第5课 基于深度学习的chatbot

寒小阳

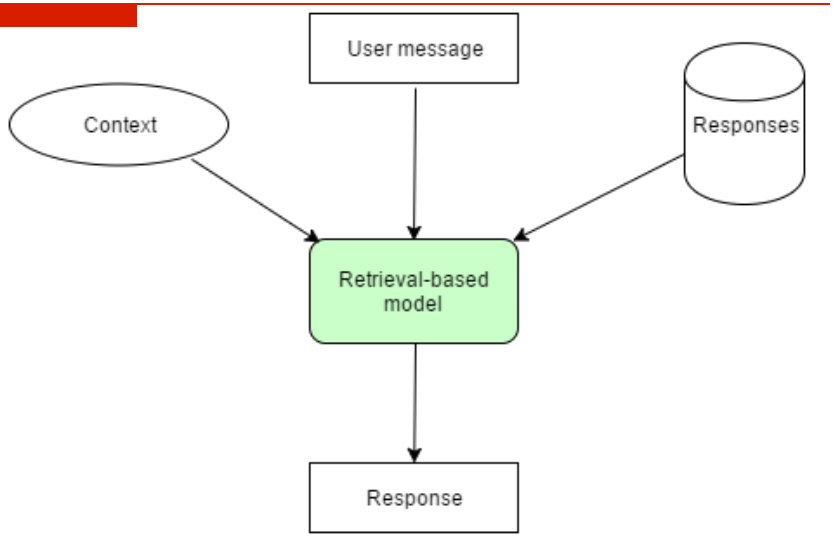
2017.04.22

主要内容

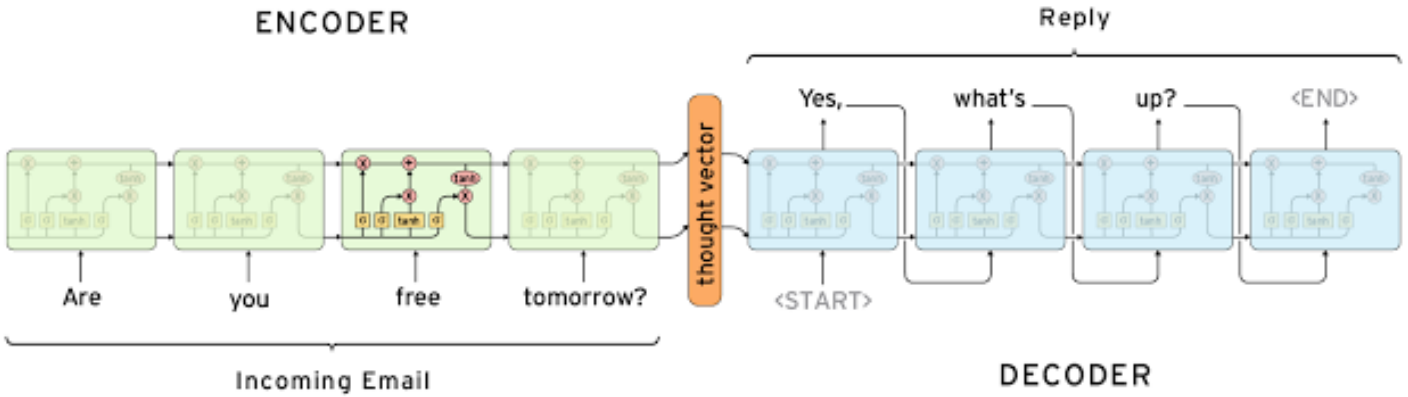
- 更聪明的聊天机器人
  1. 生成式模型 VS 检索匹配模型
  2. Chatterbot的进化：深度学习提高智能度
- 模型构建
  1. 问题的分析与转化
  2. 数据集与样本构造方法
  3. 网络结构的构建
  4. 模型的评估
  5. 代码实现与解析

聊天机器人

- 基于检索的chatbot



- 基于生成模型的chatbot



聊天机器人的一些思考

- 基于检索的chatbot
  - 根据input和context，结合知识库的算法得到合适回复
  - 从一个固定的数据集中找到合适的内容作为回复
  - 检索和匹配的方式有很多种
  - 数据和匹配方法对质量有很大影响
- 基于生成模型的chatbot
  - 典型的是seq2seq的方法
  - 生成的结果需要考虑通畅度和准确度
- 以前者为主(可控度高)，后者为辅
- 深度学习发挥什么作用？
  - 需要算法的地方就可以考虑深度学习的优势

回顾chatterbot

- 机器人应答逻辑 => Logic Adapters
  - Closest Match Adapter
    - 字符串模糊匹配(编辑距离)
  - Closest Meaning Adapter
    - 借助nltk的WordNet，近义词评估
  - Time Logic Adapter
  - …

chatterbot的问题

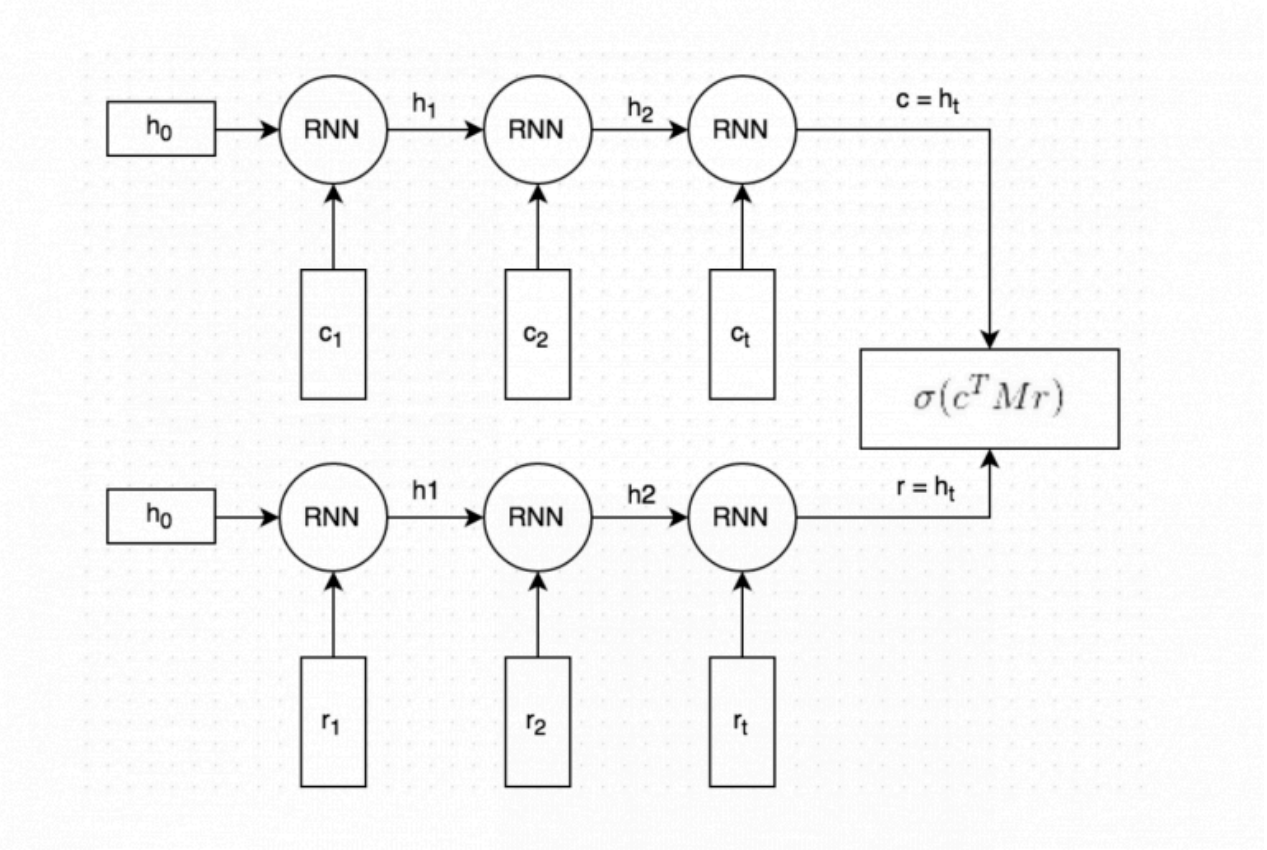
- 应答模式的匹配方式太粗暴
  - 编辑距离无法捕获深层语义信息
  - 核心词 + word2vec无法捕获整句话语义
  - LSTM等RNN模型能捕获序列信息
  - …

- 用深度学习来提高匹配阶段准确率！！

应该怎么做

- 匹配本身是一个模糊的场景
  - 转成排序问题
- 排序问题怎么处理？
  - 转成能输出概率的01分类问题
- 数据构建？
  - 我们需要正样本(正确的回答) 和 负样本(不对的回答)
- Loss function？
  - 回忆一下logistic regression

用深度学习来完成



IMPLEMENTING A RETRIEVAL-BASED MODEL IN TENSORFLOW，WILDML BLOG，2016

关于数据

- Ubuntu对话语料库
- 训练集：
  - Ubuntu对话数据集，来自Ubuntu的IRC网络上的对话日志
  - 训练集1000000条实例，一半是正例（label为1），一半是负例（label为0，负例为随机生成）
  - 样本包括上下文信息(context，即Query)和一段可能的回复内容，即Response；Label为1表示Response和Query的匹配，Label为0则表示不匹配。
  - query的平均长度为86个word，而response的平均长度为17个word

关于数据

Ubuntu对话语料库

- 训练集：

```
In [62]: pd.options.display.max_colwidth = 500
         train_df.head()
```

Out[62]:

	Context	Utterance	Label
0	i think we could import the old comment via rsync , but from there we need to go via email . i think it be easier than cach the status on each bug and than import bite here and there __eou__ __eot__ it would be veri easi to keep a hash db of message-id __eou__ sound good __eou__ __eot__ ok __eou__ perhap we can ship an ad-hoc apt_preferec __eou__ __eot__ version ? __eou__ __eot__ thank __eou__ __eot__ not yet __eou__ it be cover by your insur ? __eou__ __eot__ yes __eou__ but it 's realli no...	basic each xfree86 upload will not forc user to upgrad 100mb of font for noth __eou__ no someth i do in my spare time . __eou__	1
1	i 'm not suggest all - onli the one you modifi . __eou__ __eot__ ok , it sound like you re agre with me , then __eou__ though rather than " the one we modifi '' , my idea be " the one we need to merg '' __eou__ __eot__	sorri __eou__ i think it be ubuntu relat . __eou__	0
2	afternoon all __eou__ not entir relat to warti , but if grub-instal take 5 minut to instal , be this a sign that i should just retri the instal :) __eou__ __eot__ here __eou__ __eot__ you might want to know that thinic in warti be buggi compar to that in sid __eou__ __eot__ and appar gnome be sudden almost perfect (out of the thinic problem) , nobodi report bug : -p __eou__ i do n't get your question , where do you want to past ? __eou__ __eot__ can i file the panel not link to ed ? :) ...	yep . __eou__ oh , okay . i wonder what happen to you __eou__ what distro do you need ? __eou__ yes __eou__	0
3	interest __eou__ grub-instal work with / be ext3 , fail when it be xfs __eou__ i think d-i instal the relev kernel for your machin . i have a p4 and it instal the 386 kernel __eou__ holi crap a lot of stuff get instal by default :) __eou__ you be instal vim on a box of mine __eou__ ;) __eou__ __eot__ more like osx than debian ;) __eou__ we have a select of python modul avail for great justic (and python develop) __eou__ __eot__ 2.8 be fix them iirc __eou__ __eot__ pong __eou__ vino will...	that the one __eou__	1
4	and becaus python give mark a woodi __eou__ __eot__ i 'm not sure if we re mean to talk about that public yet . __eou__ __eot__ and i think we be a " pant off '' kind of compani ... : p __eou__ you need new glass __eou__ __eot__ mono 1.0 ? dude , that 's go to be a barrel of laugh for total non-releas relat reason dure hoari __eou__ read bryan clark 's entri about networkmanag ? __eou__ __eot__ there be an accompani irc convers to that one < g > __eou__ explain ? __eou__ i guess you could s...	(i think someon be go to make a joke about .au bandwidth ...) __eou__ especi not if you re use screen ;) __eou__	1

关于数据

- Ubuntu对话语料库
- 验证/测试集：
  - 每个样本，有一个正例和九个负例数据(也称为干扰数据)。
  - 建模的目标在于给正例的得分尽可能的高，而给负例的得分尽可能的低。(有点类似分类任务)
  - 语料做过分词、stemmed、lemmatized等文本预处理。
  - 用NER(命名实体识别)将文本中的实体，如姓名、地点、组织、URL等替换成特殊字符

关于数据

```
In [66]: pd.options.display.max_colwidth = 500
         test_df.head()
```

Out[66]:

	Context	Ground Truth Utterance	Distractor_0	Distractor_1	Dis
0	anyon know whi my stock oneir export env var usernam ' ? i mean what be that use for ? i know of *userbutnot* usernam . my precis instal doe n't export usernam __eou__ __eot__ look like it use to be export by lightdm , but the line have the comment `` // fixm : be this requir ? '' so i guess it be n't surpris it be go __eou__ __eot__ thank ! how the heck do you figur that out ? __eou__ __eot__ https : //bugs.launchpad.net/lightdm/+bug/864109/comments/3 __eou__ __eot__	nice thank ! __eou__	wrong channel for it , but check efnet.org , unoffici page . __eou__	everi time the kernel chang , you will lose video __eou__ yep __eou__	ok

关于数据

- Ubuntu对话语料库

详见ipython notebook

评估准则

- recall@k
  - 常见的Kaggle比赛评判准则
  - 经模型对候选的response排序后，前k个候选中存在正例数据(正确的那个)的占比
  - k值越大，指标值越高，对模型性能的要求越松。

评估准则

- recall@k

```
def evaluate_recall(y, y_test, k=1):
    num_examples = float(len(y))
    num_correct = 0
    for predictions, label in zip(y, y_test):
        if label in predictions[:k]:
            num_correct += 1
    return num_correct/num_examples
```

基线模型：random guess

```
# 随机预测器
def predict_random(context, utterances):
    return np.random.choice(len(utterances), 10, replace=False)

# 评估随机预测器
y_random = [predict_random(test_df.Context[x], test_df.iloc[x,1:].values) for x in range(len(test_df))]
y_test = np.zeros(len(y_random))
for n in [1, 2, 5, 10]:
    print("Recall @ ({}, 10): {:g}".format(n, evaluate_recall(y_random, y_test, n)))
```

```
Recall @ (1, 10): 0.0937632
Recall @ (2, 10): 0.194503
Recall @ (5, 10): 0.49297
Recall @ (10, 10): 1
```

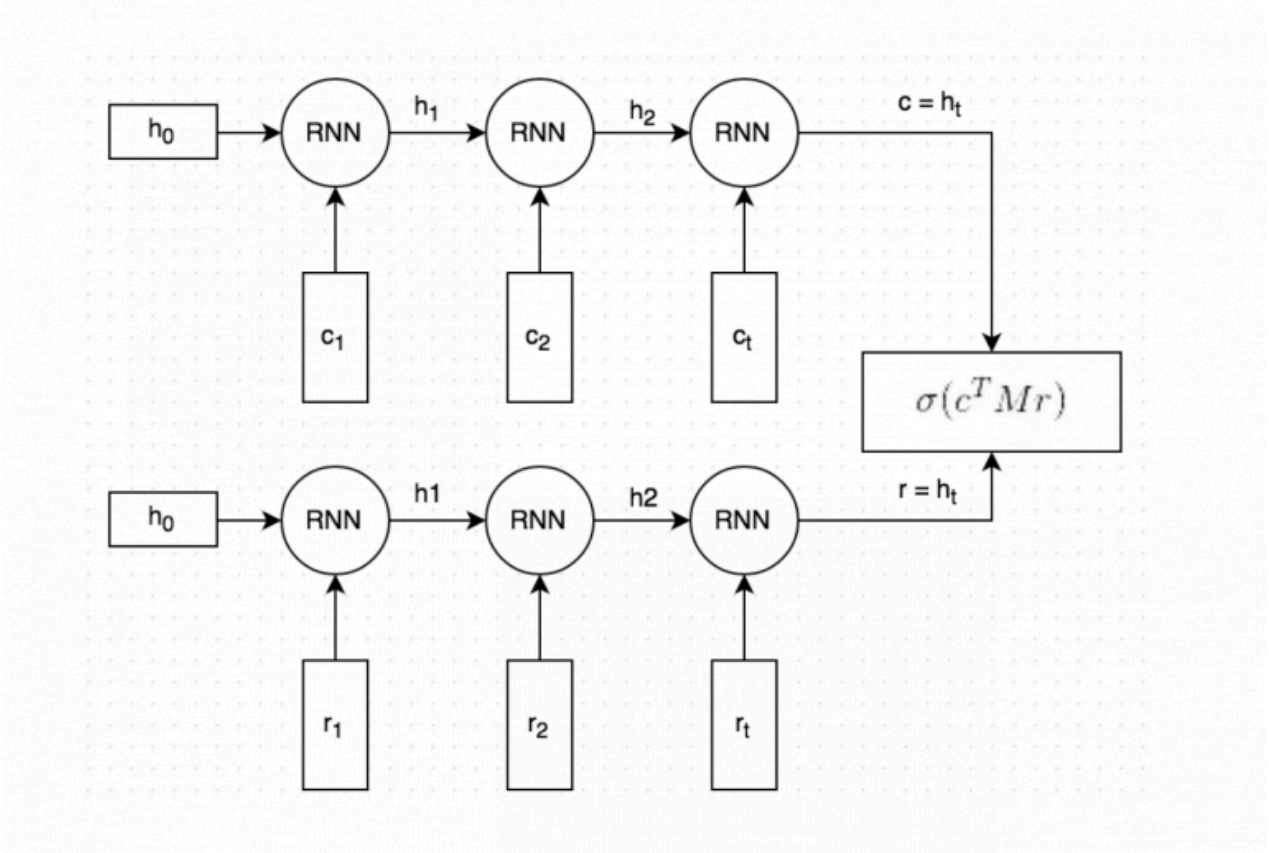
基线模型：TF-IDF检索

```
class TFIDFPredictor:
    def __init__(self):
        self.vectorizer = TfidfVectorizer()

    def train(self, data):
        self.vectorizer.fit(np.append(data.Context.values,data.Utterance.values))

    def predict(self, context, utterances):
        # 把文本内容转成tf-idf向量
        vector_context = self.vectorizer.transform([context])
        vector_doc = self.vectorizer.transform(utterances)
        # 评估向量之间的相近程度
        result = np.dot(vector_doc, vector_context.T).todense()
        result = np.asarray(result).flatten()
        # 排序输出
        return np.argsort(result, axis=0)[::-1]
```

基线模型：TF-IDF检索

```
# 评估TF-IDF预估器
pred = TFIDFPredictor()
pred.train(train_df)
y = [pred.predict(test_df.Context[x], test_df.iloc[x,1:].values) for x in range(len(test_df))]
for n in [1, 2, 5, 10]:
    print("Recall @ ({}, 10): {:g}".format(n, evaluate_recall(y, y_test, n)))

Recall @ (1, 10): 0.495032
Recall @ (2, 10): 0.596882
Recall @ (5, 10): 0.766121
Recall @ (10, 10): 1
```

神经网络建模



IMPLEMENTING A RETRIEVAL-BASED MODEL IN TENSORFLOW，WILDML BLOG，2016

神经网络建模

(1) Query和Response都是经过分词 和 embedding映射的。初始词向量使用GloVe/word2vec。

(2) 分词且向量化的Query和Response经过相同的RNN(word by word)。RNN最终生成一个向量表示，捕捉了Query和Response之间的[语义联系](图中的c和r)；这个向量的维度是可以指定的，这里指定为256维。

(3) 将向量c与一个矩阵M相乘，来预测一个可能的回复r'。如果c为一个256维的向量，M维256*256的矩阵，两者相乘的结果为另一个256维的向量，我们可以将其解释为[一个生成式的回复向量]。矩阵M是需要训练的参数。

神经网络建模

(4) 通过点乘的方式来预测生成的回复r'和候选的回复r之间的相似程度，点乘结果越大表示候选回复作为回复的可信度越高；之后通过sigmoid函数归一化，转成概率形式。(sigmoid作为压缩函数经常使用)

(5) 损失函数：二元的交叉熵(binary cross-entropy) 函数/对数损失函数。回想逻辑回归，交叉熵损失值为L = -y * ln(y') - (1 - y) * ln(1 - y')。

- 公式的意义是直观的，即当y=1时，L = -ln(y')，我们期望y'尽量地接近1使得损失函数的值越小；反之亦然。

神经网络建模

数据预处理、网络搭建，Tensorflow训练与评估

请见课堂代码详解

感谢大家么么哒！

恳请大家批评指正！