[**一个Hierarchical Attention神经网络的实现**](https://blog.csdn.net/triplemeng/article/details/78269127)

标签： [神经网络](http://so.csdn.net/so/search/s.do?q=%E7%A5%9E%E7%BB%8F%E7%BD%91%E7%BB%9C&t=blog) [nlp](http://so.csdn.net/so/search/s.do?q=nlp&t=blog" \t "_blank) [attention](http://so.csdn.net/so/search/s.do?q=attention&t=blog) [tensorflow](http://so.csdn.net/so/search/s.do?q=tensorflow&t=blog" \t "_blank)

2017年10月18日 09:12:23404人阅读 [评论](javascript:void(0);)(0) [收藏](javascript:void(0);) [举报](javascript:void(0);)

https://csdnimg.cn/release/phoenix/images/category_icon.jpg 分类：

深度学习，人工智能（1） https://csdnimg.cn/release/phoenix/images/arrow_triangle%20_down.jpg 自然语言处理，注意力模型

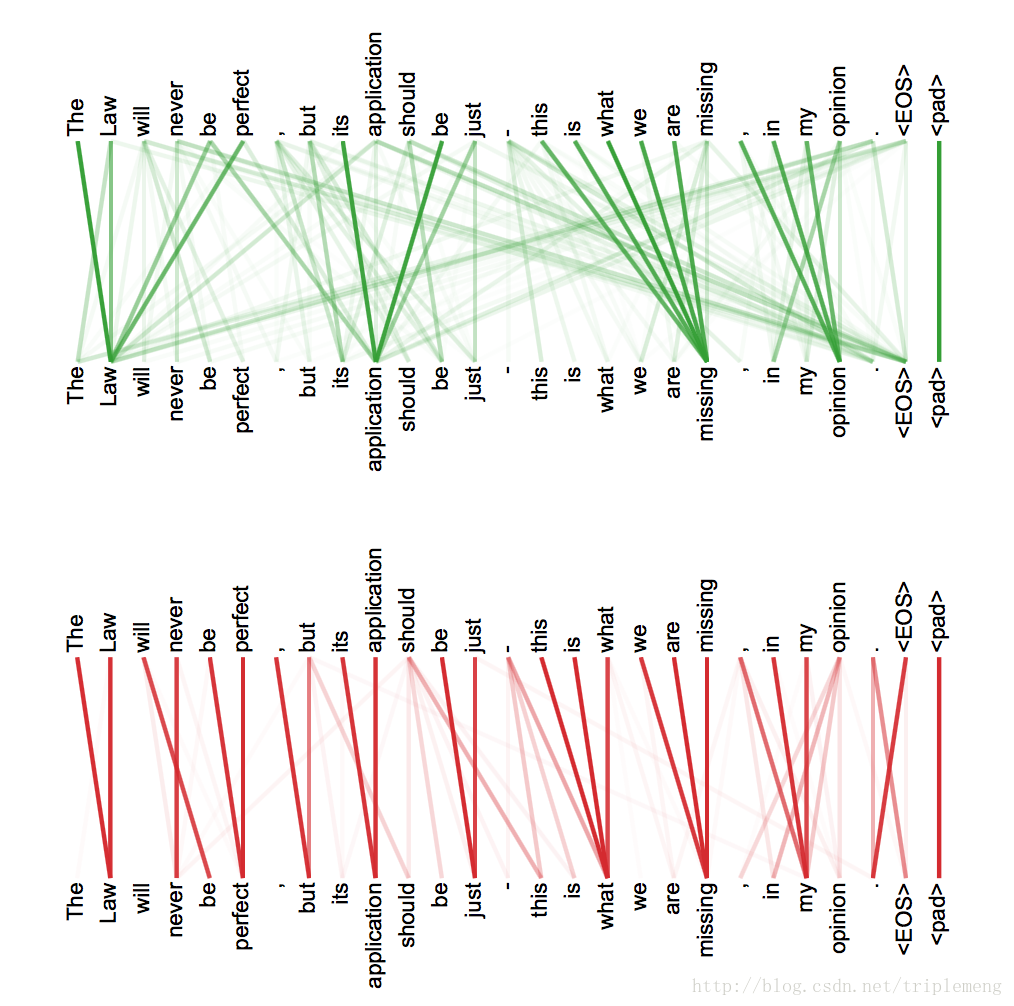
最近我突然有了一些富余的整块时间。于是我实现了一些有意思的论文的idea, 其中印象最深的还是《Hierarchical Attention Networks for Document Classification》。我把相关代码放到这里了：   
<https://github.com/triplemeng/hierarchical-attention-model>

**综述**

今天，基本上所有的NLP方面的应用，如果想取得state-of-art的结果，就必须要经过attention model的加持。比如machine translation, QA(question-answer), NLI(natural language inference), etc, etc…. 但是这里有个问题: 传统上的attention的应用，总是要求我们的task本身同时有源和目标的概念。比如在machine translation里, 我们有源语言和目标语言，在QA里我们有问题和答案，NLI里我们有sentence pairs …… 而Attention常常被定义为目标和源的相关程度。

但是还有很多task不同时具有源和目标的概念。比如document classification, 它只有原文，没有目标语言/文章， 再比如sentiment analysis（也可以看做是最简单的一种document classification）,它也只有原文。那么这种情况下attention如何展开呢？ 这就需要一个变种的技术，叫intra-attention（或者self-attention）, 顾名思义，就是原文自己内部的注意力机制。

intra-attention有不同的做法。比如前一段Google发的那篇《Attention is All You Need》，在machine translation这个任务中，通过把attention机制formularize成Key-Value的形式，很自然的表达出源语言和目标语言各自的intra-attention. 这么做的好处是在句子内部产生清晰的1.语法修饰 2. 语义指代关系 方面的理解，也就是说对句子的结构和意义有了更好的把控。

如下盗图所示：   


如上两图分别代表了原文在两个不同子空间的投影的注意力结果(具体做法见原Google的论文，这里的介绍从略)。我们可以看到清晰的指代关系和修饰关系。文章引入的self-attention机制加上positional embedding的做法，可能是将来的一个发展方向。

**方法**

这篇论文采取了非常不一样的做法。它引入了context vector用来发现每个词语和每个句子的重要性。

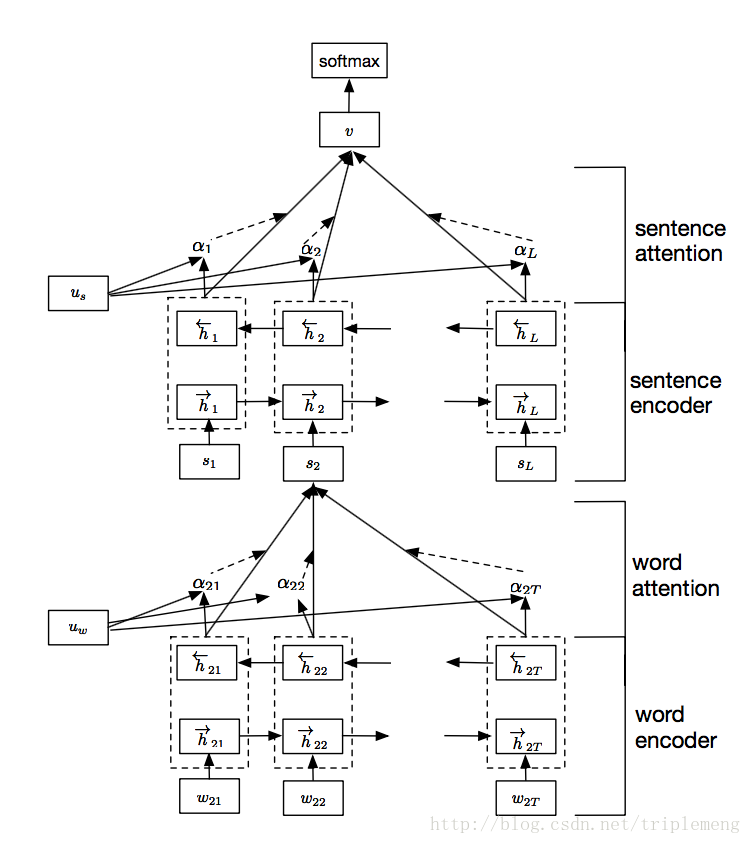
它基于这样的observation:

每个document由多个句子组成，而在决定文章的类型时，每个句子有不同的重要性。有的更相关一些，有的用处不大。比如说在一篇有关动物科学的文章中，某些句子和文章的主题相关性就很高。比如包含类似于“斑马”或者“猎食者”，“伪装”这样词语的句子。我们在建造模型时，最好能够给这样的句子更多的“attention”。 同样的，对于每个句子而言，它所包含的每个词语的重要性也不一样，比如在IMDB的review中， 如like, amazing, terrible这样的词语更能够决定句子的sentiment

所以，在分类任务中，如果我们给模型一篇文章，我们想问模型的问题是：1. 在这篇文章中，哪些句子更重要，能够决定它的分类？ 2. 在这篇文章的某个句子中，哪些词语最重要，能够影响句子在文章里的重要性？

那么，如何向模型提出这样的问题呢? 或者说如何让模型理解我们的意图呢? 作者是通过引入context vector做到的。这个context vector 有点天外飞仙的感觉, 之所以给我这样的感觉，是因为：

1. context vector是人工引入的，它不属于task的一部分。它是随机初始化的。
2. 它代替了inter-attention中目标语言/句子，能够和task中的原文产生相互作用，计算出原文各个部分的相关程度，也就是我们关心的attention。
3. 它是jointly learned。 也就是说，它本身，也是学习得来的 ！！

具体而言，网络的架构如下：   


网络由四个部分组成：word sequence layer, word-attention layer, sentence sequence layer, and sentence-attention layer

如果没有图中的 uw(词语级别的context vector)和 us(句子级别的context vector)，这个模型也没有什么特殊的地方。它无非是由word sequence layer和sentence sequence layer组成的一个简单的层级的sequence模型而已。而有了这两个context vector, 我们就可以利用它们产生attention layer, 求出每个词语和每个句子的任务相关程度。

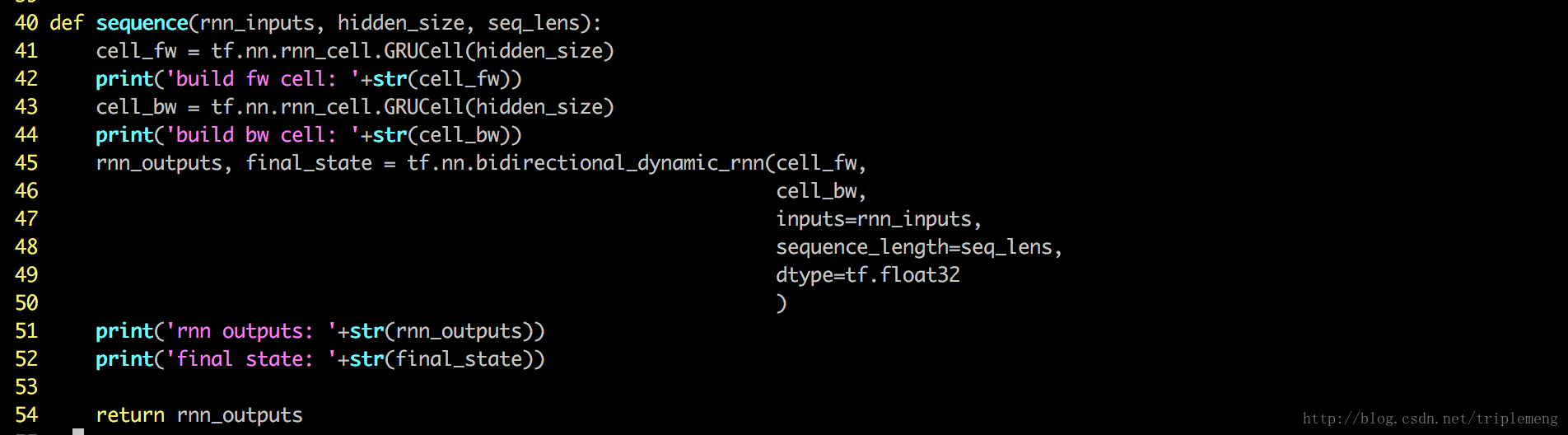
具体做法如下，针对每一个句子，用sequence model, 就是双向的rnn给表达出来，在这里用的是GRU cell。每个词语对应的hidden vector的输出经过变换（affine+tanh）之后和uwuw相互作用（点积），结果就是每个词语的权重。加权以后就可以产生整个sentence的表示。从高一级的层面来看(hierarchical的由来)，每个document有L个句子组成，那么这L个句子就可以连接成另一个sequence model, 同样是双向GRU cell的双向rnn，同样的对输出层进行变换后和 us相互作用，产生每个句子的权重，加权以后我们就产生了对整个document的表示。最后用softmax就可以产生对分类的预测。

每次的“提问”，都是由 uw和 us来实现的，它们用来找到高权重的词语和句子。下面看看实现。

**实现**

下面是我用tensorflow的实现。具体见我的github: <https://github.com/triplemeng/hierarchical-attention-model>

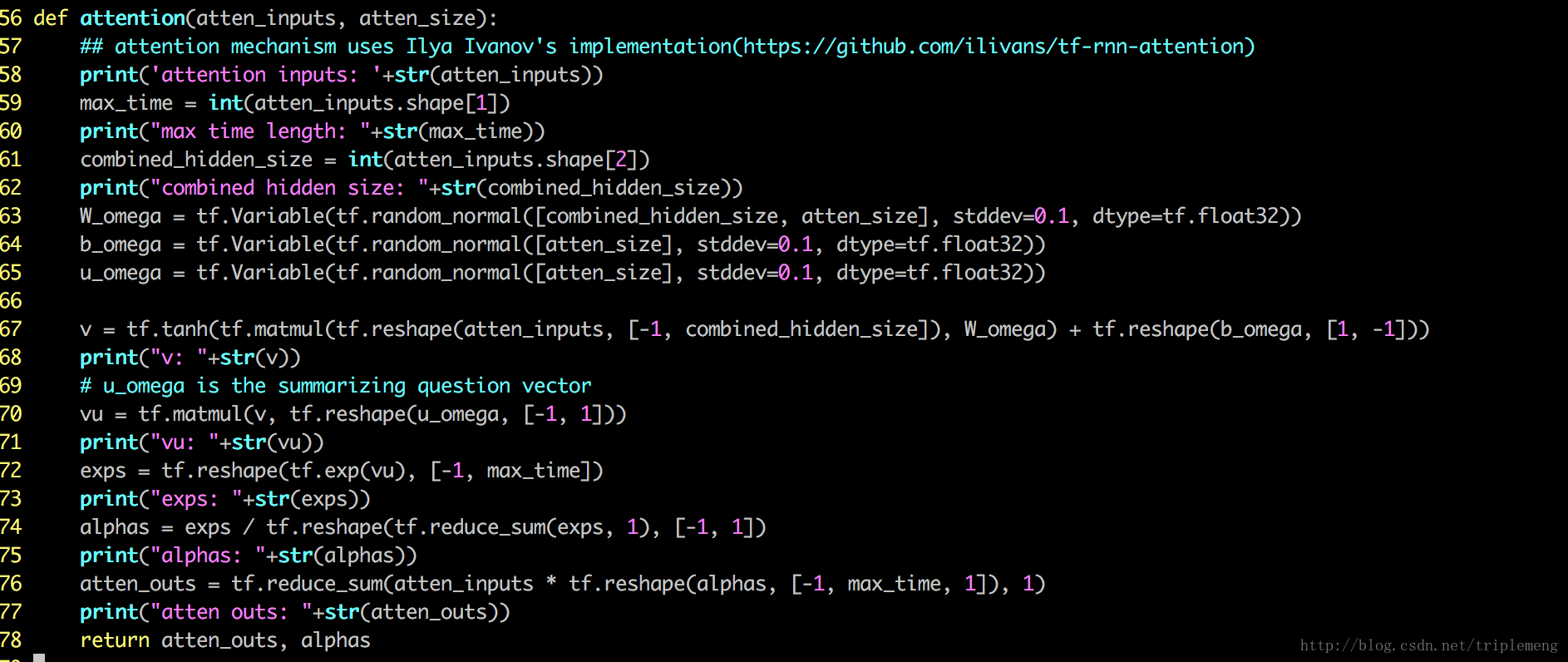
word-sequence和sentence-sequence都通过下面这个module实现。



这里使用了双向的dynamic的rnn，实际上static的rnn效果也不错

不同的IMDB的reviews大小不一样，有的包含了几十个句子，有的只包含了几个句子, 为了让rnn模型更加精确，我把每个batch内部的review的实际长度（review所包含的句子个数）存在了seq\_lens里面。这样在调用bidirectional\_dynamic\_rnn时，rnn精确的知道计算该在哪里停止。

word-attention和sentence-attention都通过调用这个module来实现



注意第65行  uw就是我们说的context vector (我在第69行把它称之为summarizing question vector)

结果

根据原paper的做法，利用data先pretrain了word embeddings

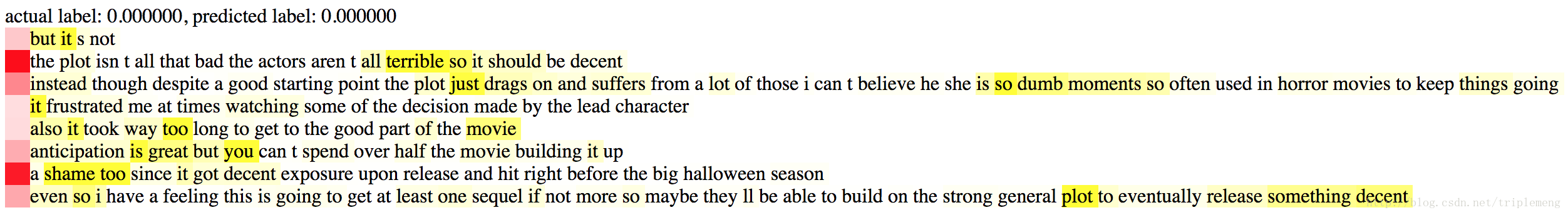
sequence层之后加了dropout, 用处不甚明显。

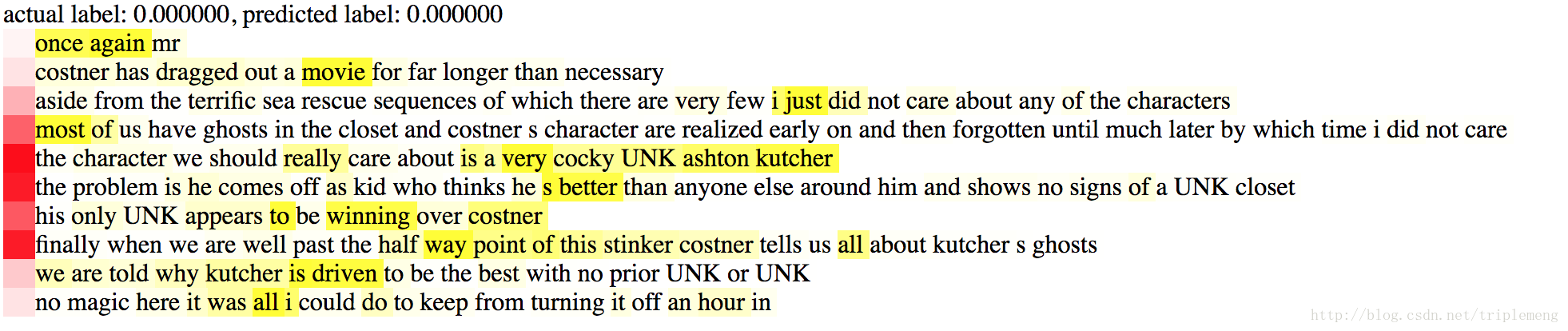
最大的review长度控制在15个句子， 每个句子的长度固定为70。

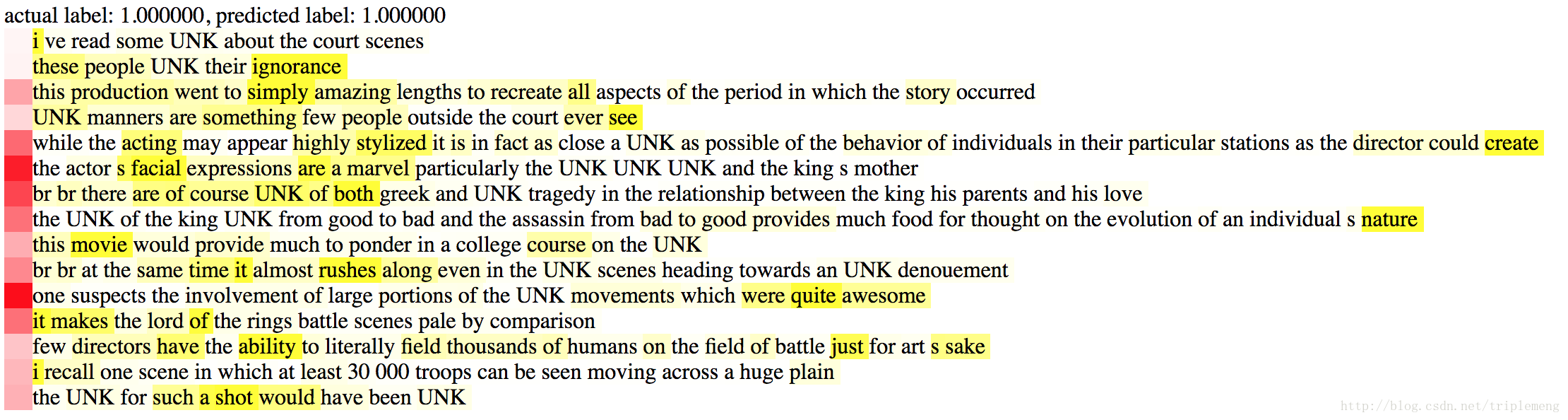
最后的大概精度是0.9左右。我现在的计算资源非常有限，所以不能实验太多别的参数。

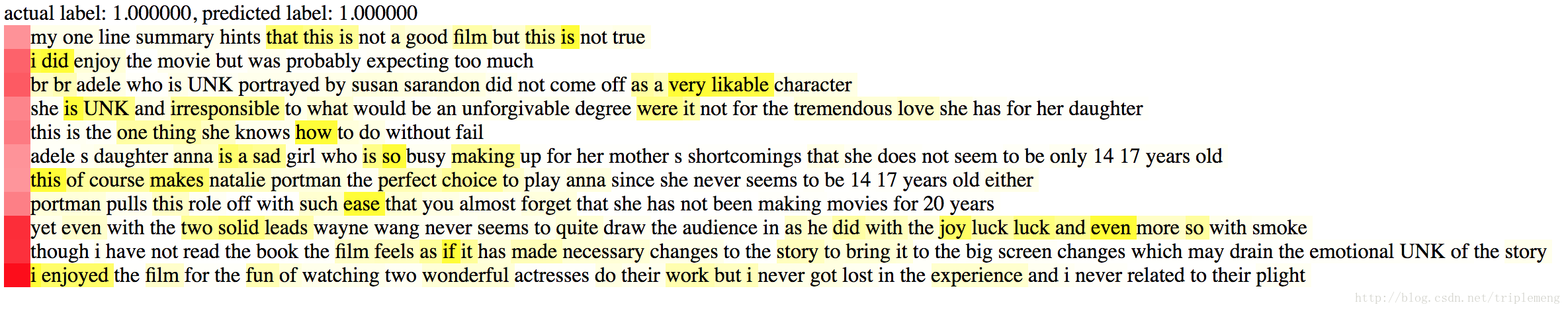
以下是几个例子。红色的深度代表句子对于review的sentiment的重要性，黄色的深度代表词语对于句子的重要性。

0代表negative, 1代表positive



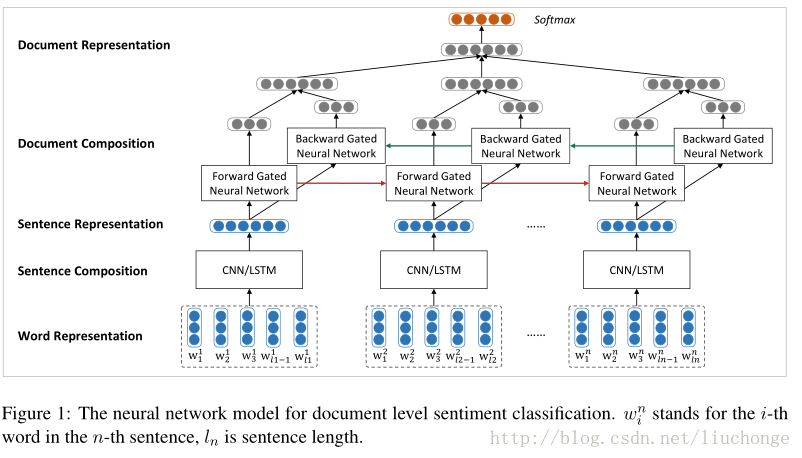






总结一下： 这种 intra-attention 机制很有创意，也挺有效。根据我不完全的实验，hierarchical attention model 明显好于其他的比如stacked bi-directional RNN, CNN text(Yoon Kim的版本)。 说不完全是因为我的计算资源非常有限，我只是实现了这些模型，没有仔细的调参, 而且我只对2-class的IMDB作了实验。 所以上面的结论不够严谨，仅供参考。

我当然相信attention的效果，但是比较无法忍受sequence model的速度。最近新的突破（SRU）或许能够大幅度提升sequence model的效率，但是我还是想试试CNN+attention， 原因是一来比较看好CNN的速度，而来attention可以一定程度上弥补CNN在长程相关性上的缺陷。



https://zhuanlan.zhihu.com/p/26892711

在componet里面定义了sequence 和 attention 两个模块

在models里面build graph的时候，

先用一个embeddings 对词进行向量化**词编码器，reshape之后得到** word\_rnn\_inputs\_formatted，作为sequence的输入，

word\_rnn\_outputs = sequence(word\_rnn\_inputs\_formatted, hidden\_size, None)

在sequence模块中用双向的GRU网络,可以将正向和反向的上下文信息结合起来，获得隐藏层输出。

然后把隐藏层拼在一起atten\_inputs = tf.concat(word\_rnn\_outputs, 2)，作为attention模块的输入。

后面句子层面类似