSTM 与 beam_search

- Beam Search (beam_size = 2)

F	p(e1|F)	e1	p(e2|e1,F)	e2	p(e3|e2,e1,F)	e3
F	0.3	a	0.8	a	0.3	a
					0.7	b
			0.2	b	0.8	a
					0.2	b
	0.7	b	0.5	a	0.3	a
					0.7	b
			0.5	b	0.5	a
					0.5	b

b : 0.7	ba: 0.35	bab: 0.245	
a : 0.3	bb: 0.35	bba: 0.175	

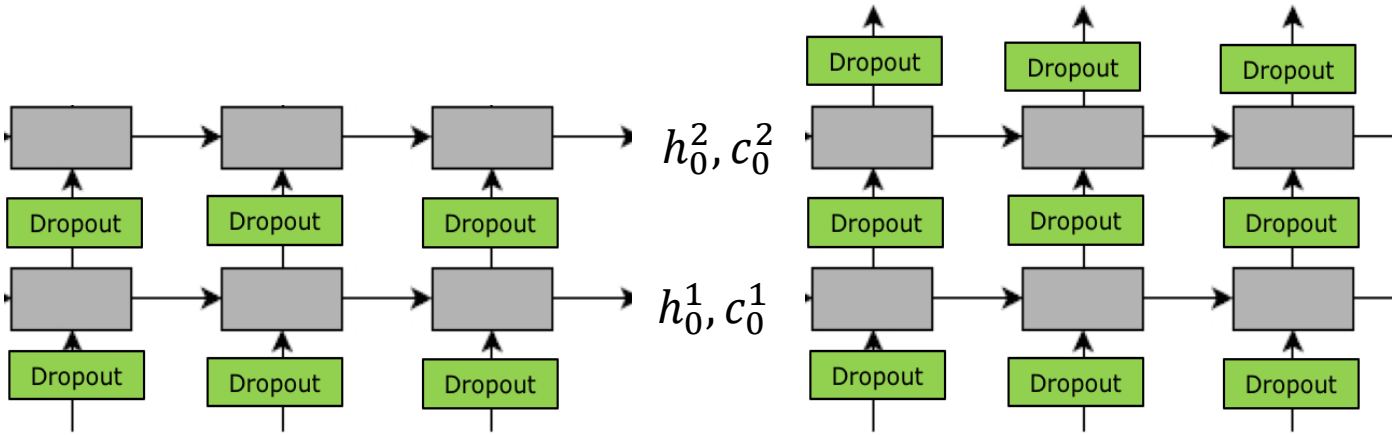
勘误：LSTM 与 beam_search

潜在Bug#1

a:0.3 b:0.7
a:0.3 b:0.7

错误：

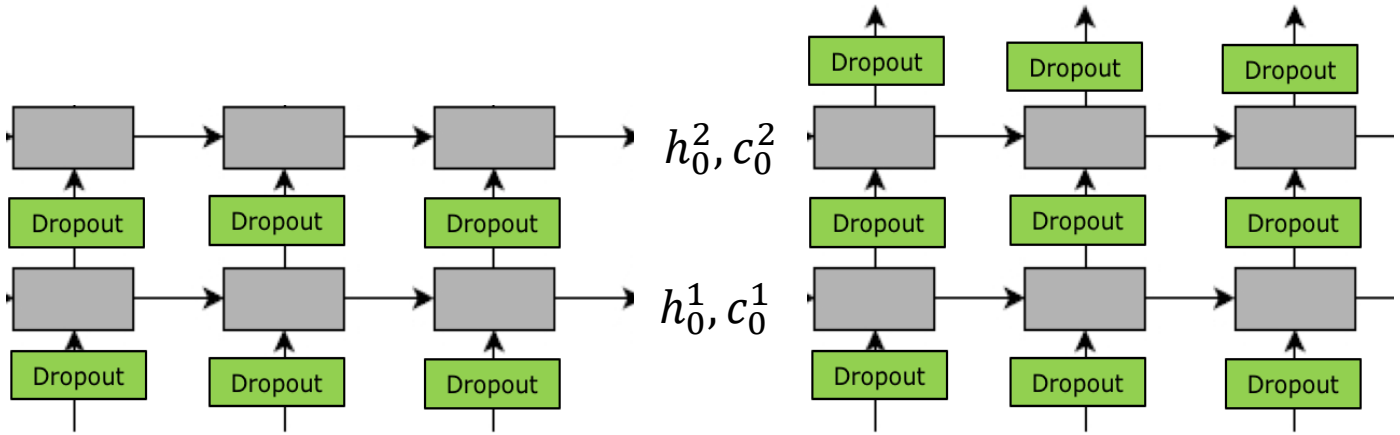
a:0.3 b:0.7
a:0.3 b:0.7



h_0^1 : [0.2,-0.3] [0.2,-0.3] =》 h_1^1 : [1.0,-3.3] [1.0,-3.3]

勘误：LSTM 与 beam_search

a:0.5 b:0.5
a:0.8 b:0.2



f1 f2 f3

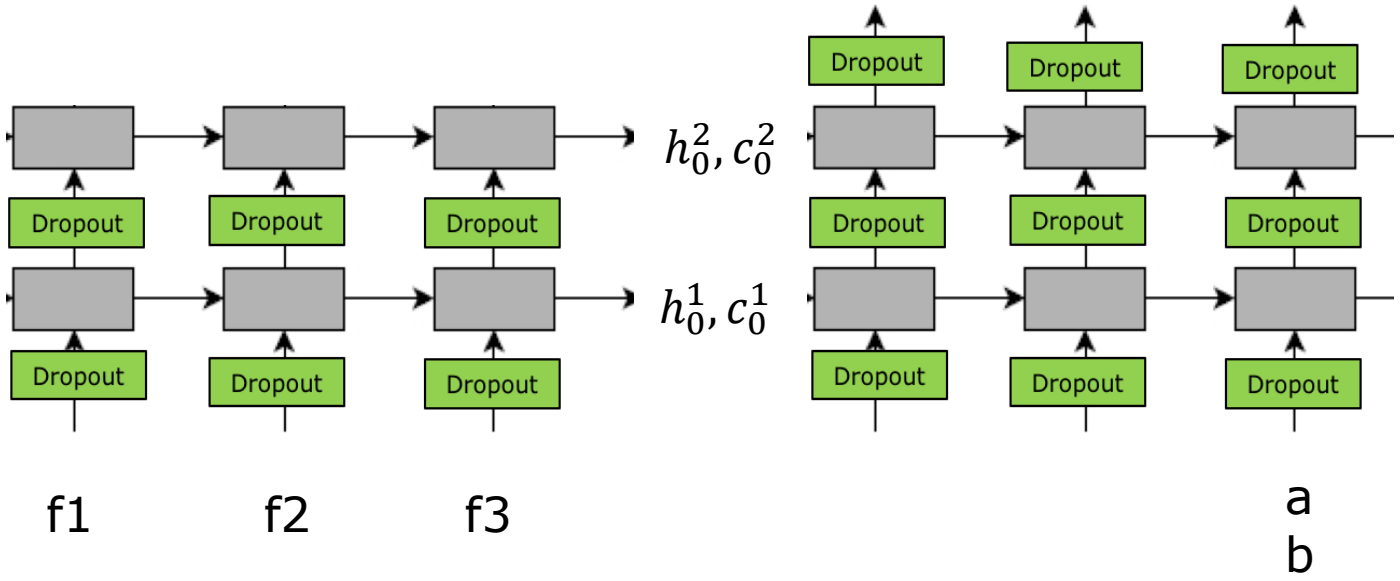
b
a

h_1^1 : [1.0,-3.3] [1.0,-3.3] =》 h_2^1 : [-1.3,-0.3] [1.1,-1.5] =》 [-1.3,-0.3] [-1.3,-0.3]

潜在Bug#2

勘误：LSTM 与 beam_search

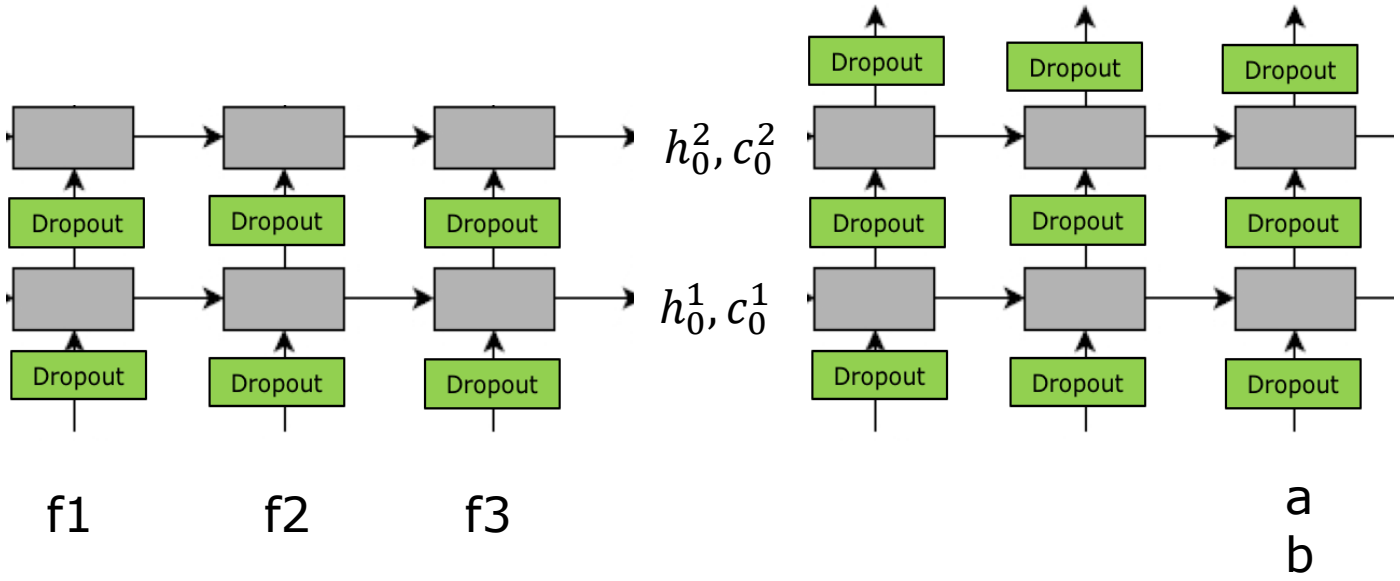
a:0.3 b:0.7
a:0.5 b:0.5



h_2^1 : [-1.3,-0.3] [-1.3,-0.3] =》 h_3^1 : [-1.5,-2.3] [1.8,-2.1]

勘误：LSTM 与 beam_search

a:0.3 b:0.7
a:0.5 b:0.5



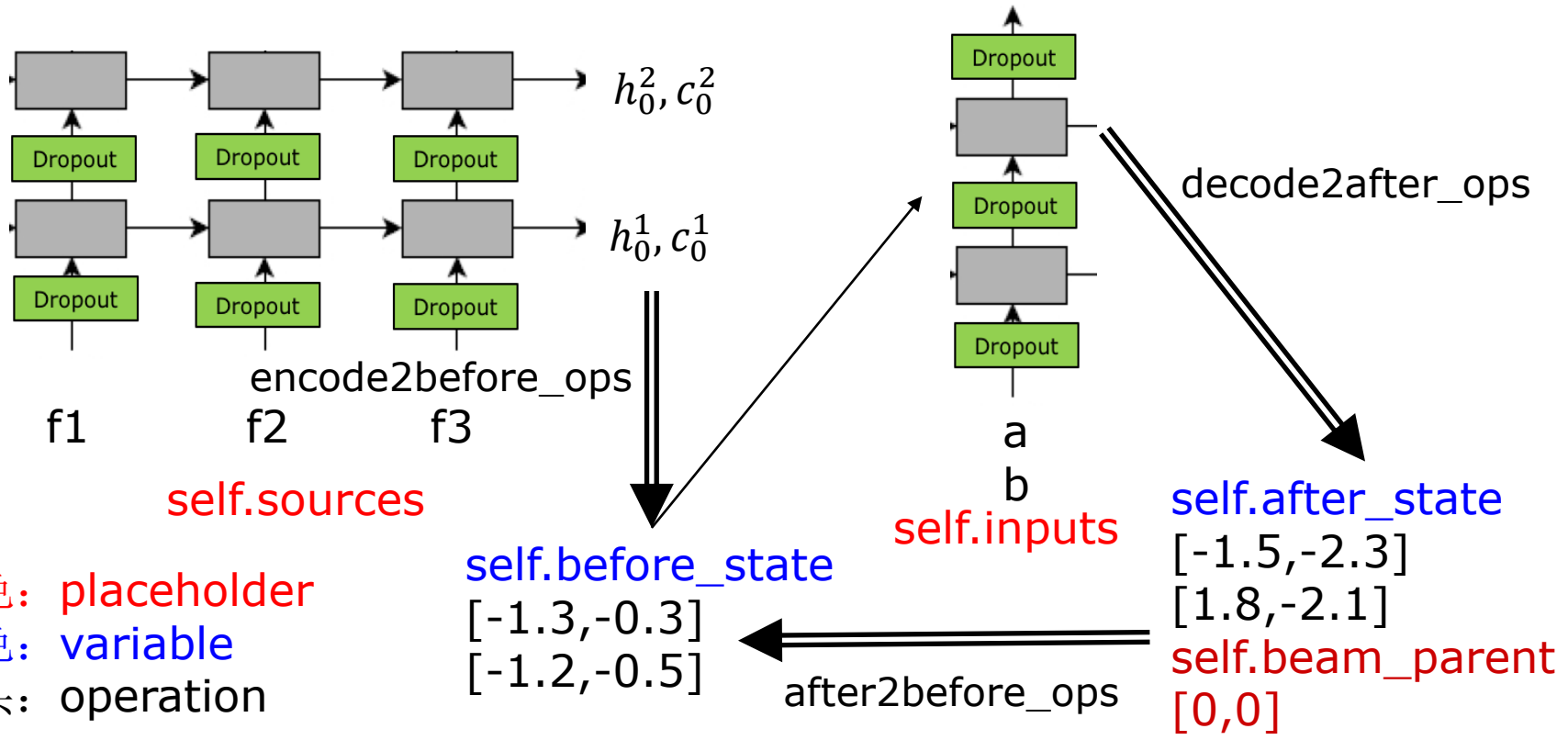
h_2^1 : [-1.3,-0.3] [-1.3,-0.3] =》 h_3^1 : [-1.5,-2.3] [1.8,-2.1]

Beam Search 代码实现

Single-step Decoder

self.top_index
self.top_value
self.eos_value

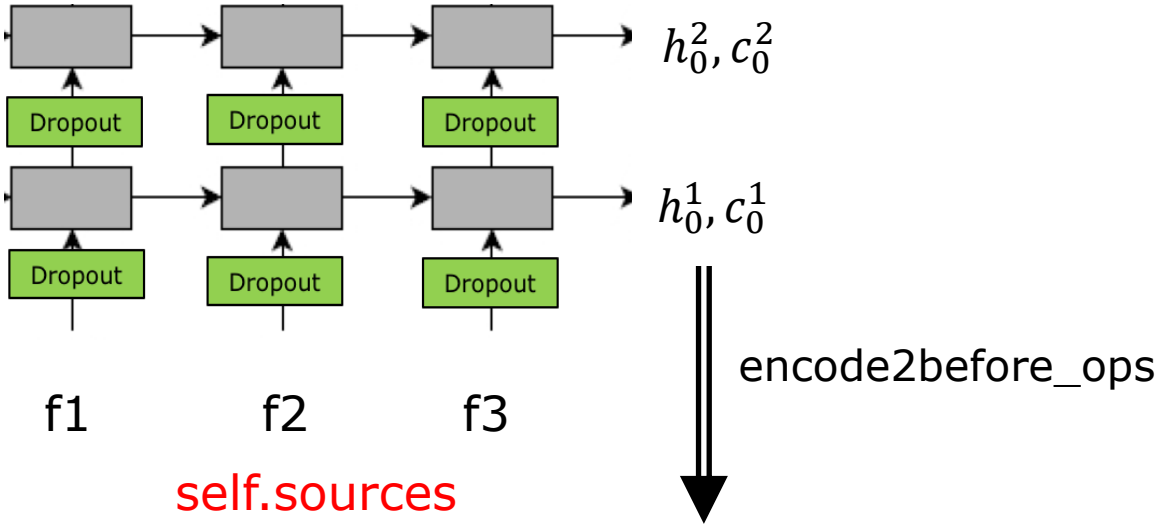
a:0.3 b:0.7
a:0.5 b:0.5



红色：placeholder
蓝色：variable
箭头：operation

Beam Search 代码实现

Single-step Decoder



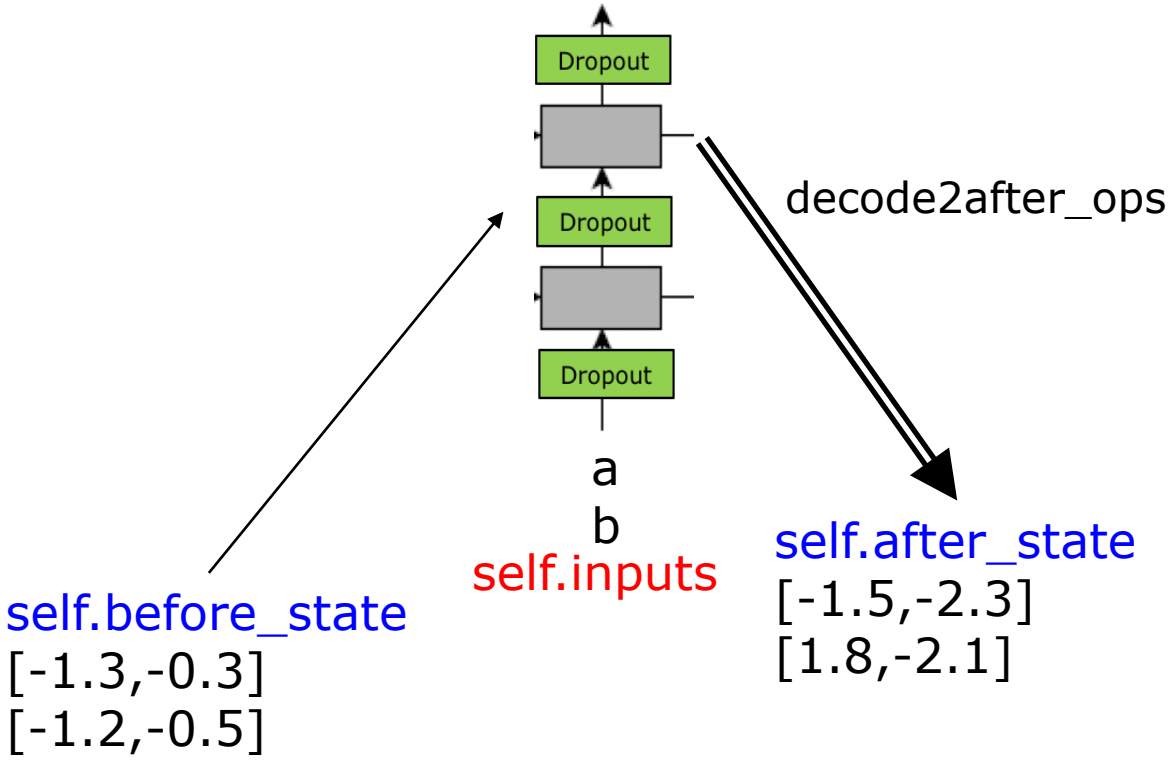
红色：placeholder
蓝色：variable
箭头：operation

Beam Search 代码实现

Single-step Decoder

self.top_index
self.top_value
self.eos_value

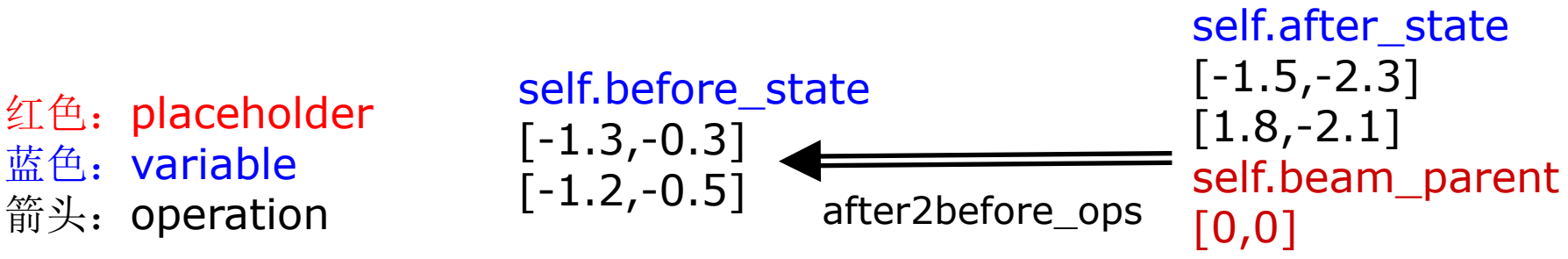
a:0.3 b:0.7
a:0.5 b:0.5



红色：placeholder
蓝色：variable
箭头：operation

Beam Search 代码实现

Single-step Decoder

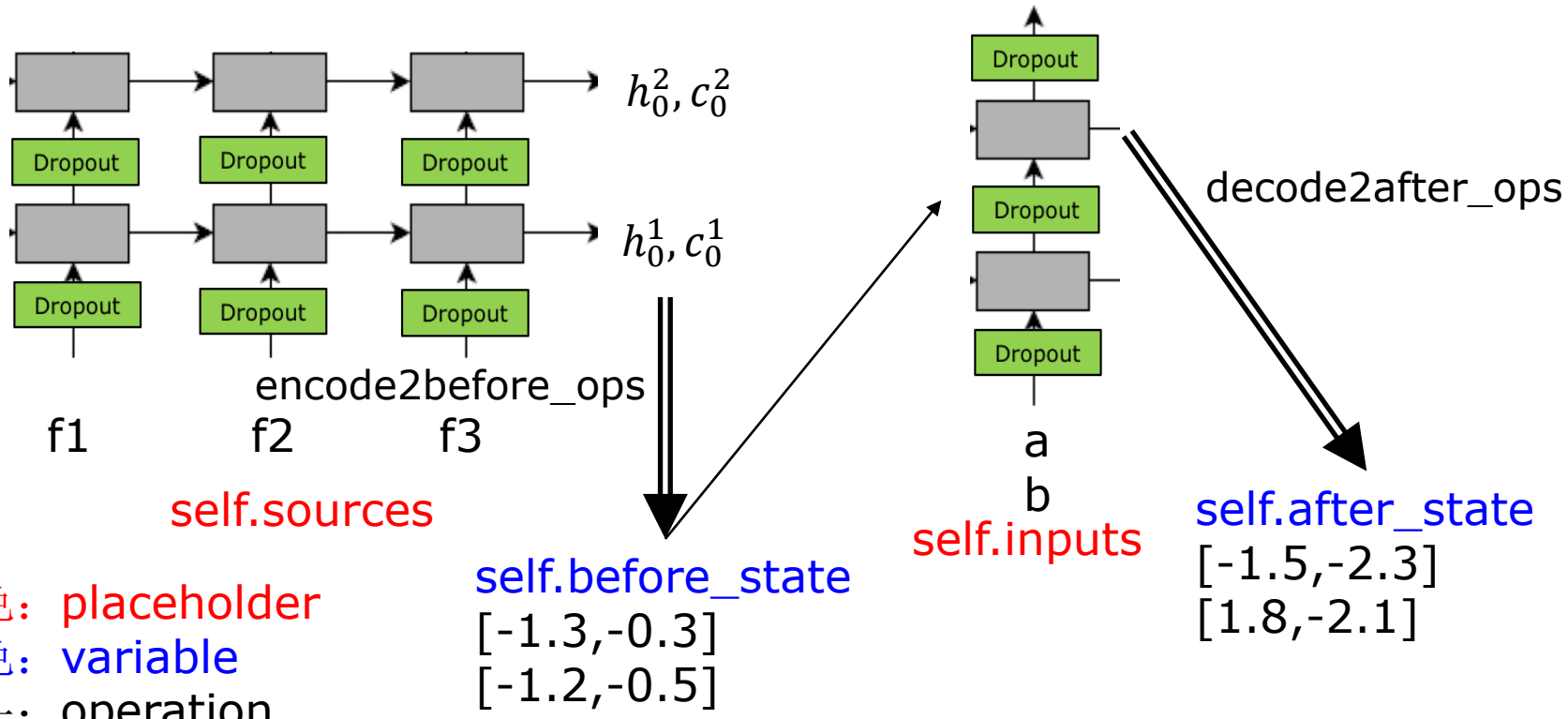


Beam Search 代码实现

beam_step(index=0)

self.top_index
self.top_value
self.eos_value

a:0.3 b:0.7
a:0.5 b:0.5



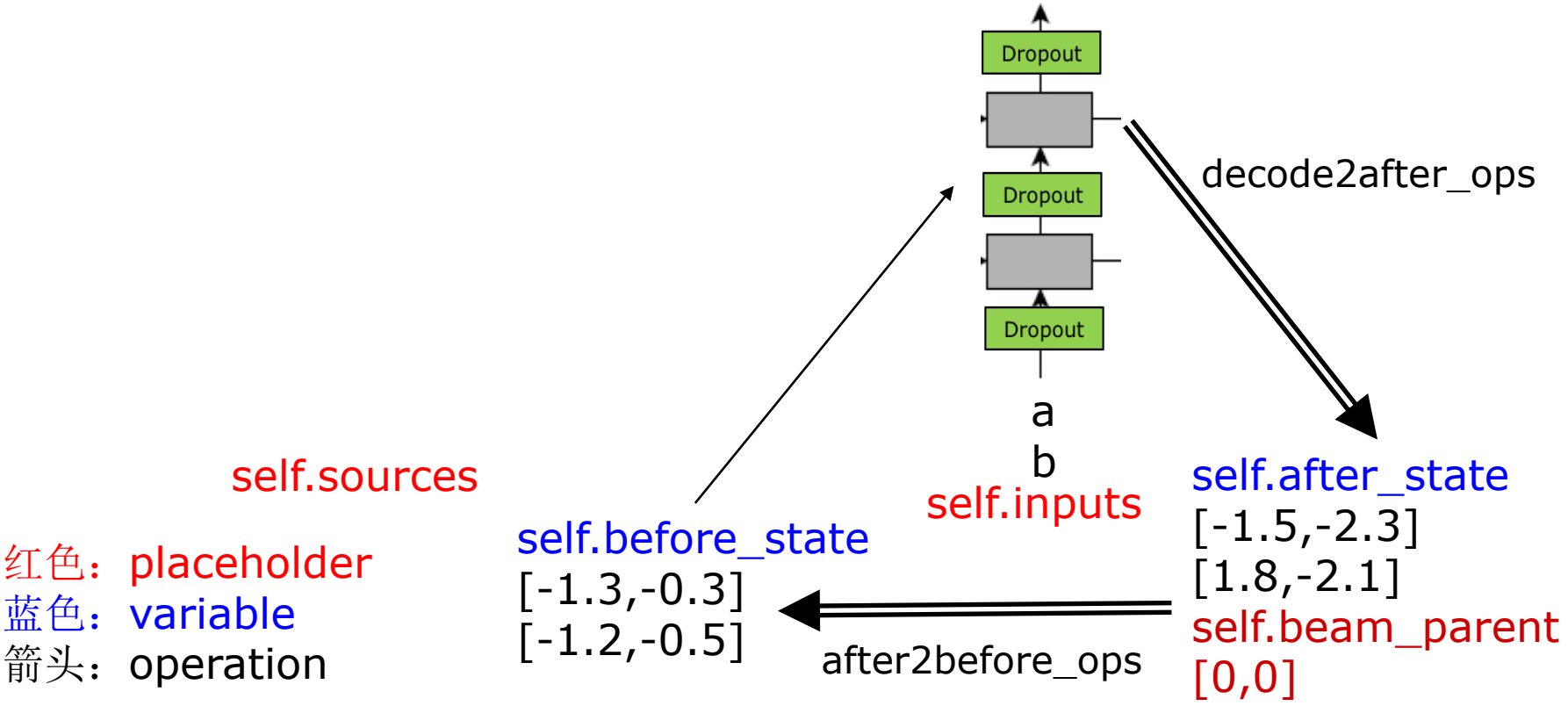
红色：placeholder
蓝色：variable
箭头：operation

Beam Search 代码实现

beam_step(index>0)

self.top_index
self.top_value
self.eos_value

a:0.3 b:0.7
a:0.5 b:0.5



Beam Search 代码实现

- run.py
  - beam_decode()
    - 潜在bug#1
    - EOS
      - 当生成EOS的时候，就加入候选句子中
      - 最后一步时，直接强制输出beam中所有的句子，需要查询EOS的值
      - 最长最短的控制(max_ratio, min_ratio)

Beam Search 代码实现

- BLEU score
  - 评价机器翻译的标准

$$\text{BLEU} = \min\left(1, \frac{output\text{-}length}{reference\text{-}length}\right) \left(\prod_{i=1}^{4} precision_i\right)^{\frac{1}{4}}$$

Beam Search 代码实现

BLEU score

SYSTEM A: Israeli officials responsibility of airport safety
2-GRAM MATCH 1-GRAM MATCH

REFERENCE: Israeli officials are responsible for airport security

SYSTEM B: airport security Israeli officials are responsible
2-GRAM MATCH 4-GRAM MATCH

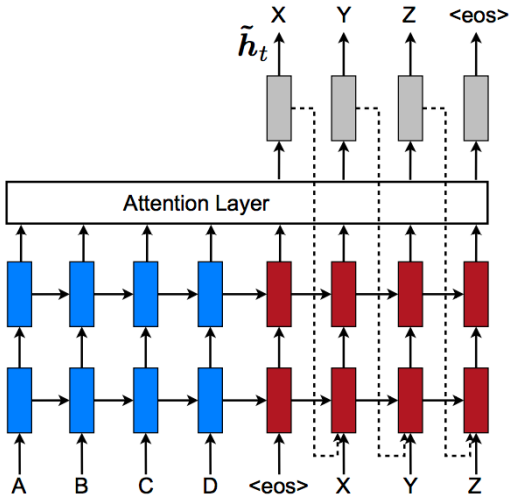
Metric	System A	System B
precision (1gram)	3/6	6/6
precision (2gram)	1/5	4/5
precision (3gram)	0/4	2/4
precision (4gram)	0/3	1/3
brevity penalty	6/7	6/7
BLEU	0%	52%

Beam Search 代码实现

- BLEU score
  - bash beam_decode_small.sh
  - bash bleu_small.sh

Beam Search 代码实现

- 高难度，高价值的作业
  - 实现attention model的beam search
    - def beam_attention_seq2seq()
    - feed_input是否需要加before_ht_att和after_ht_att



联系我们

小象学院：互联网新技术在线教育领航者

- 微信公众号：大数据分析挖掘
- 新浪微博：ChinaHadoop