自然语言处理系列（2）：Word2Vec

这篇文章翻译自word2vec Parameter Learning Explained

Mikolov等人的word2vec模型和应用在近两年受到了广泛的关注。由word2vec模型学习的单词的向量表示已经被证明具有语义意义，并且在各种NLP任务中都很有用。随着越来越多的研究人员实验word2vec或类似的技术,我注意到缺乏全面解释字嵌入模型的参数学习过程的细节的材料,从而使得非神经网络专家的研究人员无法理解此类模型的工作机制。

本文章提供了word2vec模型参数更新方程的详细推导和解释，包括原始的连续bag-of-word (CBOW)和skip-gram (SG)模型，以及优化技术，包括分层的softmax和负采样。对梯度方程的直观解释也提供了数学推导。

在附录中，提供了关于神经元网络和反向传播的基础知识的综述。我还创建了一个交互式演示，wevi，以促进对模型的直观理解。

一、Continuous Bag-of-Word Model

1.1 One-word context

我们从Mikolov et al. (2013a)中引入的最简单的一种连续的单词模型(CBOW)开始。我们假设每个上下文只考虑一个词，这意味着模型将根据上下文单词来预测一个目标单词，这就像一个三元模型。对于不熟悉神经网络的读者来说，建议通过附录A来快速回顾重要的概念和术语，然后再进一步讨论。

图1显示了简化的上下文定义下的网络模型。我们设定词汇量大小为V，隐藏层的大小为N。相邻层上的单元是全连接的。输入是一个one-hot编码向量，这意味着对于给定的输入上下文，在V个单元x_1，·，·，x_V中只有一个单元为1，所有其他单元都是0。输入层和输出层之间的权值可以表示为一个$V \times N$矩阵W。W的每一行是与输入层的相关词的N维向量表示v_w。形式上，W的第i行是v_w^T。给定一个上下文(一个单词)，假设$x_k = 1$、$x_{k'} = 0$（$k \neq k$）,我们有

$$h = W_T \cdot x = W^T_{(k,:)} := v^T_{w_I} \cdot \cdot \cdot \cdot \cdot \cdot \quad (1)$$

本质上就是复制W的第k行至h。v_{wI}是输入词w_I的向量表示。这意味着隐含层单元的链接(激活)函数是线性的(即:，直接将其和输入加权求和至下一层。

从隐层到输出层,有一个不同的权重矩阵$W' = w'_{ij}$，这是一个$N \times V$的矩阵。使用这些权重，我们可

以计算词库中每个单词的得分u_j。

$$u_j = v'^{T}_{w_j} \cdot h \cdot \cdots \quad (2)$$

v'_w是矩阵W'的第j列，然后我们可以使用softmax，一个对数线性j分类模型，来获取单词的后验分布，这是一个多项分布。

$$p(w_j|w_I) = y_j = \frac{exp(u_j)}{\sum_{j'=1}^{V} exp(u_{j'})}, \cdots \quad (3)$$

其中y_j是输出层的第j个单元的输出。将(1)和(2)代入(3)，得到。

$$p(w_j|w_I) = \frac{exp(v'^{T}_{w_j} v_{w_I})}{exp(v'^{T}_{w_j} v_{w_I})} \cdot \cdots \quad (4)$$

注意v_w和v'_w是词w的两种表示。v_w来自矩阵W的行，是输入层至隐藏层的权重矩阵，v'_w来自矩阵W的列，是隐藏层到输出层的权重矩阵。在随后的分析中，我们把v_w称为“输入向量”，把v'_w称为词w的“输出向量”。

> 隐藏层到输出层权重的更新公式

现在让我们推导这个模型的权值更新方程。虽然实际的计算是不切实际的(解释如下)，但我们正在做以下的推导，以获得对这个原始模型的见解，并没有使用任何技巧。有关反向传播的基础知识，请参阅附录a。

训练目标(一个训练样本)是最大化(4)式，在给定输出上下文单词w_I的条件下，观察实际输出单词w_o的条件概率。

$$maxp(w_O|w_I) = maxy_{j^*} \quad (5) \qquad = maxlogy_{j^*} \quad (6) \qquad = u_{j^*} - log \sum_{j'=1}^{V} exp$$

其中$E = -logp(w_O|w_I)$是我们的损失函数（我们希望最小化E），j^*是输出层中的实际输出词的索引。注意，这个损失函数可以理解为两个概率分布之间的交叉熵衡量的一个特例。

现在让我们推导出隐藏和输出层之间权重的更新方程。求E关于第j个单位网络输入u_j的导数，我们得到：

$$\frac{\partial E}{\partial u_j} = y_j - t_j := e_j (8)$$

比如$t_j = 1(j = j^*)$，只有当第j个单元是实际输出词时，t_j为1，否则$t_j = 0$。注意，这个导数只是输出层的预测误差e_j。

接下来我们对$w_{ij}^{'}$求导获得从隐藏层到输出层的梯度。

$$\frac{\partial E}{\partial w_{ij}^{'}} = \frac{\partial E}{\partial u_j} \cdot \frac{\partial u_j}{\partial w_{ij}^{'}} = e_j \cdot h_i \qquad (9)$$

因此,使用随机梯度下降法,得到隐藏的权重更新方程→输出权值:

$$w_{ij}^{'(new)} = w^{'(old)}_{ij} - \eta \cdot e_j \cdot h_i \qquad (10)$$

或者

$$v_j^{'(new)} = v_{w_j}^{'(old)} - \eta \cdot e_j \cdot h \qquad forj = 1, 2, \cdots, V. \qquad (11)$$

其中$eta > 0$为学习率，$e_j = y_j - t_j$，h_i是隐藏层的第i个单元。$v_{w_j}^{'}$是w_j的输出向量。注意，这个更新方程意味着我们必须遍历词汇表中的每个可能单词，检查它的输出概率y_j，并将y_j与它的期望输出t_j(0或1)进行比较。如果$y_j > t_j$(即高估)，我们就从$v_{w_j}^{'}$移除一部分的隐藏层向量h(比如v_{w_I})，使得$v_{w_j}^{'}$远离v_{w_I}；如果$y_j < t_j$(即低估，当且仅当$t_j = 1$时成立，比如$w_j = w_O$)，我们就添加一些h至$v_{w_O}^{'}$，使得$v_{w_O}^{'}$靠近v_{w_I}。如果y_j非常接近t_j，根据更公式，权重只会发生非常小的变化。再次注意，v_w(输入向量)和$v_w^{'}$(输出向量)是词w两种不同的向量表示。

> 输入层到隐藏层的权重更新公式

已经获取了$W^{'}$的更新公式，现在我们来看看W。我们对E求隐藏层的输出的导数，得到：

$$\frac{\partial E}{\partial h_i} = \sum_{j=1}^{V} \frac{\partial E}{\partial u_j} \cdot \frac{\partial u_j}{\partial h_i} = \sum_{j=1}^{V} e_j \cdot w_{ij}^{'} := EH_i \qquad (12)$$

其中h_i是隐藏层第i个单元的输出；u_j是在(2)式中被定义的，即输出层的第j个单元的网络输入；$e_j = y_j - t_j$是输出层中第j个词的预测误差。EH是一个N维的向量，是词库中所有单词的输出向量的预测误差加权求和。

接下来我们对E求W的导数，首先，回想一下隐藏层对输入层的值进行线性计算。展开（1）中的

向量表示，我们得到

$$h_i = \sum_{k=1}^{V} x_k \cdot w_{k_i}$$

现在我们可以求E关于W的每个元素的导数，得到:

$$\frac{\partial E}{\partial w_{k_i}} = \frac{\partial E}{\partial h_i} \cdot \frac{\partial h_i}{\partial w_{k_i}} = EH_i \cdot x_k \qquad (14)$$

这是x和EH的点积等式，

$$\frac{\partial E}{\partial W} = x \otimes EH = xEH^T \qquad (15)$$

这样我们得到了一个$V \times N$的矩阵。因为x向量中只有一个元素非零，$\frac{\partial E}{\partial W}$中只有一行非零，且那一行 的值为$EH^T$，一个N维的向量。我们得到了w的更新公式：

$$v_{w_I}^{(new)} = v_{w_I}^{(old)} - \eta EH^T \qquad (16)$$

其中v_{w_I}是W的一行，及唯一的上下文词的输入向量，也是唯一的导数不为零的W的行，所有其他的W的行在迭代后保持不变，因为他们的导数为零。

从直觉上来看，因为向量EH是语料库中所有单词的输出并进行预测误差加权后得到的总和，我们可以将（16）理解成为词汇中的每个输出向量的一部分添加到上下文单词的输入向量中。如果在输出层中，一个单词wj作为输出词的概率被高估($y_j > t_j$)，那么上下文单词w_I的输入向量将倾向于远离w_j的输出向量。反之，如果w_j作为输出词的概率被低估($y_j < t_j$)，则输入向量w_I将趋向于接近w_j的输出向量。如果w_j的概率相对准确的预测，那么它对w_I的输入向量的改变量很小。w_I的输入向量的变动是由词汇中所有向量的预测误差决定的。预测误差越大，一个单词对上下文单词输入向量的变动产生的影响就越大。当我们通过从训练语料库中生成的上下文目标词对来迭代更新模型参数时，对向量的影响将会累积。我们可以想象一个单词w的输出向量被w的相邻邻域的输入向量“拖拽”，就好像w的向量和它的邻域的向量之间有物理的弦。同样，输入向量也可以被认为是被许多输出向量拖拽的。这种解释可以提醒我们注意重力，或者是力向图的布局。每个虚弦的平衡长度与相关的两个单词之间的共存强度以及学习速率有关。经过多次迭代，输入和输出向量的相对位置最终会趋于稳定。

1.2 Multi-word context

图2显示了带有多词上下文的CBOW模型。当计算隐层输出,而不是直接复制的输入向量输入上下文的话,CBOW模型需要的平均向量的输入上下文的话,和使用的产品输入→隐藏权重矩阵和平均向量作为输出。

$$h = \frac{1}{C}W^T(x_1 + x_2 + \cdots + x_C) \qquad (17) = \frac{1}{C}(v_{w_1} + v_{w_2} + \cdots + v_{w_C})^T \qquad (18)$$

其中C是上下文中词汇的数量，$w_1, \cdots, w_C$是上下文的词汇，v_w是一个词w的输入向量。损失函数为：

$$E = -logp(w_O|w_{I,1}, \cdots, w_{I,C}) \qquad (19) = -u_{j^*} + log\sum_{j'=1}^{V} exp(u_j') \qquad (20) = -v_{w_O}^T \cdot h + log\sum_{j'=1}^{V} exp($$

可以看到和（7）式one-word-context模型的目标函数相同，除了h不同，将（1）式替换为（18）式。

从隐藏层到输出层权重更新公式和one-word-context模型保持一致，copy如下：

$$v_j^{'(new)} = v_{w_j}^{'(old)} - \eta \cdot e_j \cdot h \qquad forj = 1, 2, \cdots, V. \qquad (22)$$

注意,我们需要对隐藏层→输出层的权重矩阵的每个元素为每个训练实例进行更新。

输入层到隐藏层的权重更新公式与(16)类似,只是现在我们需要应用下列方程至上下文的每一个词:

$$v_{w_I}^{(new)} = v_{w_I}^{(old)} - \frac{1}{C}\eta EH^T \qquad for\ j = 1, 2, \cdots, \quad C \qquad (23)$$

其中$v_{w_{I,c}}$是在输入上下文中第c个词的输入向量；η是一个正的学习率；$EH = \frac{\partial E}{\partial h_i}$由（12）式给出。这个更新公式的直观解释和（16）式类似。